Requirements Document

EECS 2311 Z – *Software Development Project*

Group 15: Aya Abu Allan, Gianluca Corvinelli, Mark Savin

Table of Contents

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 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**
Authoring App	The name of this product
Braille	A form of written language for the visually-impaired
Factory Scenarios Selection file	The location in which scenarios will be saved to and are found when loading or editing one
MVC	Stands for 'Model-View-Controller' and is a Microsoft-developed web application

PRISM Lab	A computer lab located at York University
Scenario	Essentially a lesson plan consisting of one or more scenes
Scene	A specific instance of an interaction between the teacher and the student. This interaction can consist of anything from simply asking a question all the way to creating a quiz with feedback
Screen reader	The external software component of the larger system that allows a visually-impaired user to navigate 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
User/ Teacher	Referenced throughout the document, is the person who will be using this product to create educational scenarios

1.4: *Overview of Document*

The next section, *Overall Description*, describes the functionality of the product. It explains the foundation of the program and provides context for the third section, *Requirements Specification*. That section will showcase the main functions of the *Authoring App* through Use Cases and describes the functional requirements. The fourth section will elaborate on the environment that the product is expected to function within and paint an image of the product interacting with users via User Acceptance Test cases. The last section of this document will briefly sum the overall system attributes that can serve maintenance purposes.

Section 2 - Overall Description

2.1: *Product Perspective*

The *Authoring App* allows the user to "program" a system using building blocks via 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 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.

2.2: *User Characteristics*

The targeted user-base is expected to have teaching experience and be comfortable with making lessons plans from scratch. A competent 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: *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 students 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 comes with a built in screen reader called Apple VoiceOver. This application is best compatible with NVDA.

3.2: *Use Cases*

UC1

Use Case Name: Creating a Scenario
Objective: The user wishes to create a new scenario and the system will redirect them.
Priority: High
Preconditions: The Authoring App is open and the user is at the Main Menu
Post conditions: A new scenario is saved to the Factory Scenarios Selection File
Flow of Events:

1. Basic Flow
   - 1.1. User selects ‘Create Scenario’ from the Main Menu
   - 1.2. User enters scenario name and the number of braille cells and buttons desired
   - 1.3. As user selects continue, they are redirected to the Scenario Builder
   - 1.4. User names scene
   - 1.5. User inputs a question under Option 1
   - 1.6. User inputs answer(s) under Option 2
   - 1.7. User saves scenario
   - 1.8. User selects ‘Finish Scenario’ and is redirected to the Main Menu
2. Alternative Flow 1: At step 1.6, user creates an interactive button under Option 3
   - 2.1. User records an Audio message to play if interactive button is initiated
   - 2.2. Return to step 1.7
3. Alternative Flow 2: At step 1.5, user inputs a message under Option 1
   - 3.1. User saves scene
   - 3.2. User selects ‘New Scene’

3.3. User names scene
3.4. Return to step 1.5-1.7
3.5. User selects the first scene in the List of Scene panel on the left
3.6. User Selects ‘Move Scene Down’
3.7. Return to step 1.8

UC2

Use Case Name: Loading a Scenario
Objective: The user wishes to load and deploy an existing scenario.
Priority: High
Preconditions: The Authoring App is open and the user is at the Main Menu. There is at least one existing scenario.
Post conditions: The scenario was deployed
Flow of Events:

1. Basic Flow
   1.1. User selects ‘Load Scenario’ from the Main Menu
   1.2. User browses Factory Scenarios Selection file for their desired scenario
   1.3. User selects ‘Open’
   1.4. The system deploys the Interactive Scenario Simulator

UC3

Use Case Name: Editing a Scenario
Objective: The user wishes to edit an existing scenario.
Priority: High
Preconditions: The Authoring App is open and the user is at the Main Menu. There is at least one existing scenario.
Post conditions: Changes made by the user are saved and the scenario is updated
Flow of Events:

1. Basic Flow
   1.1. User selects ‘Edit Scenario’ from the Main Menu
   1.2. User selects ‘Choose Scenario’
   1.3. User browses Factory Scenarios Selection file for their desired scenario
   1.4. User selects ‘Open’
   1.5. User selects ‘Edit Scenario’
   1.6. System redirects to the Scenario Editor
   1.7. User renames the first scene
   1.8. User saves scene
   1.9. User selects ‘Finish Editing Scenario’ and is redirected to the Main Menu

2. Alternative Flow 1: At step 1.7, the user rewrites the question under Option 1
    2.1. Return to step 1.8
3. Alternative Flow 2: At step 1.7, the user selects ‘Delete Currently Selected Scene’
    3.1. The user affirmatively answer the system’s prompt
    3.2. Return to step 1.9

UC4
Use Case Name: Accessing Sample Scenarios
Objective: The user wishes to consult the Sample Scenarios provided by the Authoring App.
Priority: Mid
Preconditions: The Authoring App is open and the user is at the Main Menu
Post conditions: The user has viewed a Sample Scenario
Flow of Events:

1. Basic Flow
    1.1. User selects ‘Sample Scenario’ from the Main Menu
    1.2. User selects one of the three provided Sample Scenarios
    1.3. The system deploys the Interactive Scenario Simulator
2. Alternative Flow: At step 1.2, user unchecks the ‘Are you visually capable?’ box
    2.1. Return to 1.2
    2.2. The system incorporates a screen reader

UC5
Use Case Name: Viewing the User Manual
Objective: The user wishes to view the User Manual.
Priority: Mid
Preconditions: The Authoring App is open and the user is at the Main Menu
Post conditions: The user has access to the User Manual
Flow of Events:

1. Basic Flow
    1.1. User selects ‘User Manual’ from the Main Menu
    1.2. The User Manual is available for the user

3.3: *Use-case-Derived Functional Requirements*

UC-1.1.2	The system shall redirect to Scenario Setup when ‘Create Scenario’ is selected The system shall provide a max of 12 braille cells and button

UC-1.1.3	The system shall only enable the 'Continue' button when the user names the scenario
UC-1.1.5	The system shall provide a field for the user to make comments or questions as Option 1
UC-1.1.6	The system shall provide a drop-down menu to select the braille the user wishes to employ
	The system shall provide a braille grid to manually customize what the currently selected braille will display
	The system shall provide a 'Clear Pins' button to clear the braille grid
UC-1.1.7	The system shall save a scene have it appear in the List of Scene column
UC-1.1.8	The system shall save the scenario to the Factory Scenarios Selection file
UC-1.2.0	The system shall provide two drop-down menus, one indicating the button number and the second dictating the type of interaction said button will provide. Interaction can be any of the following: No Interaction, Play Correct Audio Clip, Play Wrong Audio Clip, Repeat Scene, and Skip to Next Scene.
UC-1.2.1	The system shall provide the option to record an audio message in a comment box
UC-1.3.2	The system shall only enable 'New Scene' after the user saves the current scene
UC-1.3.6	The system shall provide the ability of reordering the scenes in the List of Scene panel once they have been saved
UC-2.1.2	The system shall have a Factory Scenarios Selection window appear for the user to browse
UC-2.1.4	The system shall deploy an Interactive Scenario Simulator once the user has selected an existing scenario file
UC-3.1.2	The system shall have a window prompting the user to 'Choose Scenario'
UC-3.1.5	The 'Edit Scenario' Button shall not be enabled till the user has selected one
UC-3.1.6	The system shall redirect the user to the Scenario Editor. This window shall operate identically to the Scenario Builder.
UC-3.3.0	The system shall provide the option of deleting the currently selected scene
UC-4.1.2	The system shall provide three Sample Scenarios that showcase different features
UC-4.2.0	The system shall be usable by visually-impaired users
UC-4.2.2	The system shall support at least 2 screen readers for 2 platforms
UC-5.1.2	The system shall provide access to the User Manual from the Main Menu

Section 4 – User Acceptance Testing

4.1: *Scope*

The scope of the User Acceptance Test Cases covers the main functionalities of the application, namely creating, loading, and editing scenarios. The scope of testing did not extend beyond the PRISM lab computers and Windows computers.

4.2: *Assumptions and Constraints*

Firstly, it is assumed all 3 users are experienced programmers, familiar with the program. Also, necessary hardware and software was provided by PRISM lab. Lastly, test cases and any issues accompanying them are well-documented in the project file.
Some constraints include the omission of TBB testing and loss of access to PRISM lab due to labour disturbances.

4.3: *Risks*

Description	Probability [high|mid|low]	Impact [high|mid|low]	Mitigation
Incomplete test environment due to constraints	med	med	Realistic time and resource planning
Improper error-handling by testers	low	high	Standardize bug reporting and update shared project file
Acceptance case testing failures	low	high	Complete all required features prior to testing

4.4: *Test Cases*

Use Case	Description	Expected Results	Pass/Fail
Creating a New Scenario: user attempts to create a new scenario from the Main Menu	1. Open the Authoring App. 2. Select 'Create Scenario' from Main Menu 3. The system brings up a new Scenario Setup window with 3 prompts. Enter a name and indicate the number of braille cells and button you wish to have	1. The Scenario Setup window appears when 'Create Scenario' is selected 2. 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	Pass Pass
Creating a New Scene: The user wishes to create a new scene within a scenario	1. Open the Authoring App. 2. Select 'Create Scenario' from Main Menu 3. The system brings up a new Scenario Setup window with 3 prompts. Enter a name and indicate the number of braille cells and button you wish to have 4. Save current scene 5. Select 'New Scene'	1. Once the user saves the current scene, the system saves the current scene in the List of Scenes panel on the left and enables the 'New Scene' button 2. After selecting 'New Scene', the system resets the Scenario Builder for a new scene	Pass Pass
Finish Scenario: The user is done with implementing scenes and wishes to save and finish the scenario	1. Open the Authoring App. 2. Select 'Create Scenario' from Main Menu 3. The system brings up a new Scenario Setup window with 3 prompts. Enter a name and indicate the	The system saves and redirects back to the Main Menu	Pass

	number of braille cells and button you wish to have 4. Select ‘Finish Scenario (Save Scenario)’		
Loading an Existing Scenario: The user wishes to load an existing scenario to use	1. Open the Authoring App. 2. Select ‘Load Scenario’ from Main Menu 3. Navigate to the specific scenario you wish to display and select ‘Continue’	1. Factory Scenarios Selection window appears when ‘Load Scenario’ is selected 2. System starts up the simulator with the first scene ready to play	Pass Pass
Editing a Scenario: The user wishes to rename a scene within a scenario	1. Open the Authoring App. 2. Select ‘Edit Scenario’ from Main Menu 3. A Scenario Editor window will pop up; select ‘Choose Scenario’ 4. Navigate the Factory Scenarios Selection window to the specific scenario you wish to edit and select ‘Continue’ 5. Rename the first scene and select ‘Save Scene’	1. A Scenario Editor window appears when ‘Edit Scenario’ is selected 2. Factory Scenarios Selection window appears when ‘Choose Scenario’ is selected 3. Scenario Editor window appears when ‘Continue’ is selected 4. The renamed scene should appear in the List of Scenes Panel on the left.	Pass Pass Pass Pass

Section 5 – Software System Attributes

5.1: *Reliability*

- This is an educational tool, minimum failures tolerated
- All data entered by user to their scenario is saved in the SceneAA.java and ScenarioAA.java classes

5.2: *Portability*

- Adaptable with at least two operating systems' file readers

5.3: *Maintainability*

- Swift updates to fix bugs
- Authoring App uses MVC layout
- Each window has its own Controller and View

5.4: *Security*

- Limited security