The ePortfolio Generator
Software Requirements Specification

Author: Richard McKenna
Debugging Enterprises™

Based on IEEE Std 830™-1998 (R2009) document format

1 Introduction

A well designed, informative and inviting personal Web site is an essential tool for any University student. A good personal site represents yourself online, providing a succinct summary of your academic career accented with personal style. It lets potential employers, grad schools, and other academics know a bit about your academic experiences, i.e. the courses you've taken and the projects you've worked on, while providing some details that let others know you are an interesting person who cares about the proper presentation of your work.

The process of making a good ePortfolio can, in part, be automated, which is the point of this project. We are to make an application that makes the generation of such a site a breeze for any student so that they do not have to be experienced in Web design or programming. Using our tool, a well-designed, colorful, informative site can be generated using content entered by a student user, where they may customize the site in all sorts of ways. The site could then be moved to a Web server to be shared with the world.

1.1 **Purpose**

The purpose of this document is to specify how our ***ePortfolio Generator*** application should work. The intended audience for this document is all the members of the development team, from the Web designers to the software engineers and even salespeople who will ultimately interface with potential University customers and advertisers. This document serves as an agreement among all parties and a reference for how the application should ultimately be constructed. Upon completing the reading of this document, one should clearly visualize how it will operate as well as what it ultimately will produce.

1.2 Scope

For Debugging Enterprises, the goal is to first make an application that Stony Brook University students of all majors might like to use and then to extend it to students at other schools. Therefore it is important to make good design decisions in this project that will result in reusable components that can accommodate all sorts of content as needed by many different types of students. As such, a framework (or set of frameworks), could be designed and constructed to be used to make ***The ePortfolio Generator*** a living piece of software that can easily accommodate change.

1.3 Definitions, acronyms, and abbreviations

Digication – A Web-based software product currently used by Stony Brook University for students to create ePortfolios.

ePortfolio – Short for Electronic Portfolio, it is typically a Web site that summarizes ones work, as in an academic setting.

Framework – In an object-oriented language, a collection of classes and interfaces that collectively provide a service for building applications or additional frameworks all with a common need.

GUI – Graphical User Interface, visual controls like buttons inside a window in a software application that collectively allow the user to operate the program.

IEEE – Institute of Electrical and Electronics Engineers, the "world's largest professional association for the advancement of technology".

UML – Unified Modeling Language, a standard set of document formats for designing software graphically.

Use Case Diagram – A UML document format that specifies how a user will interact with a system. Note that these diagrams do not include technical details. Instead, they are fed as input into the design stage (stage after this one) where the appropriate software designs are constructed based in part on the Use Cases specified in the SRS.

1.4 References

IEEE Std 830™-1998 (R2009) – IEEE Recommended Practice for Software Requirements Specification

Wikipedia – Provides adequate definitions for **ePortfolio** and **Digication**.

1.5 Overview

This SRS will clearly define how the ***ePortfolio Generator*** application should operate. Note that this is not a software design description (SDD), which would design how to construct the software using UML. This document does not specify how to build the appropriate technologies, it is simply an agreement concerning what to build. Section 2 of this document will provide the context for the project and specify all the conceptual design. Section 3 will present what user interface controls are needed and all program functionality. Section 4 provides a Table of Contents, an Index, and References.

2 Overall description

If you are a Stony Brook University student then you are likely already familiar with ***ePortfolios*** (see https://stonybrook.digication.com/ePortfolio-help/home). Digication provides a Web application that students may currently use to generate such a site. It is our intention to make a desktop application that allows one to make such an **ePortfolio** that one may then upload to ones own Web server.

2.1 Product perspective

This product will need to be easy to make customized content. Students from different majors will have very different course project content to show off, so we will have to accommodate those differences. This means students should be able to present their work using images and text as well as via slideshows and video content. In addition, students of different majors are likely to use very different styles and so our application must make it easy to select colors, fonts, and layouts that can make a site look more artistic or warm or cold or whatever.

2.1.1 System Interfaces

The ***ePortfolio Generator*** application will need provide a File Toolbar for managing things like the creation, loading, and saving of ePortfolios. The application's workspace should be switchable between a Page Editor and a Site Viewer. The Page Editor should be concerned with all the controls needed for entering and customizing content. The Viewer is for seeing how the site looks inside a Web browser. Note that we should make sure to integrate the viewer into our application's workspace rather than in a separate window. The user will want to constantly switch back and forth between the editor and viewer and each switch will trigger an update to the display if necessary. The Workspace Mode Toolbar will let the user do such switching. There should also be a Site Toolbar in the Page Editor Workspace for adding, removing, and selecting pages to edit. The UI regions for this application are specified below in **Figure 2.1**. Note that the ***ePortfolio Generator*** will ultimately exist as a standalone Java application and will generate Web content that should work on any Web server/browser combination.

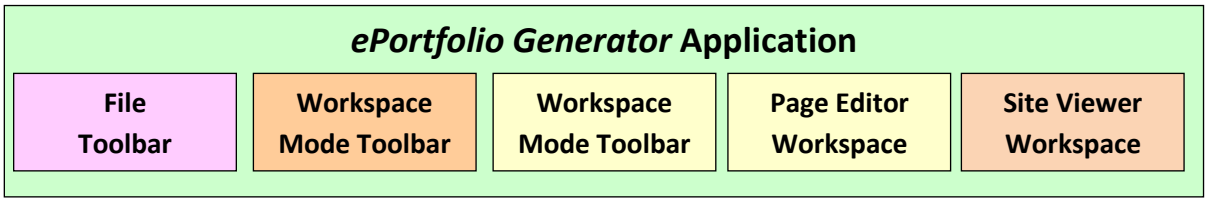

Figure 2.1: The broad screen regions for the *ePortfolio Generator*.

2.1.2 **User Interfaces**

Figure 2.2 below summarizes the ways with which the user will interact with our ***ePortfolio Generator*** application, which will be further detailed using UML Use Case diagrams. Note that "Clicks" always refers to a left-mouse-button click. These Use-Case diagrams should be fed as input directly into Section 3.1, external interfaces, which is where the design of the user interface is specified. Here is the full list of UML Use-Case Diagrams:

Figure 2.2: Overview of Use-Case Diagrams

Use Case	UI Region	Use Case
2.1	File Toolbar	New ePortfolio
2.2	File Toolbar	Load ePortfolio
2.3	File Toolbar	Save ePortfolio
2.4	File Toolbar	Save As ePortfolio
2.5	File Toolbar	Export ePortfolio
2.6	File Toolbar	Exit
2.7	Site Toolbar	Add Page
2.8	Site Toolbar	Remove Page
2.9	Site Toolbar	Select Page
2.10	Workspace Mode Toolbar	Select Page Editor Workspace
2.11	Workspace Mode Toolbar	Select Site Viewer Workspace
2.12	Page Editor Workspace	Select Layout Template
2.13	Page Editor Workspace	Select Color Template
2.14	Page Editor Workspace	Select Banner Image
2.15	Page Editor Workspace	Select Component
2.16	Page Editor Workspace	Choose Component Font
2.17	Page Editor Workspace	Update Page Title
2.18	Page Editor Workspace	Update Student Name
2.19	Page Editor Workspace	Update Footer
2.20	Page Editor Workspace	Add Text Component
2.21	Page Editor Workspace	Add Image Component
2.22	Page Editor Workspace	Add Slideshow Component
2.23	Page Editor Workspace	Add Video Component
2.24	Page Editor Workspace	Remove Component
2.25	Page Editor Workspace	Edit Text Component
2.26	Page Editor Workspace	Edit Image Component
2.27	Page Editor Workspace	Edit Slideshow Component
2.28	Page Editor Workspace	Edit Video Component
2.29	Page Editor Workspace	Add Text Hyperlink
2.30	Page Editor Workspace	Edit Text Hyperlink
2.31	Site Viewer Workspace	Navigate Site

Use Case 2.1: New ePortfolio

Use-Case:	New ePortfolio
Primary Actor:	Student
Goal in Context:	The user may at any point create a new ePortfolio to work on. Note that the application can only edit one ePortfolio at a time.
Preconditions:	The application has been started and initialized
Trigger:	The user clicks on the "New ePortfolio" button
Key Shortcut:	N/A
Scenario:	**1.** User starts the application, which loads app settings **2.** User clicks on the "New ePortfolio" button
Exceptions:	This button should always be enabled in the file toolbar. Note that should an ePortfolio already be loaded and unsaved, the application should prompt the user to save first.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Used very often, at least once for each ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.2: Load ePortfolio

Use-Case:	Load ePortfolio
Primary Actor:	Student
Goal in Context:	The user may at any point load an existing ePortfolio to work on. Note that the application can only edit one ePortfolio at a time.
Preconditions:	The application has been started and initialized
Trigger:	The user clicks on the "Load ePortfolio" button
Key Shortcut:	N/A
Scenario:	1. User starts the application, which loads app settings 2. User clicks on the "Load ePortfolio" button 3. User then selects the saved ePortfolio to load 4. ePortfolio then loaded for editing
Exceptions:	This button should always be enabled in the file toolbar. Note that should a ePortfolio already be loaded and unsaved, the application should prompt the user to save first.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Used very often, many times per ePortfolio.
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.3: Save ePortfolio

Use-Case:	Save ePortfolio
Primary Actor:	Student
Goal in Context:	The user may at any point save the current ePortfolio being worked on, as long as one is in progress. Note that the application can only edit one ePortfolio at a time.
Preconditions:	The application has been started and initialized and either a new ePortfolio has been started or an existing one has been loaded
Trigger:	The user clicks on the "Save ePortfolio" button
Key Shortcut:	N/A
Scenario:	1. User starts the application, which loads app settings 2. User either creates a new ePortfolio or loads an existing one 3. User edits various ePortfolio details 4. User clicks on "Save ePortfolio" button 5. If first time saving, user is prompted for ePortfolio file name 6. ePortfolio is saved to file using proper file name
Exceptions:	This button should always be provided in the file toolbar, but should only be enabled if the ePortfolio has been edited since the last save.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Used very often, many times per ePortfolio.
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.4: Save As ePortfolio

Use-Case:	Save As ePortfolio
Primary Actor:	Student
Goal in Context:	The user may at any point save the current ePortfolio being worked under a different file name, as long as one is in progress. Note that the application can only edit one ePortfolio at a time.
Preconditions:	The application has been started and initialized and an existing ePortfolio has been loaded
Trigger:	The user clicks on the "Save As ePortfolio" button
Key Shortcut:	N/A
Scenario:	1. User starts the application, which loads app settings 2. User loads an existing ePortfolio 3. User edits various ePortfolio details 4. User clicks on "Save As ePortfolio" button 5. User is prompted for ePortfolio file name 6. ePortfolio is saved to file using proper file name
Exceptions:	This button should always be provided in the file toolbar, but should only be enabled if an ePortfolio is currently being edited.
Priority:	Essential, must be implemented
When available:	Second Benchmark

Frequency of use:	Used at most once per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.5: Export ePortfolio

Use-Case:	Export ePortfolio
Primary Actor:	Student
Goal in Context:	Export current ePortfolio as a complete working Web site.
Preconditions:	The application has started and the user edited an ePortfolio. This would typically be done when a ePortfolio is completed, but could be done prior as well
Trigger:	The user clicks on the "Export ePortfolio" button.
Key Shortcut:	N/A
Scenario:	1. User is editing an ePortfolio, entering content, etc. 2. User clicks on "Export ePortfolio" button 3. Program generates and saves all pages, files, and directories needed for nicely formatted Web site to prescribed site directory. 4. Site is loaded into ePortfolio site viewer and viewer is made active in workspace
Exceptions:	This button should always be provided in the top toolbar on all screens, but should only be enabled if an ePortfolio is currently being edited.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Used at least once per ePortfolio.
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.6: Exit

Use-Case:	Exit
Primary Actor:	Student
Goal in Context:	The user wishes to close the application
Preconditions:	The user may wish to do so at any point, so the program must be prepared to handle the exit function at any time.
Trigger:	The user clicks on the "Exit" button.
Key Shortcut:	N/A
Scenario:	1. User is using application in any possible way 2. User clicks on "Exit" button 3. If an ePortfolio is unsaved, prompt user to save 4. Exit application
Exceptions:	N/A
Priority:	Essential, must be implemented

When available:	Second Benchmark
Frequency of use:	More than once per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.7: Add Page

Use-Case:	Add Page
Primary Actor:	User
Goal in Context:	While editing an ePortfolio, the user can add pages to their site, which is what this button is for.
Preconditions:	The app has started and the user is editing an ePortfolio
Trigger:	The user clicks on the "Add Page" button
Key Shortcut:	N/A
Scenario:	1. User is editing an ePortfolio 2. User clicks on the "Add Page" button 3. Application adds a New Page with a default title to the site and loads the Page Editor Workspace for that page so that the user may edit the page's details
Exceptions:	These Site Toolbar controls should only be enabled when ePortfolio editing is underway.
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Once per ePortfolio site page.
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.8: Remove Page

Use-Case:	Remove Page
Primary Actor:	Student
Goal in Context:	While editing an ePortfolio, the user can remove pages from their site, which is what this button is for.
Preconditions:	The app has started and the user is editing an ePortfolio
Trigger:	The user clicks on the "Remove Page" button
Key Shortcut:	N/A
Scenario:	1. User is editing an ePortfolio 2. User selects the page to edit, which loads it into the Page Editor Workspace 3. User clicks on the "Remove Page" button 4. Application Removes the page being edited and switches to another page in the site (if there is one)

Exceptions:	These Site Toolbar controls should only be enabled when ePortfolio editing is underway.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Multiple times per session
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.9: Select Page

Use-Case:	Select Page
Primary Actor:	Student
Goal in Context:	During editing, a user may wish to edit one of a number of different site pages. This provides a mechanism for selecting the page to edit.
Preconditions:	The application has been started and an ePortfolio is being edited with at least one Page.
Trigger:	The user clicks on a button named for the site page to select.
Key Shortcut:	N/A
Scenario:	1. User is editing an ePortfolio that has at least one page 2. User wishes to edit a different existing page and so clicks on the button titled the same as that page 3. Application loads the details for that page into the Page Editor Workspace
Exceptions:	If there are no pages, there are no buttons. If there is only one page, presumably that's the only page to edit, so it has no effect.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per ePortfolio.
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.10: Select Page Editor Workspace

Use-Case:	Select Page Editor Workspace
Primary Actor:	Student
Goal in Context:	While editing an ePortfolio the workspace can currently either be a page editor or site viewer. This Use Case is for when the user wishes to go to the Page Editor Workspace.
Preconditions:	ePortfolio editing is underway and the user is currently viewing the site viewer in the workspace
Trigger:	The user clicks on the "Page Editor Workspace" button
Key Shortcut:	N/A
Scenario:	**1.** User is viewing the site in the workspace **2.** User clicks on "Page Editor Workspace" button **3.** Application switches workspace to Page Editor, with most recently viewed page loaded for editing
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per session for every session
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.11: Select Site Viewer Workspace

Use-Case:	Select Site Viewer Workspace
Primary Actor:	Student
Goal in Context:	While editing an ePortfolio the workspace can currently either be a page editor or site viewer. This Use Case is for when the user wishes to go to the Site Viewer Workspace.
Preconditions:	ePortfolio editing is underway and the user is currently viewing the Page Editor in the workspace
Trigger:	The user clicks on the "Site Viewer Workspace" button
Key Shortcut:	N/A
Scenario:	1. User is editing a page in the workspace 2. User clicks on "Site Viewer Workspace" button 3. Application switches workspace to Site Viewer, with most recently edited page loaded for viewing
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per session for every session
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.12: Select Layout Template

Use-Case:	Select Layout Template
Primary Actor:	Student
Goal in Context:	Each site page should select a layout template for arranging the regions like navigation links, optional banner images, headers, footers, and content. Note that the application must provide at least 5 different layout templates. This Use Case is for when the user selects such a template for the page being edited.
Preconditions:	An ePortfolio is being edited and it has at least one page, which is being edited
Trigger:	The user clicks one of the 5 layout button options
Scenario:	**1.** User is editing an ePortfolio **2.** User is on the Page Editor workspace **3.** User clicks one of the 5 layout button options for the page **4.** Page is organized according to the selected layout
Exceptions:	Note that the application must be designed such that content can be easily changed from one layout to another without losing any content.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of buttons should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.13: Select Color Template

Use-Case:	Select Color Template
Primary Actor:	Student
Goal in Context:	The user can select from a choice of pre-made color schemes for the site page
Preconditions:	The user is editing a page
Trigger:	The user clicks one of the 5 layout button options
Scenario:	1. User is editing an ePortfolio 2. User is on the Page Editor workspace 3. User clicks one of the 5 color theme button options for the page 4. Page style is updated according to the selected color theme
Exceptions:	Note that the application must be designed such that content can be easily changed from one color theme to another without losing any content.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of buttons should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.14: Select Banner Image

Use-Case:	Select Banner Image
Primary Actor:	Student
Goal in Context:	Some layouts will allow for banner images, this is for selecting that image
Preconditions:	The user is editing a site page with a banner image
Trigger:	The user clicks the "Select Banner Image" button
Key Shortcut:	N/A
Scenario:	1. User is editing a page that uses a layout that has a banner image 2. User clicks on the "Select Banner Image" button 3. User selects image to use 4. User clicks ok once image is selected 5. User can then view how image looks in page
Exceptions:	Should the current layout for the page not have a banner image, this button should be deactivated
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per session for every game
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.15: Select Component

Use-Case:	Select Component
Primary Actor:	Student
Goal in Context:	The user will add content to a page through the use of different types of components, which can be selected and customized
Preconditions:	User is editing a page and selects and existing component in the page
Trigger:	The user clicks on a component
Key Shortcut:	N/A
Scenario:	1. User is editing a page 2. User clicks on a component to select it 3. Component is highlighted to denote that it is selected
Exceptions:	All components should be selectable
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.16: Choose Component Font

Use-Case:	Choose Component Font
Primary Actor:	Student
Goal in Context:	The user wishes to specify the font, meaning family, style, and size
Preconditions:	The user has selected a textual component
Trigger:	The user starts changing font settings for the selected textual component
Key Shortcut:	N/A
Scenario:	1. User is editing a page 2. User selects a textual component 3. User selects Google font family, font style, and font size from appropriate controls 4. User may then switch to site viewer to see results
Exceptions:	None, all text should allow for custom fonts
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.17: Update Page Title

Use-Case:	Update Page Title
Primary Actor:	Student
Goal in Context:	All sites should use textual navigation menus which will be located in different places depending on the layout used. Each page will therefore have a title in the menu. This is for changing the title for a particular page as it would appear in the navigation bar menu.
Preconditions:	The user editing a site page
Trigger:	The user changes the title text for the page
Scenario:	1. User is editing a page 2. User changes text in the title text editor for a page 3. Site should update navigation bar menu text for that page on all site pages
Exceptions:	Users can name their navigation links whatever they like.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	More than once per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.18: Update Student Name

Use-Case:	Update Student Name
Primary Actor:	Student
Goal in Context:	The name of the student should be prominently placed on each page of a student's ePortfolio. This is for updating the name.
Preconditions:	The user is editing an ePortfolio
Trigger:	User changes text in the student name text editor
Scenario:	1. User is editing an ePortfolio 2. User changes text inside student name text editor 3. Name I updated for all pages on site
Exceptions:	No ePortfolio should have multiple names, so one control should govern all pages of the site.
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	More than once per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.19: Update Footer

Use-Case:	Update Footer
Primary Actor:	Student
Goal in Context:	While editing an ePortfolio the user wishes to edit a page footer
Preconditions:	The site must have at least one page
Trigger:	The user clicks inside the Footer text editor and changes the content
Key Shortcut:	N/A
Scenario:	1. User is editing a site page 2. User clicks inside page footer text editor and changes content 3. Footer updates as text is entered
Exceptions:	All pages should have a footer with the location dependent upon layout
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per game ePortfolio.
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.20: Add Text Component

Use-Case:	Add Text Component
Primary Actor:	Student
Goal in Context:	The user has started editing an ePortfolio and now wishes to place a text component into a page. Note that a text component means either a paragraph, header, or list.
Preconditions:	The user is on a Page Editor screen for a page in the current ePortfolio site
Trigger:	The user clicks on the "Add Text Component" button
Key Shortcut:	N/A
Scenario:	1. User is on Page Editor workspace 2. User clicks on "Add Text Component" button 3. Dialog box opens allowing the user to select the type of text component (paragraph, header, or list) and enter content for the component 4. User enters desired settings and clicks OK to finalize choices and close dialog. 5. Page is updated
Exceptions:	If the ePortfolio doesn't have any pages, this button should be disabled
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.21 Add Image Component

Use-Case:	Add Image Component
Primary Actor:	Student
Goal in Context:	The user has started editing an ePortfolio and now wishes to place an image component into a page
Preconditions:	The user is on a Page Editor screen for a page in the current ePortfolio site
Trigger:	The user clicks on the "Add Image Component" button
Scenario:	A) User is on Page Editor workspace B) User clicks on "Add Image Component" button C) Dialog box opens allowing the user to select the image file to be shown as well as the video display size within the page. D) User enters desired settings and clicks OK to finalize choices and close dialog. E) Page is updated
Exceptions:	If the ePortfolio doesn't have any pages, this button should be disabled
Priority:	Essential, must be implemented

When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.22: Add Slideshow Component

Use-Case:	Add Slideshow Component
Primary Actor:	Student
Goal in Context:	The user has started editing an ePortfolio and now wishes to place a slideshow component into a page
Preconditions:	The user is on a Page Editor screen for a page in the current ePortfolio site
Trigger:	The user clicks on the "Add Slideshow Component" button
Scenario:	A) User is on Page Editor workspace B) User clicks on "Add Slideshow Component" button C) Dialog box opens allowing the user to edit the slideshow in all necessary ways. D) User enters desired settings and clicks OK to finalize choices and close dialog. E) Page is updated
Exceptions:	If the ePortfolio doesn't have any pages, this button should be disabled
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.23: Add Video Component

Use-Case:	Add Video Component
Primary Actor:	Student
Goal in Context:	The user has started editing an ePortfolio and now wishes to place a video component into a page
Preconditions:	The user is on a Page Editor screen for a page in the current ePortfolio site
Trigger:	The user clicks on the "Add Video Component" button
Scenario:	F) User is on Page Editor workspace G) User clicks on "Add Video Component" button H) Dialog box opens allowing the user to select the video file to be shown as well as the video display size within the page. I) User enters desired settings and clicks OK to finalize choices and close

	dialog. J) Page is updated
Exceptions:	If the ePortfolio doesn't have any pages, this button should be disabled
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.24: Remove Component

Use-Case:	Remove Component
Primary Actor:	Student
Goal in Context:	The user is editing an ePortfolio page and would like to remove one of the page components
Preconditions:	The user is editing an ePortfolio page and a component is selected
Trigger:	The user clicks on the "Remove Component" button
Scenario:	1. User is on ePortfolio Page Editor workspace 2. User selects a component 3. User clicks on the "Remove Component" button 4. Component is removed from page, which can then be seen if desired in page viewer.
Exceptions:	If the ePortfolio is over, this button should be disabled
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.25: Edit Text Component

Use-Case:	Edit Text Component
Primary Actor:	Student
Goal in Context:	The user has started editing an ePortfolio and now wishes to change the contents for a text component in a page
Preconditions:	The user is on a Page Editor screen for a page in the current ePortfolio site that has an existing text component
Trigger:	The user changes clicks on the "Edit" button
Scenario:	A) User is in the Page Editor Workspace B) User selects a textual component

	C) User clicks on Edit button D) Dialog box opens allowing the user to change the textual content within the component, including adding, editing, or removing paragraphs, lists, and headers. E) User enters desired settings and clicks OK to finalize choices and close dialog.
Exceptions:	If no textual component this button should be disabled
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.26: Edit Image Component

Use-Case:	Edit Image Component
Primary Actor:	Student
Goal in Context:	The user has started editing an ePortfolio and now wishes to change the settings for an image component in a page
Preconditions:	The user is on a Page Editor screen for a page in the current ePortfolio site and has selected an existing image component
Trigger:	The user clicks on the "Edit" button
Scenario:	A) User is in the Page Editor Workspace B) User selects an image component C) User clicks on Edit button D) Dialog box opens allowing the user to select the image file to be shown as well as the display size within the page. E) User enters desired settings and clicks OK to finalize choices and close dialog.
Exceptions:	If no image component this button should be disabled
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.27: Edit Slideshow Component

Use-Case:	Edit Slideshow Component
Primary Actor:	Student
Goal in Context:	The user has started editing an ePortfolio and now wishes to change the settings for

	a slideshow component in a page
Preconditions:	The user is on a Page Editor screen for a page in the current ePortfolio site and has selected an existing slideshow component
Trigger:	The user clicks on the "Edit" button
Scenario:	A) User is in the Page Editor Workspace B) User selects a slideshow component C) User clicks on Edit button D) Dialog box opens allowing the user to customize slideshow images, captions, slide order, width and height, etc. E) User enters desired settings and clicks OK to finalize choices and close dialog.
Exceptions:	If no slideshow component this button should be disabled
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.28: Edit Video Component

Use-Case:	Edit Video Component
Primary Actor:	Student
Goal in Context:	The user has started editing an ePortfolio and now wishes to change the settings for a video component in a page
Preconditions:	The user is on a Page Editor screen for a page in the current ePortfolio site and has selected an existing video component
Trigger:	The user clicks on the "Edit" button
Scenario:	A) User is in the Page Editor Workspace B) User selects a video component C) User clicks on Edit Video button D) Dialog box opens allowing the user to select the video file to be shown as well as the video display size within the page. E) User enters desired settings and clicks OK to finalize choices and close dialog.
Exceptions:	If no video component this button should be disabled
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of button should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.29: Add Text Hyperlink

Use-Case:	Add Text Hyperlink
Primary Actor:	Student
Goal in Context:	The user has created an ePortfolio and looking to add a hyperlink to
Preconditions:	The user is on the Page Editor workspace and a textual component exists on the page
Trigger:	The user clicks on the "Add Hyperlink" button
Scenario:	1. User is on ePortfolio Page Editor workspace 2. User selects text in text component 3. User clicks on "Add Hyperlink" button 4. User enters URL for hyperlink and clicks ok 5. User should see some feedback regarding hyperlink in text component and can switch to site viewer to in Web page
Exceptions:	If the ePortfolio doesn't have any textual copmonents
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of Site Viewer Workspace should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.30: Edit Text Hyperlink

Use-Case:	Edit Text Hyperlink
Primary Actor:	Student
Goal in Context:	The user has created an ePortfolio and would like to change or remove a hyperlink from a textual component
Preconditions:	The user is on the Page Editor workspace and a textual component exists on the page such that it has an existing hyperlink
Trigger:	The user clicks on a hyperlink
Scenario:	1. User is on ePortfolio Page Editor workspace 2. User selects text component 3. User clicks on hyperlink in textual component 4. User enters URL for hyperlink and clicks ok or requests deletion 5. User should see some feedback regarding hyperlink in text component and can switch to site viewer to in Web page
Exceptions:	If the ePortfolio doesn't have any textual copmonents
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of Site Viewer Workspace should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

Use Case 2.31: Navigate Site

Use-Case:	Navigate Site
Primary Actor:	Student
Goal in Context:	The user has created an ePortfolio and is looking at the generated site in the Web site viewer.
Preconditions:	The user is on the Site Viewing Workspace for an existing ePortfolio
Trigger:	The user clicks on links in the site
Scenario:	1. User is on ePortfolio Site Viewing Workspace 2. User clicks scrolls site page and clicks on hyperlinks 3. User views updated to reflect new page
Exceptions:	If the ePortfolio doesn't have any pages, it can't be navigated
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per ePortfolio
Open Issues:	Size, location, and style of Site Viewer Workspace should be finalized by UI designer. Should use an elegant, descriptive icon that suits application theme.

2.1.3 **Hardware Interfaces**

The target platform is the PC. This application requires a lot of textual input, which means we should target devices with keyboards like computers.

2.1.4 **Software Interfaces**

The ePortfolio Generator will be developed using the Java language. Note that since this may be our version of many, we should be careful in designing our classes to make sure they can accommodate future revisions. As part of this, it is important to manage the program data independent of its presentation. The application should use Java's JavaFX framework for building user interfaces as it is best suited to accommodating dynamic Web content.

2.1.5 **Communications Interfaces**

Note that in this product version the application will generate the ePortfolio site, but will not deploy it to a Web server or communicate with any other ePortfolio applications in any way. This is simply for building singular stand-alone sites that can be upload to existing Web servers via tools like SSH.

2.1.6 **Memory Constraints**

Since this application can be used for making any number of ePortfolio sites which all are likely to be content-rich, with images, slideshows, and videos, it is important that the application is smart in how it manages all this content.

2.1.7 **Operations**

It is important to note that making an ePortfolio is a fluid process by which the user may continually change their mind regarding what content to include. Therefore any action for adding content to a page must be undoable. For example, if one can add a textual component, one must be able to remove a textual component. The same goes for all other components.

2.1.8 **Site Adaptation Requirements**

N/A

2.2 **Product functions**

Note that the application will not allow for limitless page content types. Regarding textual components, they can only be paragraphs and lists. The application will not allow for tables, or all sorts of other HTML types.

2.3 **User characteristics**

Note that we are working on the assumption that university students would wish to use our application to make their sites, but in reality there may be many other types of users who might make good use of such an application. For that reason, we should try to limit references to “student” as much as we can within the actual application.

2.4 **Constraints**

Note that despite the fact that the site provides premade layouts and colors, users are still able to make bad decisions regarding the content they provide. The application will not do spell checking or grammar checking or make sure the user provides worthwhile content.

2.5 **Assumptions and dependencies**

It is assumed that all site pages will have navigation links and that page layouts will smartly locate and arrange all content for the site in well-organized regions. In addition, it is assumed that the preset color schemes will provide a good choice of color themes using colors that complement each other well and are themes Stony Brook students may wish to actually use.

2.6 Apportioning of the Requirements

Note that the benchmark scheduling is specified in the table but this is subject to change depending upon progress made along the way. Slippage in ones schedule is always an issue and so periodic review of schedule may help.

3 Specific requirements

The ePortfolio Generator user interface is to be carefully designed such that it accommodates all Use Cases specified in this document. The UI designer should aggregate all needed controls and then apportion them to their proper toolbars and other controls containers so as to make the application easy to use and attractive. Note that.

3.1 **External interfaces**

UI layouts and styles to be determined by UI designer.

3.2 **Functions**

One of the important things to consider in our application is providing the appropriate feedback to the user while the application is running. Users need feedback to enjoy their experience. This is typically done with visual cues like dialog boxes or highlighting (like when components are selected).

3.3 **Performance requirements**

N/A

3.4 **Logical database requirements**

N/A

3.5 **Design constraints**

JavaFX will be used because it effectively leverages each system's available rendering technologies and provides platform independence for personal computers. Therefore the assumption is that the user will have a mouse pointing device and keyboard. We will not assume any other devices.

3.6 **Software system attributes**

As professionals, all members of this project must take this project seriously. We are dedicated to producing robust software that exceeds the expectations of our customers. In order to achieve this level of quality, we should build a product with the following properties in mind:

3.6.1 Reliability – The program should be carefully planned, constructed and tested such that it behaves flawlessly for the end user. Bugs, including rendering problems, are unacceptable. In order to minimize these problems, all software will be carefully designed using UML diagrams and a Design to Test approach should be used for the Implementation Stage.

3.6.2 Availability –Customers may download and install the application application for free.

3.6.3 Security – All security mechanisms will be addressed by future revisions

3.6.4 Extensibility – It is important that more features can be added to the application, so file formats for should be carefully considered such that the game can be easily extended.

1.6.5 Portability – To start with, the app will target desktop Java applications. Future ports may be as a Web app.

3.6.6 Maintainability – Update mechanisms will be addressed by future revisions.

3.7 **Organizing the specific requirements**

Note that the application is simple enough that we need not worry about using an alternative arrangement of the content of this document. The specific requirements for this application already fit neatly into the sections listed in the IEEE's recommended SRS format.

3.8 Additional comments

It is important to keep in mind that the UI designers, map creators, and sound designers should make updates to the game themes and content as need to make something that looks great. It will be to their discretion to design all the interface controls in an effective, interactive style.

4 Supporting Information

Note that this document should serve as a reference for the designers and coders in the future stages of the development process, so we'll provide a table of contents to help quickly find important sections.

4.1 Table of contents

4.2 **Appendixes**

N/A