Software Requirements Specification

for

Crowd Sourced Transcriptions

Version 1.0 approved

Prepared by: Richard Kanin, Louis Calbet,

Logan Nguyen, Nicholas Davis

CSC 380 – Team F

30th August, 2021

Table of Contents

Table of Contents ii

Revision History 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/ 2

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 3

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 3

2.6 User Documentation 3

2.7 Assumptions and Dependencies 4

3. External Interface Requirements 4

3.1 User Interfaces 4

3.2 Hardware Interfaces 4

3.3 Software Interfaces 4

3.4 Communications Interfaces 5

4. System Features 5

4.1 System Feature 1 5

4.2 System Feature 2 (and so on) 6

5. Other Nonfunctional Requirements 6

5.1 Performance Requirements 6

5.2 Safety Requirements 6

5.3 Security Requirements 6

5.4 Software Quality Attributes 6

5.5 Business Rules 6

6. Other Requirements 7

Appendix A: Glossary 7

Appendix B: Analysis Models 7

Appendix C: To Be Determined List 7

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Louis Calbet | 08/30/21 | Initial Revising | 1.0 |
| Nicholas Davis | 08/31/21 | 1.4 adding to the scope and rewording | 1.1 |
| Logan Nguyen | 09/03/21 | 1.4, 2.1, 2.3, 2.6 | 1.2.1 |
| Louis Calbet | 09/06/21 | Revising past work, working further | 1.2.2 |

# Introduction

## Purpose

A software that transcribes written documents via crowd sourcing.

~~<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>~~

## Document Conventions

*<Italic> --*  instructions on how to use the template

Non-italic – the solutions for the section

n/A – not applicable – section deemed irrelevant for the project – not yet an answer

TBD -- to be determined – don’t have enough info à not sure yet and need to go back on it

~~<Red Strikethrough>~~  ~~-~~ info that was there is candidate for decision

## Intended Audience and Reading Suggestions

* Professor of the CSC 380 course, Bastian Tenbergen
* Primary Stakeholder, Zachary Vickery
* Anyone who used the software and wants to check the IEEE design document
* TBD

~~<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>~~

## Product Scope

**1.4.1 What the product is about**

* Provide a web application to help libraries with the digital preservation of decaying documents through crowdsourcing transcription tasks.
* Compare and subsequently alter user entries to increase the reliability of the transcriptions
* Capacity to zoom in to what you are transcribing
* Implement a Geographical visualizer to show the “to: and from:” path the letters took.
* Make the transcribed content searchable
* Make the software engaging/interactive/user-friendly enough to maintain engagement
* Further specification *TBD*, awaiting response from Zachary Vickery

**1.4.2 What the product is not about**

* Not for translating between languages, only transcribing english.
* Will not directions for the path of the letters, only a straight line to illustrate distance.
* *TBD*

<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>

## References

<https://nyheritage.org/collections/millard-fillmore-papers>

<https://omeka.org/>

<https://www.virginiamemory.com/transcribe/>

<https://github.com/LibraryofVA/MakingHistory-transcribe-2.0>

<https://crowd.loc.gov/>

<https://crowd.loc.gov/about/>

Further references to be added at a later time

~~<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>~~

# Overall Description

## Product Perspective

This product is being designed as a replacement to existing competitor’s products, with the main additional benefit of incorporating a system to display the location of the origination and destination of the letters that are being transcribed. The software will accomplish this by creating an array of strings split at whitespace characters and comparing the results with other user’s inputs. The strings that match most will be considered correct, and users with high percentages of matches will be rewarded for such (reputation for contributions like Github stars or Stackoverflow reputations) (TBD). (Diagram to be incorporated at a later date)

~~<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>~~

## Product Functions

* Compare different user’s inputs, match them to create an agreeable output
* Show users the location of origination and destination of the letter they are transcribing
* Analyze validity of user inputs checked against each other and reward after a certain confidence level has been reached.
* Store the names of all participants, after all the documents have reached the desired confidence level, create a plaque with transcription participants’ names.
* Upon reaching said confidence level, the data stored will be trimmed to just the array of strings that was agreed upon most, and the array of participants.
* Further functions TBD

~~The major functions that this software is intended to perform is the comparison of different user’s inputs to create a transcription with the highest rate of agreeability among users possible. The software will be designed in a way that is engaging to its users and it will include a functionality to allow users to visualize the path the letter would have taken when it was originally sent.~~

~~<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>~~

## User Classes and Characteristics

* Transcribers
  + American Civil War Historians
  + Enthusiastic members of society committed to preserving history
  + Cursive writing experts
* Administrative Users
  + Archivists who are interested in the preservation of decaying letters
  + Librarians
* Database Accessors
  + Researches
  + Professors
  + Students

~~<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

This software will be hosted online for the public to access via any machine capable of connecting to the internet and running javascript.

Further specificity TBD

~~<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>~~

## Design and Implementation Constraints

Memory requirements are likely not a concern, as it will be hosted on the Oswego.edu servers. There is a 14 week time constraint on production of this software. The software will operate with integration onto an html platform hosting javascript. This software will implement a system for linking names of participants to the accounts that they use using a secure third party API.

~~<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>~~

## User Documentation

Included on the website will be a frequently asked questions page for common questions and issues and their answers and solutions. Additionally, we will include an email/or a message request form to contact the software developers to report new bugs and issues.

~~<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>~~

## Assumptions and Dependencies

This software is going to run in combination with multiple packages/libraries from other projects to smoothly operate and implement the product functions. We will likely be using Omeka and the Scripto plugin for transcription, implemented with javascript.

Further specification TBD

~~<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>~~

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

## Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## System Feature 1

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>