Rapport Final

Video To Text

DÉVELOPPEMENT D'UNE APPLICATION DE TRANSCRIPTION AUTOMATIQUE VIDÉO-TEXTE : CONCEPTION ET RÉALISATION DE ''TEXTIFY''

Réalisé Par:

- **Lina AIT IDER**
- **Hayat IMHAH**
- **Imane BAHAMD**
- **Saida EL HAOUARANI**
- **Ezzahra FADYL**

Encadré PAR:

Pr. NEJOUI

Année Universitaire : 2024/2025

Remerciements

Nous tenons à exprimer notre sincère gratitude à tous les membres de notre groupe qui ont contribué à la réalisation de notre projet **"Textify"**. Sans leur engagement et leur collaboration, cette application n'aurait pas pu être finalisée dans les délais impartis.

Tout d'abord, nous souhaitons remercier chaleureusement notre professeur, **M. Abderrazzak Nejeoui**, pour son encadrement, ses réponses précises à nos questions concernant le projet, et son soutien constant tout au long de sa réalisation. Sa passion pour l'enseignement et son expertise ont été une source d'inspiration, nous poussant à approfondir nos connaissances en développement d'applications multimédias avec la langue Java.

Enfin, nous adressons notre reconnaissance à notre établissement, l'**École Nationale des Sciences Appliquées de Marrakech (ENSAM)**, pour avoir mis à notre disposition un environnement d'apprentissage stimulant ainsi que les ressources nécessaires pour mener à bien ce projet.

Table de matière

Table des Annexes

Résumé :

Le projet consiste en la création d'une application de traitement multimédia permettant à l'utilisateur de convertir des fichiers MP4 en MP3, de télécharger des vidéos YouTube et de les convertir en MP3, et d'extraire le texte d'un fichier audio via une transcription.

Pour cela, plusieurs technologies ont été utilisées. **FFmpeg** a été intégré pour assurer la conversion des fichiers multimédia, **yt-dlp** pour télécharger les vidéos YouTube, et l'API **AssemblyAI** a été utilisée pour la transcription audio. L'interface utilisateur a été développée en **JavaFX**, garantissant une expérience simple et intuitive pour l'utilisateur.

Le système permet une interaction fluide et efficace avec l'utilisateur, tout en offrant une gestion robuste des erreurs et une architecture modulaire pour une future évolution. Le projet répond ainsi à un besoin croissant de solutions multimédia polyvalentes et accessibles.

Mots-clés :

· Conversion MP4 en MP3

· Téléchargement vidéo YouTube

· Transcription audio

· FFmpeg

· Yt-dlp

· API AssemblyAI

· JavaFX

· Interface utilisateur

· Gestion des erreurs

· Architecture modulaire

· Traitement multimédia

· Solution multimédia

· Programmation Java

Introduction :

L'application **Textify** a été développée dans le cadre d'un projet visant à simplifier et automatiser la transcription de vidéos en texte. Cette solution permet aux utilisateurs de télécharger des fichiers vidéo ou de fournir des liens YouTube afin d'effectuer une conversion rapide et précise des contenus audio en texte écrit. L'objectif de ce projet est de rendre accessible le contenu vidéo à travers des transcriptions textuelles, ce qui peut être particulièrement utile dans des contextes tels que la création de sous-titres, l'indexation de contenu pour la recherche, ou la simplification de la prise de notes.

Pour répondre à cette problématique, **Textify** exploite des technologies modernes et des bibliothèques robustes. L'application a été développée en utilisant JavaFX pour la création d'une interface utilisateur graphique (GUI) intuitive et interactive, facilitant l'expérience de l'utilisateur lors de l'importation des fichiers ou de la saisie de liens vidéo. L'interface est conçue de manière à être simple, claire et ergonomique, permettant une utilisation rapide sans nécessiter une expertise technique.

Au niveau de la gestion des fichiers multimédias, le projet met en œuvre un traitement dynamique des vidéos téléchargées. En effet, **Textify** utilise une combinaison de bibliothèques pour extraire l'audio des vidéos et le convertir en texte via un système de reconnaissance vocale. Cela permet une automatisation du processus de transcription, rendant l'application particulièrement efficace pour les utilisateurs ayant besoin d'obtenir des transcriptions fiables en peu de temps.

Ainsi, **Textify** représente une avancée importante dans le domaine de l'accessibilité numérique et de la gestion de contenu multimédia, et démontre l'application des concepts de programmation moderne dans la résolution de problèmes réels liés à la conversion et à la gestion de données audio-visuelles.

Cahier de charge :

3.1. Problématique :

Avec l'essor des plateformes de partage de vidéos comme YouTube, le contenu multimédia devient une source importante d'informations et de savoirs. Cependant, bien que les vidéos soient un format riche et attractif, elles posent un problème majeur pour les personnes qui souhaitent extraire ou traiter le contenu d'une vidéo sans nécessairement la regarder en entier. En effet, l'extraction d'informations utiles à partir d'une vidéo implique généralement une écoute attentive et une compréhension du discours, ce qui peut être chronophage et difficile dans des environnements où il n'est pas possible de visionner la vidéo, comme lors de déplacements ou dans des situations où l'audio ne peut pas être écouté.

De plus, les vidéos comportent souvent un grand nombre de données audio qui peuvent être difficiles à manipuler sans les transformer en texte. La transcription manuelle de l'audio en texte est un processus long, coûteux et sujette à des erreurs humaines. Ce problème est encore plus accentué lorsqu'il s'agit de vidéos en langues étrangères ou avec un discours rapide, rendant l'extraction du contenu encore plus complexe. Dans ce contexte, la problématique centrale du projet est donc de créer une solution automatique et efficace permettant de :

1. Extraire l'audio d'une vidéo en un format facilement manipulable (MP3).
2. Transcrire cet audio en texte de manière fiable et précise, tout en prenant en compte les différents défis tels que les accents, les bruits de fond, ou les interruptions dans le discours.

L'enjeu réside donc dans la mise au point d'un système capable de combiner plusieurs technologies :

— Téléchargement et extraction de l'audio de vidéos, ainsi que l'utilisation d'outils de reconnaissance vocale

— Pour offrir une solution complète et accessible à tout utilisateur souhaitant obtenir rapidement un texte transcrit à partir d'une vidéo. La principale difficulté ici est de garantir la précision et la fluidité du texte généré, tout en s'assurant que l'application reste intuitive et simple à utiliser, même pour un public non technique.

Contexte du projet :

Le projet porte sur la création d'une application permettant de télécharger des vidéos YouTube, de les convertir en format MP3 et de transcrire le contenu audio en texte. Cette solution est particulièrement utile pour les utilisateurs qui souhaitent extraire rapidement des informations d'une vidéo sous forme de texte, par exemple pour des sous-titres, des transcriptions ou des résumés. L'application s'appuie sur des outils tiers comme **FFmpeg** et **Yt-dlp** pour la conversion de vidéo et l'extraction de l'audio. Ces outils, bien connus pour leur efficacité et leur flexibilité, permettent de traiter une large gamme de formats vidéo. Par ailleurs, l'application inclut un module de transcription utilisant des API comme **AssemblyAI**, permettant de transformer l'audio en texte avec une précision remarquable.

Objectifs du projet :

L'objectif principal de ce projet est de développer une application en Java capable de convertir des vidéos YouTube en fichiers audio MP3. Une fois l'audio extrait, l'application devra ensuite le transcrire en texte, facilitant ainsi l'accès à l'information sans avoir à regarder la vidéo. Ce processus automatisé vise à simplifier la gestion de contenu multimédia pour les utilisateurs. L'application doit être simple à utiliser, avec une interface graphique claire, permettant de charger une URL de vidéo, de choisir les options de conversion et de lancer l'extraction et la transcription. Le projet repose sur l'intégration de plusieurs technologies, telles que le traitement vidéo et audio, l'interfaçage avec des API externes et la gestion d'une interface utilisateur conviviale.

Technologies utilisées:

1. Eclipse:

Eclipse est un environnement de développement intégré (IDE) largement utilisé pour la programmation en Java. Il offre une suite complète d'outils pour le développement de logiciels, y compris un éditeur de code, des outils de débogage, un gestionnaire de versions, ainsi que des fonctionnalités d'auto-complétion et de refacturation de code. Dans ce projet, Eclipse est utilisé comme IDE principal pour coder et tester l'application, offrant un environnement stable et riche en fonctionnalités pour gérer les différentes technologies et garantir une exécution fluide du projet.

2. JavaFX:

JavaFX est un framework riche en fonctionnalités pour la création d'interfaces graphiques dans les applications Java. Il permet de concevoir des interfaces utilisateur modernes et interactives avec des éléments visuels comme des boutons, des fenêtres, des barres de progression et bien plus encore. Dans ce projet, JavaFX est utilisé pour construire une interface utilisateur fluide et intuitive, permettant aux utilisateurs de naviguer facilement à travers les différentes étapes de l'application, telles que le téléchargement de vidéos, la transcription de l'audio et l'affichage des résultats. Grâce à JavaFX, l'expérience utilisateur est optimisée et agréable.

3. Yt-dlp :

Yt-dlp est un outil en ligne de commande conçu pour télécharger des vidéos depuis des plateformes comme YouTube. Il est basé sur **youtube-dl**, mais avec des améliorations et des corrections de bugs. Dans ce projet, **yt-dlp** est utilisé pour récupérer des vidéos YouTube, ce qui permet d'en extraire l'audio pour une transcription ultérieure. Ce téléchargement est simplifié grâce à **yt-dlp**, qui gère divers formats vidéo et audio, ainsi que des métadonnées associées aux vidéos, garantissant une récupération rapide et efficace du contenu multimédia.

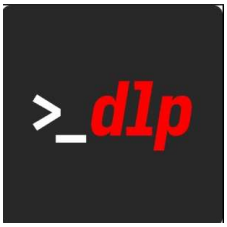

Figure 1: YT-DLP logo

4. Assembly AI

Assembly AI fournit un service de transcription audio automatique, permettant de convertir des fichiers audios en texte avec une grande précision. L'API utilise des modèles de reconnaissance vocale avancés pour analyser l'audio et générer une transcription fidèle du discours. Dans le cadre de ce projet, elle est utilisée pour convertir l'audio extrait des vidéos en texte, permettant ainsi aux utilisateurs d'obtenir une transcription précise et rapide de leurs fichiers multimédias. L'API **AssemblyAI** se distingue par sa fiabilité et sa capacité à traiter des fichiers audios de divers formats avec une grande efficacité.

Figure 2: Assembly AI logo

Diagrammes UML :

1. Diagramme de cas d'utilisation :

Le diagramme de cas d'utilisation présente les différentes interactions entre les utilisateurs et le système. Il permet de visualiser les fonctionnalités offertes par l'application et de comprendre comment les utilisateurs interagissent avec les diverses fonctionnalités du système. Ce diagramme aide à clarifier les attentes des utilisateurs et à définir les rôles et les actions possibles au sein de l'application

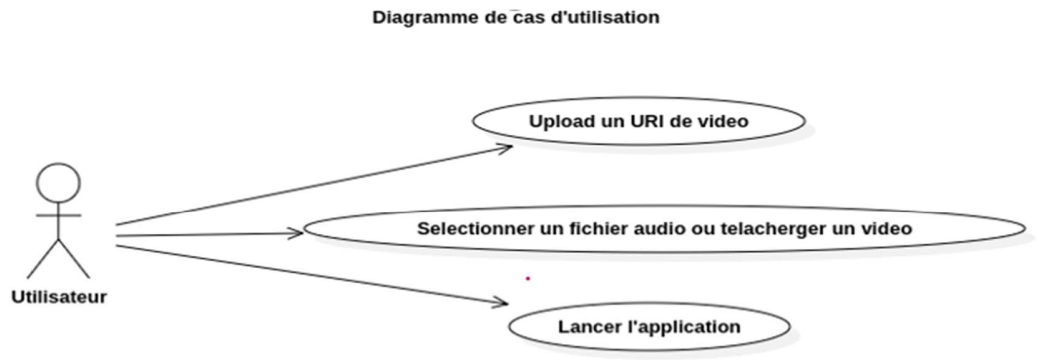

Figure 3 : Diagrame de cas d'utilisation

2. Diagramme de classe :

Le diagramme de classes illustre la structure du système en termes des ses classes, leurs attributs, leurs méthodes et leurs relations. Il permet de mieux comprendre l'architecture du code et la manière dont les différentes entités du système interagissent entre elles. Ce diagramme est essentiel pour la modélisation des données et la définition de la logique métier du système.

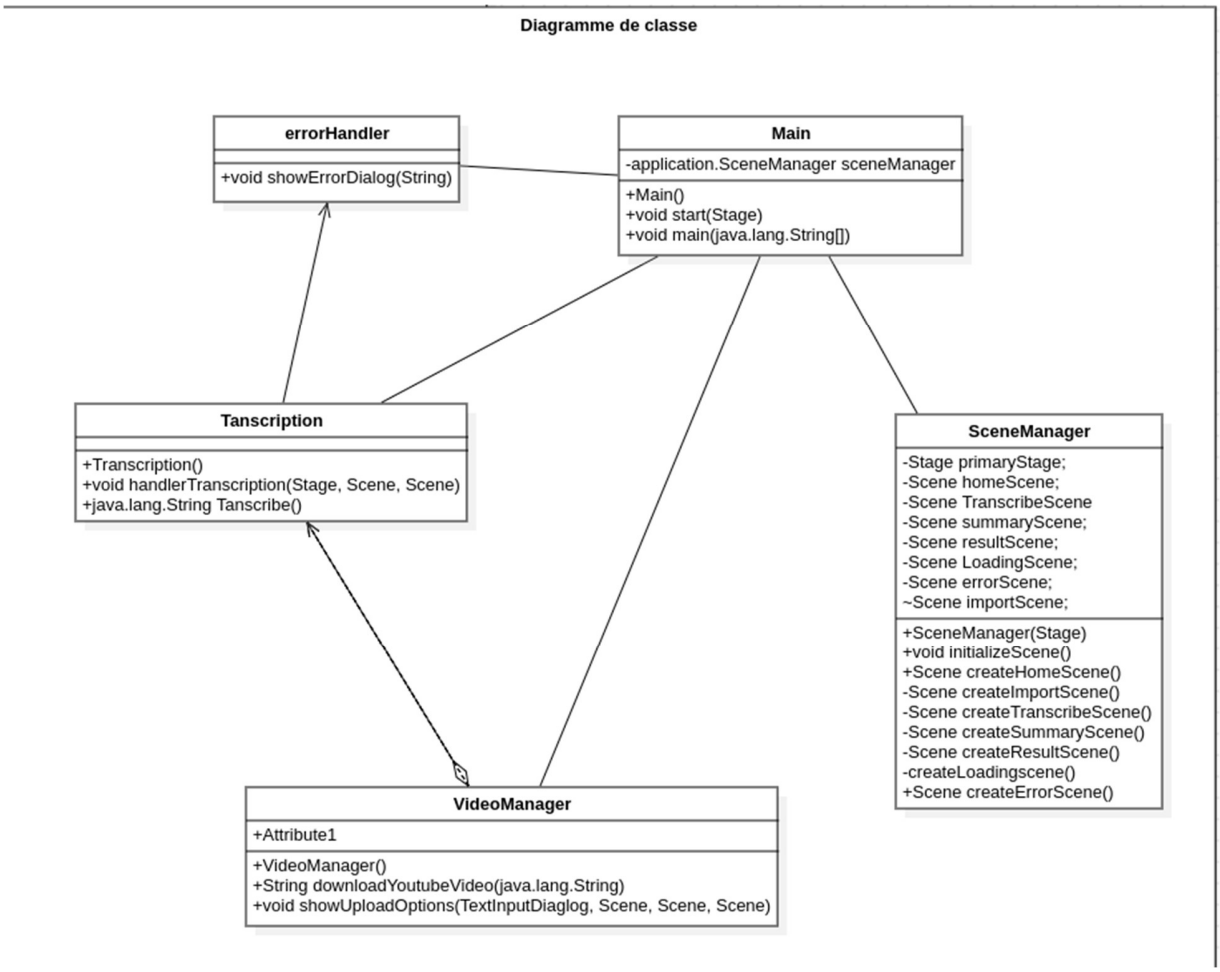

Figure 4 : Diagramme de classe

3. Diagramme d’activité :

Ce diagramme d'activité illustre le flux de processus de l'application, en détaillant les différentes étapes et décisions, allant de l'initialisation des scènes à la navigation entre les scènes (accueil, importation, transcription, résumé et résultat), en passant par la gestion des erreurs en cas d'échec d'importation de vidéo ou d'URL.

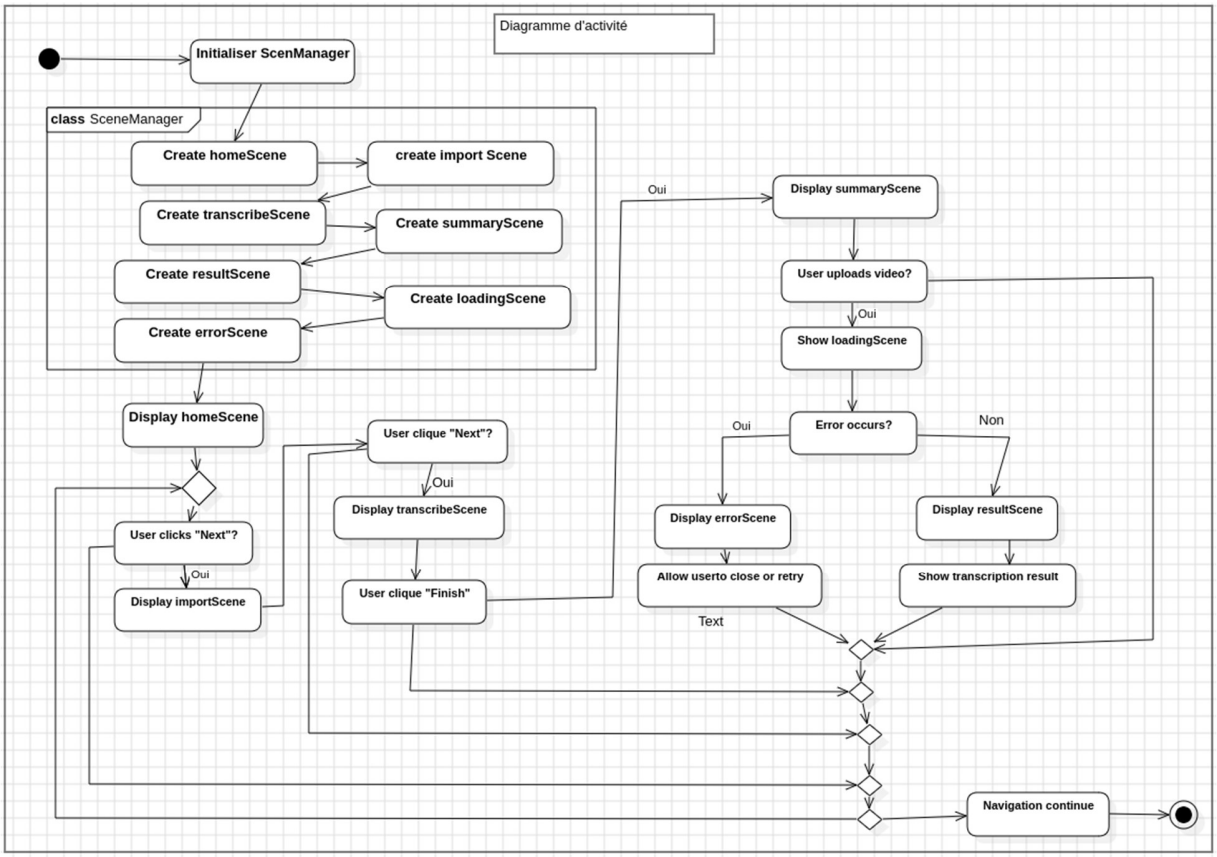

Figure 5: Diagramme d'activite

Explication des parties de code :

Ce code permet de télécharger une vidéo depuis YouTube, de l'extraire en format audio MP3, et de gérer le processus de téléchargement de manière automatique. Avant de télécharger la vidéo, il supprime tout fichier existant avec le même nom pour éviter les conflits. Ensuite, il utilise l'outil Yt-dlp pour récupérer l'audio de la vidéo et le convertir en MP3, en l'enregistrant dans un répertoire spécifié. Tout au long du processus, le code capture et affiche les logs de l'exécution pour informer l'utilisateur de l'avancement ou d'éventuelles erreurs, garantissant ainsi un suivi clair du téléchargement.

```java
public static String downloadYouTubeVideo(String videoUrl) throws IOException, InterruptedException {
        // Supprimer le fichier déjà existant
        try {
                // Chemin vers le fichier à supprimer
                File fileToDelete = new File("videos/maVideo.mp3");
                // Suppression forcée
                if (fileToDelete.exists()) {
                        FileUtils.forceDelete(fileToDelete);
                }
                System.out.println("Fichier supprimé avec succès !");
        } catch (IOException e) {
                System.out.println("Une erreur s'est produite lors de la suppression du fichier : " + e.getMessage());
                e.printStackTrace();
        }
        String outputDirectory = "videos/";
        String[] command = { "yt-dlp", "--extract-audio", "--audio-format", "mp3", "-o", outputDirectory + "maVideo",
                        videoUrl };

        ProcessBuilder processBuilder = new ProcessBuilder(command);
        processBuilder.redirectErrorStream(true);
        Process process = processBuilder.start();

        BufferedReader reader = new BufferedReader(new InputStreamReader(process.getInputStream()));
        String line;
        while ((line = reader.readLine()) != null) {
                System.out.println(line);
        }

        int exitCode = process.waitFor();
        if (exitCode == 0) {
                System.out.println("Téléchargement réussi !");

        } else {
                System.err.println("Erreur de téléchargement. Code : " + exitCode);
        }
        return null; // Si le téléchargement échoue; retourner null
}
```

Figure 6 : Fonction de telechatgement de YTB video

1. Téléchargement de vidéo YouTube

Une fois le fichier supprimé, le code prépare une commande yt-dlp (un outil de téléchargement de vidéos en ligne) pour télécharger et extraire l'audio en MP3 depuis une vidéo YouTube.

```
String outputDirectory = "videos/";
String[] command = { "yt-dlp", "--extract-audio", "--audio-format", "mp3", "-o", outputDirectory + "maVideo",
                videoUrl };
```

Figure 7 : Code d'utilisation de yt-dlp

2. Conversion de Vidéo en MP3

Avec la même ligne de code permet de convertir une vidéo en fichier audio MP3 en utilisant l'outil yt-dlp. La commande spécifiée inclut l'option --extract-audio, qui indique à yt-dlp d'extraire uniquement l'audio de la vidéo, et --audio-format mp3, qui définit le format de sortie en MP3. Le paramètre -o spécifie le répertoire de destination et le nom du fichier de sortie, ici "videos/maVideo", où sera enregistré l'audio extrait. Ainsi, ce code permet de récupérer l'audio d'une vidéo à partir de son URL et de le sauvegarder sous forme de fichier MP3 dans un répertoire désigné.

3. Transcription en Audio

La transcription consiste à convertir le contenu audio d'une vidéo en texte lisible. Dans ce cas, le processus commence par télécharger un fichier audio au format MP3 à partir d'une vidéo. Ensuite, l'API AssemblyAI est utilisée pour transcrire le discours présent dans l'audio en texte écrit. Des paramètres avancés, tels que la détection des intervenants et la langue de l'audio, sont utilisés pour affiner la transcription. Une fois le texte obtenu, il est affiché dans l'interface utilisateur de l'application et sauvegardé dans un fichier texte pour une consultation ultérieure. Ce processus de transcription permet de rendre l'audio accessible sous une forme textuelle, facilitant ainsi la recherche, la compréhension et l'archivage des informations.

```
// Transcribing the audio
Transcript transcript = client.transcripts().transcribe(new File(audioFilePath));
```

Figure 8 : Transcription de fichier mp3

```
// Handle transcription errors
if (transcript.getStatus().equals(TranscriptStatus.ERROR)) {
    throw new RuntimeException("Transcription failed: " + transcript.getError().get());
}
```

Figure 9 : Code de gestion des erreurs de transcription

```
String transcriptionResult = transcribe();

TextArea resultTextArea = (TextArea) ((VBox) ((BorderPane) resultScene.getRoot()).getCenter())
        .getChildren().get(1);
resultTextArea.setText(transcriptionResult);

// Navigate to the result scene
try {
                primaryStage.setScene(resultScene);
        } catch (Exception e) {
                e.printStackTrace();
        }
```

Figure 10 : Récupération du texte transcrit et sauvegarde dans un fichier texte

Une fois la transcription obtenue avec succès, le texte est extrait et sauvegardé dans un fichier texte, garantissant que l'utilisateur peut y accéder plus tard.

Interface utilisateur

1. Présentation générale

L'interface utilisateur de **Textify** a été conçue pour offrir une expérience fluide et simple. L'application se compose de plusieurs pages permettant de guider l'utilisateur à travers chaque étape du processus de conversion d'une vidéo YouTube en texte.

2. Étapes de l'interface utilisateur

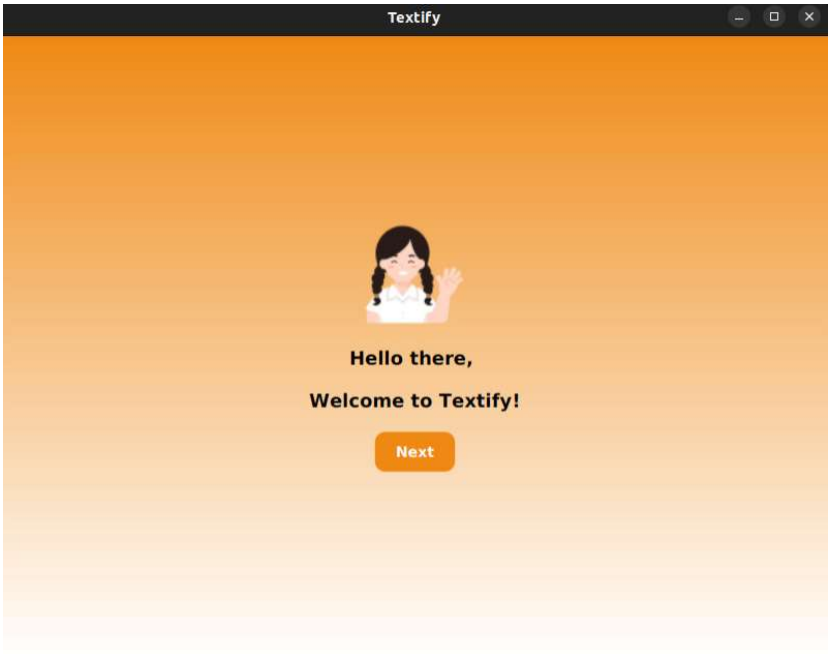

Figure 11 : Page d'accueil - Bienvenue sur TexTify

La première page affiche un message de bienvenue : *"Hello there, welcome to TexTify"*.
L'utilisateur peut cliquer sur le bouton Next pour démarrer le processus de conversi

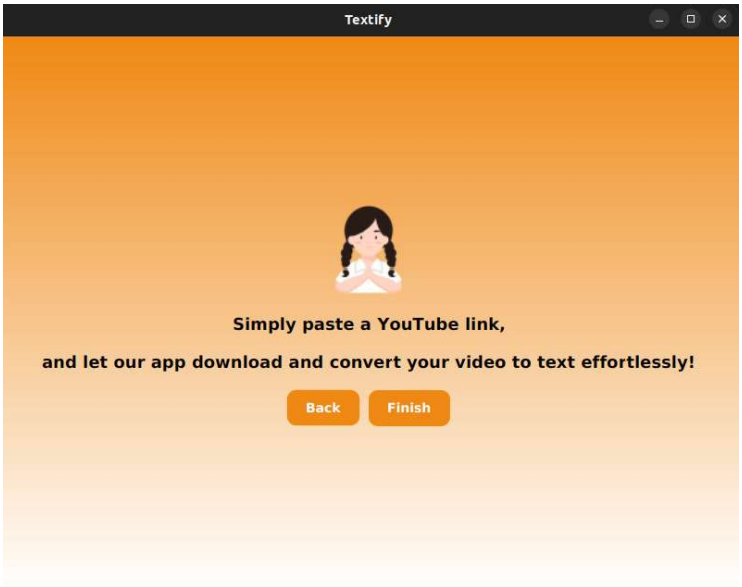

Figure 12 : Page suivante - Description de l'action

Cette page explique à l'utilisateur la fonction principale de l'application : *"Simply paste a YouTube link and let our application download and convert your video to text"*. L'utilisateur doit cliquer sur Next pour continuer vers l'étape suivante.

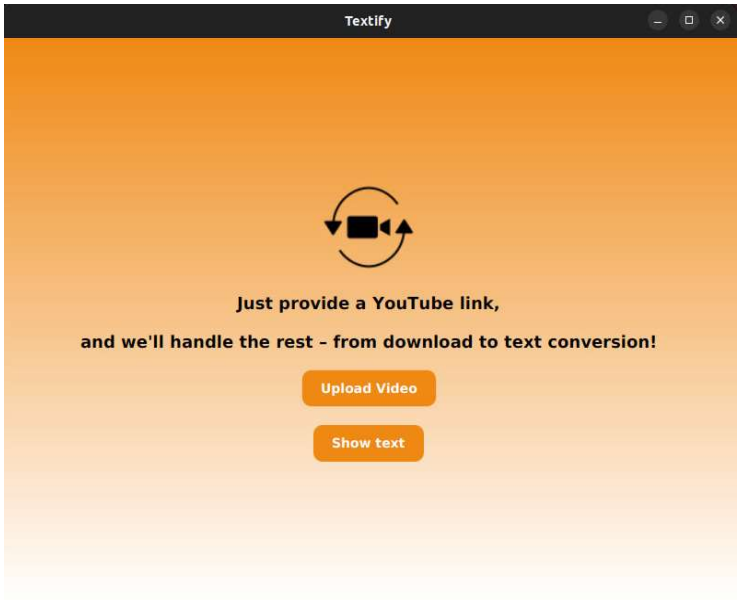

Figure 13 : Page de sélection de la vidéo - Coller le lien YouTube

Sur cette page, l’utilisateur doit coller le lien YouTube dans un champ de texte prévu à cet effet. Une fois le lien collé, l'utilisateur peut cliquer sur le bouton Show text pour lancer le téléchargement et la conversion de la vidéo en texte.

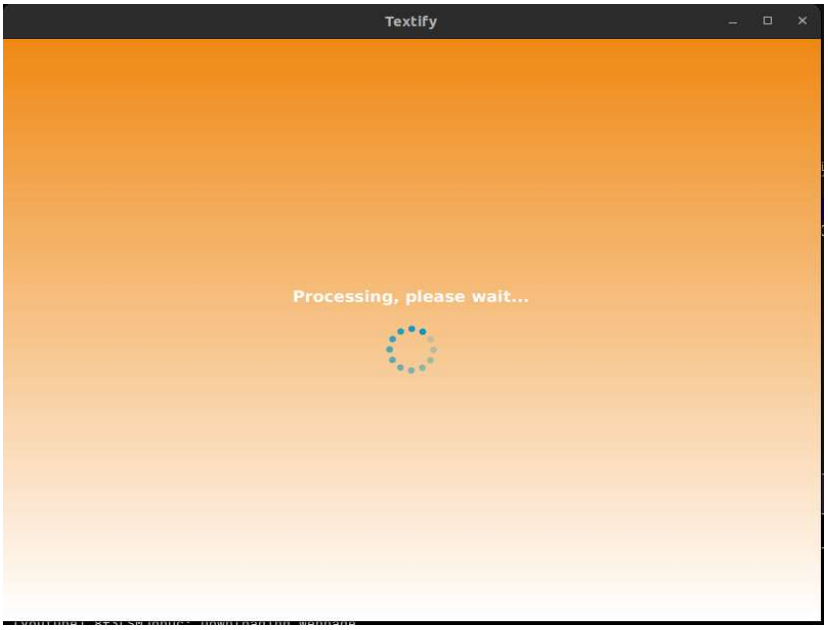

Figure 14 : Page de traitement - Processus en cour

Cette page affiche un message *"Processing, please wait"* pendant que l'application télécharge et traite la vidéo. Un indicateur de progression pourrait être ajouté pour montrer l’avancement du processus.

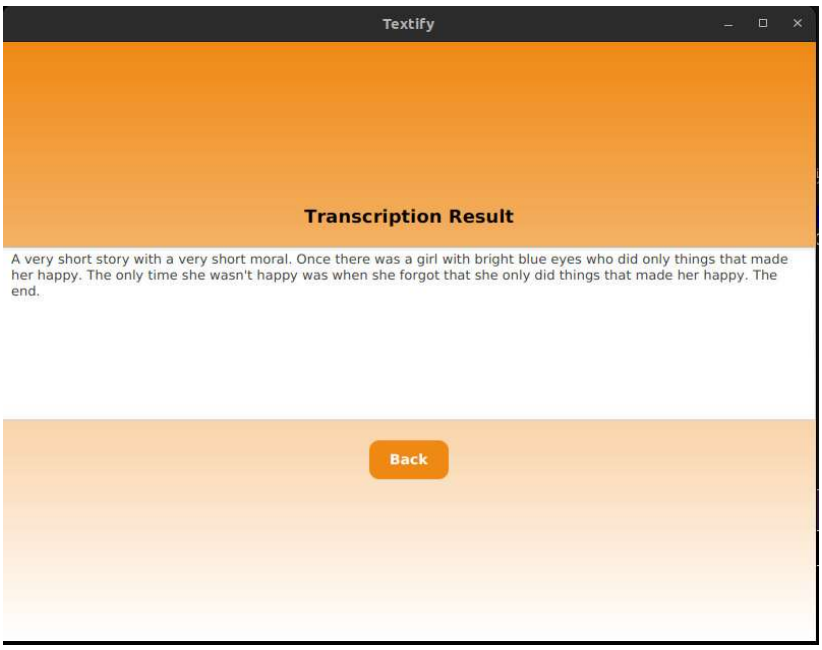

Figure 15: Page de résultats - Texte extrait

Une fois la vidéo convertie, cette page affiche le texte extrait. L'utilisateur peut copier ou exporter ce texte en fonction de ses besoins.

Design et ergonomie

- L'application est conçue de manière simple, avec des instructions claires et des boutons de navigation (Next et Back) pour que l'utilisateur puisse facilement avancer dans les étapes.
- L'interface est intuitive et minimaliste, mettant en avant les actions principales : coller un lien, télécharger la vidéo et afficher le texte obtenu , l'utilisateur a ainsi la possibilité de copier le text transcrit .

Conclusion

Le projet **"Textify"** nous a permis de développer une application innovante et fonctionnelle dédiée à la transcription automatique de contenu vidéo en texte. En intégrant des technologies avancées telles que **yt-dlp** pour le téléchargement de vidéos YouTube et l'**API AssemblyAI** pour la transcription audio, nous avons créé un outil répondant à un besoin croissant d'accessibilité et d'efficacité dans le traitement de l'information.

Tout au long de ce projet, nous avons acquis des compétences techniques précieuses en programmation et en développement d'applications. Nous avons également appris à travailler en équipe, à gérer des délais serrés et à surmonter divers défis techniques. La collaboration avec nos camarades de classe, combinée à l'encadrement attentif de notre professeur, a été un facteur clé de notre réussite.

En résumé, ce projet a été une expérience profondément enrichissante, nous permettant de mettre en pratique nos connaissances tout en développant un produit utile et pertinent. Nous envisageons de continuer à améliorer **"Textify"**, si l'opportunité se présente, afin de le rendre encore plus performant. Il est également important de souligner que ce projet a été une véritable occasion de travailler avec une équipe dynamique et a favorisé la naissance de nouvelles amitiés. Grâce à cette expérience, nous avons énormément appris, tant sur le plan technique que personnel.