System Requirements Specification

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements for **Textify**. The system will enable users to input text in a source language and get a translation in the desired target language. The system will also support features such as exporting translations and recognizing text from images(for example a book, a blackboard or a letter) via OCR (Optical Character Recognition).

1.2 Scope

This application will allow users to translate text in real-time, export translations into various data formats (both human-readable and machine-readable), and recognize text from images via OCR. The system will provide a user-friendly interface as a web application.

1.3 Definitions, Acronyms, and Abbreviations

- **API**: Application Programming Interface
- **NMT**: Neural Machine Translation
- **TTS**: Text-to-Speech
- **OCR**: Optical Character Recognition
- **Export**: The process of saving or sharing translation output in various file formats (e.g., PDF, Word, TXT).

1.4 Overview

This document describes the system requirements for **Textify**, including functional and non-functional requirements, and outlines features such as Exports, OCR-based image-to-text translation, TTS, and STT. It also describes the responsibilities of key project roles.

---

2. General System Requirements

2.1 Product Overview

- A machine translation application that provides text translations and supports exporting translations to various formats.
- The system will incorporate OCR functionality to extract and translate text from images.
- The system has the function to input text via STT,
- The System can convert text to audio files.
- The application will be a web application and can be used on almost any computer

---

2.2 Roles and Responsibilities

- **Customer**: Defines high-level requirements and provides feedback.
- **Manager**: Oversees project timelines, ensures proper execution of tasks, and provides weekly reports to the customer.
- **Designer**: Designs the UI/UX, focusing on usability and aesthetics.
- **Programmers**: Develop the application, integrate machine translation, and implement OCR, TTS, and STT functionality.
- **Testers**: Test the application, ensuring all features meet requirements and function as expected.

---

3. Functional Requirements

3.1 Text Translation

3.1.1 Requirement

- The system must be able to receive text in a source language and translate it into a target language.
- The system should support different language pairs, but the main focus is english

3.1.2 Acceptance Criteria

- The user enters text or an image with text in a field, selects the language to translate into.
- The system should return a translation.

3.3 Text-to-Speech (TTS)

3.3.1 Text Conversion to Speech

- **Requirement**: The system must be able to convert text (either original or translated) into speech.
- **Acceptance Criteria**:
    - The system accepts text input and provides an audio output that reads the text aloud.

3.3.2 Continuous Speech Playback

- **Requirement**: The system should support continuous TTS playback for long texts (e.g., paragraphs, documents).
- **Acceptance Criteria**:
    - The system should read aloud long texts continuously without interruptions.
    - The user can pause, stop, or skip sections of the speech.

3.3.3 Speech Rate and Volume Control

- **Requirement**: The system should allow the user to adjust the speech rate and volume.
- **Acceptance Criteria**:
    - Users should be able to control the speed (slow, normal, fast) and volume of the speech.

3.3.4 Audio Export Options

- **Requirement**: The system should allow users to export the generated speech as an audio file.
- **Acceptance Criteria**:
    - After generating speech from text, the system should offer an option to download the audio in standard formats.

3.4 Speech-to-Text (STT)

3.4.1 Speech Input Recognition

- **Requirement**: The system must accurately transcribe spoken language into written text.
- **Acceptance Criteria**:
    - The system accepts audio files and transcribes them into text.

3.4.2 Punctuation and Capitalization

- **Requirement**: The system must automatically insert punctuation marks and capitalization in the transcribed text.
- **Acceptance Criteria**:
    - The system should detect sentence boundaries (e.g., periods, commas) and capitalize the first word of each sentence.

3.4.3 Noise Handling

- **Requirement**: The system must be capable of accurately transcribing speech in moderate noisy environments.
- **Acceptance Criteria**:
    - The system should minimize transcription errors caused by background noise or low-quality audio input.

3.4.4 Audio Export Options

- **Requirement**: The system should allow users to export the transcribed text as a file (e.g., TXT, DOCX, PDF).
- **Acceptance Criteria**:
    - The system should offer an option to download the transcribed text in one or more common formats (TXT, DOCX, PDF).

3.4.5 Integration with Translation System

- **Requirement**: The system should allow the user to translate the transcribed text into another language.
- **Acceptance Criteria**:
    - Once the speech is transcribed into text, the system should provide the option to translate the text into a different language.

3.5 Optical Character Recognition (OCR)

3.5.1 Text Extraction from Images

- **Requirement**: The system must be able to extract text from images (e.g., scanned documents, photos with text).
- **Acceptance Criteria**:
    - The system must be able to recognize and extract text from a variety of image types (e.g., JPEG, PNG).
    - The extracted text must be displayed on the user interface.

3.5.2 Image Upload and Processing

- **Requirement**: The system must support image uploads for OCR processing.
- **Acceptance Criteria**:
    - Users can upload images (e.g., scanned documents or photos) via the web interface.

3.5.3 Accuracy of Text Recognition

- **Requirement**: The OCR engine must accurately recognize text from images, even in cases of non-standard fonts or noisy backgrounds.
- **Acceptance Criteria**:
    - The system should differentiate between images and text.

3.5.4 Support for Multilingual OCR

- **Requirement**: The system should support text recognition in multiple languages.
- **Acceptance Criteria**:
    - The OCR system should be able to extract text in different languages.

3.5.5 Image Preprocessing

- **Requirement**: The system should preprocess images (e.g., de-skewing, noise reduction) before extracting text.
- **Acceptance Criteria**:
    - The system should automatically preprocess images to improve the accuracy of text recognition.

3.5.6 Integration with Translation System

- **Requirement**: Once text has been extracted from an image, the user should be able to translate the text into another language.
- **Acceptance Criteria**:
    - After the OCR process is completed, the user can translate the extracted text into a supported language.

3.5.7 Export OCR Text

- **Requirement**: The system should allow users to export the extracted text into a file format (e.g., TXT, DOCX, PDF).
- **Acceptance Criteria**:
    - The system should allow users to download or export the recognized text in common file formats (TXT, DOCX, PDF).

3.6 User Interface

3.6.1 General Requirements

- **Requirement**: The system must provide a user-friendly interface that allows users to easily input text, select source and target languages, and view the translation.
- **Acceptance Criteria**:
    - The user interface is responsive and can be called via a web-browser

3.6.2 User Interface Layout and Design

- **Requirement**: The user interface must provide a clear, easy-to-navigate layout that guides users through the various functionalities (e.g., text translation, OCR, TTS, and STT).
- **Acceptance Criteria**:
    - The UI should have a clean and intuitive structure.
    - The main features (text translation, OCR, TTS, STT) should be easily accessible and clearly differentiated.
    - The design should work consistently across both desktop and mobile devices to ensure a smooth user experience.

3.6.3 Text Input and Language Selection

- **Requirement**: Users must be able to input text and select both the source and target languages for translation.
- **Acceptance Criteria**:
    - Dropdown menus or selection boxes should allow users to select source and target languages.
    - Visual cues (e.g., flags or language abbreviations) should clearly indicate the languages.

3.6.4 Responsiveness and Mobile-Friendliness

- **Requirement**: The user interface must be optimized for various screen sizes and resolutions, including desktop, tablet, and mobile devices.
- **Acceptance Criteria**:
    - The layout should dynamically adjust to different screen sizes and resolutions.
    - All key features (text translation, OCR, TTS, STT, export) should work seamlessly on mobile devices as well as desktops.
    - There should be no horizontal scrollbars on mobile devices, and users should be able to navigate the entire interface on smaller screens.

---

4. Non-Functional Requirements

4.1 Scalability

4.1.1 Requirement

- The system must be scalable to handle a growing number of requests and users, particularly during peak demand periods.

4.2 Security

4.2.1 Requirement

- All data transmissions must be encrypted using SSL/TLS.
- User data should not be stored unencrypted.

4.3 Privacy

4.3.1 Requirement

- User data should not be used for model training unless explicit consent is obtained from the user.

4.4 Maintainability

4.4.1 Modular Architecture

- **Requirement**: The system must be designed with a modular architecture to support easy updates, bug fixes, and future feature additions.
- **Acceptance Criteria**:
    - The system's architecture should be divided into separate, independent modules that can be modified or replaced without affecting the entire system.
    - Each module should have clear interfaces and minimal dependencies on other modules, making it easy to test and update individual components.

4.4.2 Code Documentation

- **Requirement**: The system's source code must be documented to ensure that future developers can understand, maintain, and extend the codebase.
- **Acceptance Criteria**:
    - All functions, classes, and methods should be documented with descriptions of their purpose, inputs, outputs, and any side effects.
    - Complex or critical sections of the code should have additional inline comments to explain their logic and rationale.
    - Documentation should include guidelines for code style, testing, and contributing to ensure consistency among developers.

4.4.3 Automated Testing

- **Requirement**: The system should have a comprehensive suite of automated tests to verify the correctness of the system's functionality and ensure that changes do not introduce new bugs.
- **Acceptance Criteria**:
    - Unit tests should be written for core system functions (e.g., translation, OCR processing, file exports) to ensure that each function works as intended.

4.4.4 Error Logging

- **Requirement**: The system must have a robust error logging mechanism to detect issues in real-time and support troubleshooting and maintenance activities.
- **Acceptance Criteria**:
    - The system should log errors, warnings, and critical system events with sufficient detail (e.g., error codes, timestamps, and stack traces).
    - Logs should be easily accessible and stored.

4.4.5 Version Control

- **Requirement**: The system must use a version control system (e.g., Git) to manage code changes, track revisions, and enable collaboration among developers.
- **Acceptance Criteria**:
    - All code changes should be tracked using a version control system.
    - The system must follow a branching strategy (e.g., feature branches, release branches) to ensure that development and production versions of the code are kept separate.
    - Clear commit messages should be used to explain changes, and pull requests should be reviewed by team members before merging.

4.4.6 Dependency Management

- **Requirement**: The system should have clear management of external dependencies.
- **Acceptance Criteria**:
    - The system's dependencies (e.g., libraries, frameworks) should be clearly listed and versioned in dependency management files (e.g., package.json, requirements.txt).

4.4.7 Easy Configuration and Deployment

- **Requirement**: The system must allow for easy configuration and deployment to new environments, including staging, testing, and production.
- **Acceptance Criteria**:
    - Configuration files (e.g., for environment variables, database connections) should be clearly separated from the codebase and easy to update without changing the source code.
    - Deployment scripts or tools (e.g., Docker, Kubernetes, CI/CD pipelines) should be provided to automate the deployment process and ensure consistent environments across stages.

4.7 Error Handling

4.7.1 User-Friendly Error Messages

- **Requirement**: The system must provide clear and user-friendly error messages to guide users when errors occur, ensuring they understand the issue and possible next steps.
- **Acceptance Criteria**:
    - The system should display error messages that explain the problem in simple terms, avoiding technical jargon.

4.7.2 Input Validation and Error Prevention

- **Requirement**: The system must validate user inputs to ensure that errors are prevented before they occur, particularly for text inputs, file uploads, and configuration settings.
- **Acceptance Criteria**:
    - The system should validate all user inputs (e.g., checking for unsupported file formats during image uploads, ensuring the entered text is not empty, or that the source and target languages are valid).
    - The system should proactively guide users to enter valid inputs by providing real-time feedback, such as format requirements (e.g., "Please upload an image in JPG or PNG format") or character limits (e.g., "The maximum text length is 5000 characters").
    - If the user provides invalid input, the system should immediately alert the user with a clear error message indicating the problem and offering corrective actions.

---

5. Use Cases

5.1 Use Case 1: Text Translation

5.1.1 Actors:

- **End User** (User)

5.1.2 Description:

A user inputs text into the system and receives a translation of the text into another language. The user can select both a source and a target language.

5.1.3 Preconditions:

- The user has successfully launched the system and is on the translation page.

5.1.4 Postconditions:

- The user receives the translation of the input text.

5.1.5 Main Flow:

1. The user enters the text to be translated in the input field.
2. The user selects the source and target language.
3. The system translates the text into the target language and displays the result.
4. The user can read the translated text or choose to export it.

5.1.6 Alternative Flows:

- **A1**: If the system cannot provide a translation, it displays an error message (e.g., "Translation could not be generated").

5.2 Use Case 2: OCR (Optical Character Recognition) for Image-to-Text

5.2.1 Actors:

- **End User** (User)

5.2.2 Description:

A user uploads an image containing text. The system uses OCR to extract the text from the image.

5.2.3 Preconditions:

- The user has successfully launched the system and is on the OCR page.
- The user has uploaded an image containing text (e.g., a scanned document or a photo with text).
- The user has selected the language of the scanned document.

5.2.4 Postconditions:

- The system extracts the text from the image and displays it to the user.

5.2.5 Main Flow:

1. The user uploads an image containing text (e.g., a photo or a scanned document).
2. The system uses OCR to extract the text from the image.
3. The extracted text is displayed to the user on the interface.

5.2.6 Alternative Flows:

- **A1**: If the OCR process fails to extract the text (e.g., due to poor image quality), the system displays an error message and prompts the user to upload a better-quality image.

5.4 Use Case 3: STT

5.4.1 Actors:

- **End User** (User)

5.4.2 Description:

The user uploads an audio file which is transcribed so that a text file containing the content of the audio file is output.

5.5 Use Case 4: TTS

5.5.1 Actors:

- **End User** (User)

5.5.2 Description:

The user copies or writes a text into an input field. The system synthesises the speech in an audio file. This can be played on the website and downloaded as a file.

5.5 Use Case 5: Export Translations

5.5.1 Actors:

- **End User** (User)

5.5.2 Description:

The user wants to export the translated text into a different format, such as PDF, Word, or plain text (TXT)

5.5.3 Preconditions:

- The user has entered the text for translation and received a translation.

5.5.4 Postconditions:

- The user can download the translation in the selected format (e.g., PDF, Word, TXT).

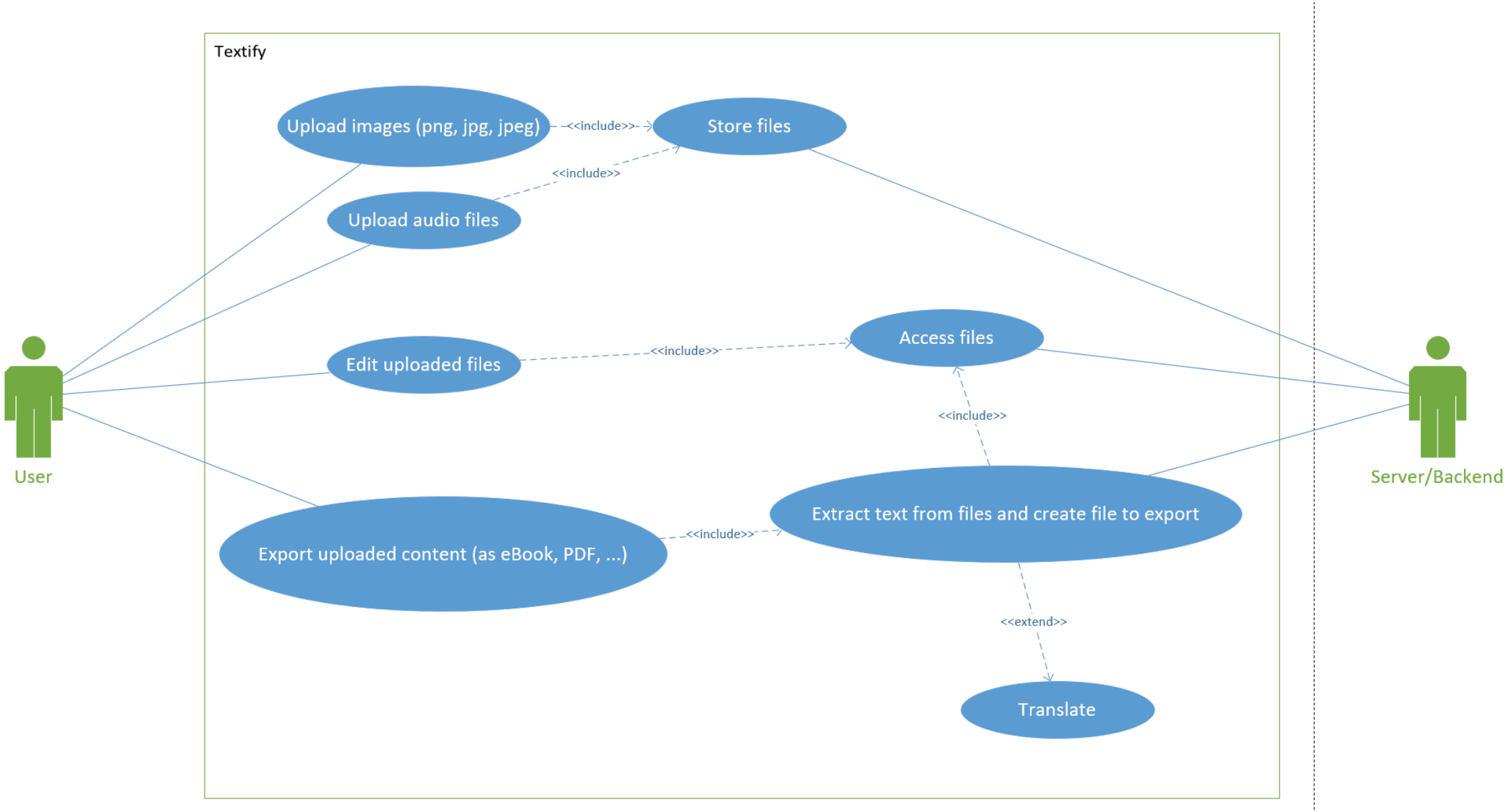
5.5.5 Main Flow:

1. The user selects the "Export" option.
2. The user selects the desired export format (e.g., PDF, Word, TXT).
3. The system starts the download for the user.

5.5.6 Alternative Flows:

- **A1**: If an error occurs during export (e.g., formatting issue), the system shows an error message.

5.6 Use Case Diagram



6. System Design and Architecture

6.1 System Architecture

- **Frontend**: Web user interface.
- **Backend**: Server hosting the AI model for translations and an API for communication between frontend and backend.
- **(Database**: Storage of user data and translation history.)

Sketch System Design: