The document describes the needs of the customer. All required features are described in detail. Acceptance test cases are provided. Professional presentation.

Requirements Document

Section 1 - General Information

1.1: *Purpose*

The purpose of this document is to provide a detailed description of the Authoring App and the requirements it must fulfill. It will elaborate on those requirements by breaking them down into functional and non-functional. It will also explain the scope and features of this system, what it will do, under what conditions it will do it, the necessary adjacent interfaces, and other system requirements. This Requirements Document is intended for clients, client representatives, development managers and maintenance programmers.

1.2: *Scope*

This software product is named the Authoring App and will be used to simplify the process of lesson-making to teachers of visually-impaired students. The system is designed to maximize flexibility without compromising simplicity of usage. It will allow teachers to program a Treasure Box Braille (TBB) device to fabricate lessons without any knowledge in programming. Closing the gap between programming aspect and the lesson-making aspect, the teacher is able to perform their duties more efficiently.

More specifically, the system will allow the user to create lessons called ‘scenarios’ and specialize them however they wish to meet their individual purposes. The system will also be able to provide this function if the user themselves is visually-impaired. It will be able to integrate an external screen reader software that is responsible for delivering audio instructions to the user for navigation purposes.

The TBB device and screen reader are outside the scope of the Authoring App and will have to be provided by the users. More on this in section 3.1.

1.3: *Glossary*

Term	Definition
User/ Teacher	Referenced throughout the document, is the person who will be using this product to create educational scenarios
Authoring App/ product	The name of this product
TBB	Stands for Treasure Braille Box, it’s the hardware component of the larger system that the Authoring App is a part of
Braille	A form of written language for the visually-impaired
Screen reader	The external software component of the larger system that allows a visually-impaired user

	navigate this product
Scenario	
Scene	
Interaction	

1.4: *Overview of Document*

In the next section, *Overall Description*, describes the functionality of the product. It explains the foundation of the program and provides context for the next section, *Requirements Specification*.

The last section of this document elaborates on the environment that the product is expected to function within and the overall system attributes that can serve maintenance purposes. This section will also paint an image of the product interacting with users via acceptance test cases.

Section 2 - Overall Description

2.1: *Product Perspective*

The Authoring App is similar to programs such as LabVIEW, where the user can "program" a system using building blocks, or in this case, sequencing various options. The product is tested and presented using a simulator that imitates the TBB, which is required for actual usage by intended customers. The speech-to-audio functionality for optional commentary by the teacher or in case the teacher is visually-impaired, is also provided externally by a screen reader. Basically, the Authoring App is a component of a larger system where programming the TBB is the end goal and the screen reader is the medium through which the user and app communicate.

User -- [screen reader] → Authoring App -- [screen reader] → TBB

2.2: *User Characteristics*

The targeted user-base is expected to have teaching experience and be comfortable with making lessons plans from scratch. An intermediate level of computer literacy is required. If the user is visually-impaired, they need to be comfortable with using a screen reader. Lastly, the user is naturally expected to be fluent in braille.

2.3: *Functional Requirements*

- Must facilitate the creation flow of a scenario
    - Ask questions
    - Receive answer
    - Make comments
- Must record audio
- Must convert scenario to appropriate format
- Must be usable by visually-impaired users
    - Must support at least 2 screen readers for 2 platforms
- Must be able to access scenario later

2.4: *Non-Functional Requirements*

- Must allow a degree of flexibility with specializing scenarios
- Must provide up to 12 braille cells and buttons
- Must allocate storage for all saved scenarios

2.5: *Assumptions and Constraints*

The Authoring App was developed under the assumption that the final product is to be paired with the Treasure Box Braille. However, in this stage of development, the software has not been tested with the TBB but using a simulator.

Section 3 - Requirements Specification

3.1: *External Interfaces*

3.1.1: *Hardware Interfaces*

As mentioned previously, the Authoring App is intended to run on a Treasure Box Braille device. The TBB helps kids learn how to read braille by presenting letter or words to them, then they respond by pressing buttons.

3.1.2: *Software Interfaces*

There are numerous screen readers available. For a Linux computer, Orca is one software that is compatible. For Windows, there is NVDA and JAWS. Lastly, Mac OS come with a built in screen reader called Apple VoiceOver. This system is compatible with

3.2: *Acceptance Test Cases*

Use Case: Creating a New Scenario
Brief Description: user attempts to create a new scenario from the Main Menu
Step-by-Step Description:

1. The user selects ‘Create Scenario’ from Main Menu
2. The system brings up a new Scenario Setup window with 3 prompts
3. The user enters a name and indicates the number of braille cells and button they wish to have
4. The system redirects to the Scenario Builder, with the name the user entered at the top, and the number of cells and buttons they indicated in the respective drop-down menu

Use Case: Creating a New Scene
Brief Description: The user wishes to create a new scene within a scenario
Step-by-Step Description:

1. The user first saves current scene
2. The system saves the current scene in the List of Scenes panel and enables the ‘New Scene’ button
3. The user selects ‘New Scene’
4. The system resets Scenario Builder for the new scene

Use Case: Finish Scenario
Brief Description: The user is done with implementing scenes and wishes to save and finish the scenario
Step-by-Step Description:

1. The user selects ‘Finish Scenario (Save Scenario)’
2. The system saves and redirects back to the Main Menu

Use Case: Loading an Existing Scenario
Brief Description: The user wishes to load an existing scenario to use
Step-by-Step Description:

1. The user selects ‘Load Scenario’
2. The system brings up a browsing window

3. The user navigates to the specific scenario they wish to display and selects ‘Continue’
4. The system starts up the simulator with the first scene ready to play

3.3: *Software System Attributes*

3.3.1: *Reliability*

- This is an educational tool, minimum failures tolerated
- Backup options

3.3.2: *Portability*

- User can download and transport data (test scores)
- Adaptable with at least two operating systems’ file readers

3.3.3: *Maintainability*

- Swift updates to fix bugs
- Authoring App uses MVC layout
- Each window has its own Controller and View
- All date entered by user to their scenario is saved in the SceneAA.java and ScenarioAA.java classes

3.3.4: *Security*

- Limited security