Software Requirements Specification

for

Scribbler

Version 1.0

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

Student Technical Community-VIT

15-02-2021

Table of Contents

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 Use Case Diagram 2

2.3 Operating Environment 2

2.4 Design and Implementation Constraints 3

2.5 User Documentation 3

2.6 Assumptions and Dependencies 3

3. External Interface Requirements 4

3.1 User Interfaces 4

3.2 Hardware Interfaces 4

3.3 Software Interfaces 4

4. Functional Requirements 4

4.1 System Feature 1 4

5. Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Software Quality Attributes 5

5.4 Business Rules 5

# Introduction

## Purpose

The purpose of this document is to present a detailed description of the website Scribbler. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate, and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system.

## 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 anyone who wants their content in a handwritten format. But the main target audience being college students who want to submit their assignments in handwritten format.

## Product Scope

This software system will be a website for anyone who wants to convert digital text to handwritten text. This system will be designed to create a 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.

## Use Case Diagram

**Use Case :** Generating handwritten text

**Diagram :**

## 

**Brief Description :** The user uploads a file with digital text and the system returns a pdf of handwritten text.

**Initial Step-By-Step Description**

Before this use case can be initiated, the User has already connected to the Scribbler site.

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

## Operating Environment

Any 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 will generate only 4 pages at max of handwritten text.
* Due to lack of resources only the first two fonts in the system can generate handwritten text for all the ASCII printable characters and the third font can generate handwritten text only for the English alphabet , digits and some characters i.e. -, @, +, (, ), semicolon, comma, ?, . , =, and %

## 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.
* While selecting font type please be careful if your text contains special characters other than . -, @, +, (, ), semicolon, comma, ?, . , =, and % since only the first two fonts are capable of generated handwritten text with all the ASCII printable characters.

## Assumptions and Dependencies

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

The package dependencies include :

* pdfkit
* formidable
* jimp
* mammoth
* pillow

# 

# External Interface Requirements

## User Interfaces

* Web Browser

## Hardware Interfaces

* Supported devices – PC , Tablet and Mobile Phones

## 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. |

# Functional Requirements

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

## Handwritten text generation

|  |  |
| --- | --- |
| **Use case Name** | Handwritten Text Generation |
| **Trigger** | The user clicks on the convert button after selecting the font type |
| **Precondition** | The Web is displayed with upload section where user has to upload a file in .docx or .txt format. |
| **Basic Path** | 1. The user clicks on the browse files and uploads a file in the upload section. 2. The user selects the font type. 3. The user clicks on the convert button. |
| **Alternative Path** | 1. The user drag n drop a file in the upload section. 2. The user selects the font type. 3. The user clicks on the convert button. |
| **Post condition** | The system returns the pdf generated in an acceptable time limit. |

# Nonfunctional Requirements

## Performance Requirements

## Safety Requirements

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

## 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.