Software Requirements Specification

for

Scribbler

Version 1.0

Prepared by - Yash Singhal, Samarthya Jha, Rishabh Budhia, Anay Rajguru

Student Technical Community-VIT

14-02-2021

Table of Contents

Table of Contents ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

# Introduction

## Purpose

The purpose of this document is to build a website which converts digital text from a Word or a text document to handwritten text.

## Document Conventions

This document uses the following conventions.

|  |  |
| --- | --- |
| pdf | Portable Document Format |
| docx | Word File |

## Intended Audience and Reading Suggestions

This project is intended for use by school and college students and professors and anyone else who wants their content in a handwritten format.

## Product Scope

The purpose of this project is to create convenient and easy to use application for generating handwritten text. The system is based on image processing which utilizes various fonts that are superimposed on a template depending on the data provided by the user.

## References

* <https://stackabuse.com/generating-pdf-files-in-node-js-with-pdfkit/>
* <https://pillow.readthedocs.io/en/stable/reference/Image.html>
* <https://stackoverflow.com/>
* <https://www.geeksforgeeks.org/>

# Overall Description

## Product Perspective

This project takes inspiration from already existing handwriting generating systems which failed to generate legitimate cursive handwriting. The project provides 3 different handwriting fonts for the user to choose from with each font being a custom one.

## Product Functions

The project follows a 3 step process which are required by the user :

* The user uploads a docx or a text file with suitable digital text written in it on the website.
* The user then selects the font type out of the 3 given font types.
* After clicking on convert the user gets the pdf of the handwritten text.

## User Classes and Characteristics (later)

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

## Operating Environment (later)

Operating environment for the system is as follows :

* Client/server system
* Operating system: Any
* Software requirements: Web Browser

## Design and Implementation Constraints

* The processing is done on a free tier instance on cloud platform so there will be some network latency in generating the handwritten text.
* The system can’t read formats such as pdf
* The system due to time and cost constrains can’t generate more than 4 pages of handwritten text.
* Due to lack of resources the system can generate handwritten text for only the English alphabet, digits and a few special symbols.

## User Documentation

The website home page includes a demo video which highlights the steps needed to be completed by user to get the desired result.

## Assumptions and Dependencies

The project uses third party APIs and packages which are assumed to work same in the future.

These packages include :

* pdfkit
* formidable
* jimp
* mammoth
* pillow

# External Interface Requirements

## User Interfaces

* Web Browser

## Hardware Interfaces

* Supported devices – PC , Tablet and Mobile Phones
* Internet Connection

## Software Interfaces

|  |  |
| --- | --- |
| **Particulars** | **Description** |
| Frontend | To implement the frontend we have used HTML5,CSS3, Bootstrap 5,Javascript, jQuery, and anime.js |
| Backend | To implement the backend we have used nodejs, express , axios, formidable, jimp , mammoth, and pdfkit. |
| Handwritten text generating API | To implement the API we have used python3, Pillow , and Flask. |
| Cloud platforms | Netlify, azure and Heroku. |

## Communications Interfaces

This project supports all types of web browsers.

# System Features

The system has only one main feature i.e to generate Handwriiten text in pdf format.

## Handwritten text generation

4.1.1 Description

The system takes .docx or a .txt file and returns a handwritten text in pdf format.

4.1.2 Stimulus/Response Sequences

* The user uploads the file
* Then user selects the font type
* The user clicks on the convert button
* The system generates and returns a pdf

4.1.3 Functional Requirements

For pdf generation:

REQ-1: The system requires a file in .docx or .txt format only to generate pdf.

REQ-2: File size shouldn’t exceed 25 KB.

REQ-3: The system should return the pdf generated in an acceptable time limit.

# Other Nonfunctional Requirements

## Performance Requirements

The fonts only have alphabets, digits and certain special symbols. The system will only have to generate 4 pages at max.

## Safety Requirements

The system should be able to recover from server crash and it should perform well in case of server overload.

## Security Requirements

None

## Software Quality Attributes

* Availability: The system should be available 24 hours.
* Correctness: The system should generate the exact copy of the digital text .
* Usability: The system should have a user friendly interface and it should allow multiple users to give request at the same time.

## Business Rules

The system should be free of cost and must be available for all type of users.

# Other Requirements

None