The *Course Site Generator* TM0
Software Requirements Specification

Author: Richard McKenna
Debugging Enterprises™

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

1 Introduction

University courses are required to provide course materials at the start of a term that help students understand course requirements and plan their semesters. These come in the form of a course syllabus and schedule that list information like course policies and dates for exams and assignments. Many times instructors find the most convenient means to disseminate this information is by way of a course Web site. Course Web sites are typically published at the start of a semester and are updated as it progresses. Such a site keeps the students up to date on deadlines and provides a place for instructors to distribute things like lecture slides.

Places like Stony Brook University's Computer Science Department have long required a Course Web Site for each taught course, and so every semester, the instructors teaching these courses generate and update this content. This process can be time consuming and tedious. In addition, many times instructors have more important things to do than build beautiful templates, and the result is each course Web site looks different, making it difficult for students to find the content they are after as they navigate differently arranged structures.

But why use course sites at all? Why not just use a tool like Blackboard for organizing course content? Well, sites that that have their own difficulties. They require time consuming login and navigation processes, they do many things, and so many things interfere with the quick retrieval for what a student is looking for, and they are general purpose sites, and so are cluttered with many things a course isn't even using. For an instructor, building and maintaining a custom course Web page still provides the best service to its students.

The ***Course Site Generator*** application intends to automate the process of building and updating a course Web site in one easy to use tool. The sites produced by this application will look good and will be customizable in a number of different ways, but will exist within a common site and page structure.

1.1 **Purpose**

The purpose of this document is to specify how our ***Course Site Generator*** program should look and operate. The intended audience for this document is all the members of the development team, from the instructors to the software engineers and designers. This document serves as an agreement among all parties and as a reference for how the site creation tool should ultimately be constructed. Upon completing the reading of this document, one should clearly visualize how the application will look and operate as well as understand the way a generated site is setup.

1.2 Scope

For this project the goal is for instructors to easily make and update course Web sites. There will be a common structure to the pages and so there are limitations on customization, but the site should be usable for instructors teaching courses in any department at any University.

1.3 Definitions, acronyms, and abbreviations

Document Object Model (DOM) – a tree data structure maintained by the browser that contains all content for the currently loaded Web page.

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.

HyperText Markup Language – a markup language used to describe Web pages. Web pages are text files encoded in HTML that can employ JavaScript and Stylesheets to build and style content.

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

JavaScript – the default scripting language of the Web, JavaScript is provided to pages in the form of text files with code that can be loaded and executed when a page loads so as to dynamically generate page content in the DOM.

Stylesheet – a static text file employed by HTML pages that can control the colors, fonts, layout and other style components in a Web page.

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.

1.4 References

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

1.5 Overview

This SRS will clearly define how the ***Course Site Generator*** application should look and 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 how the user interface should be laid out. Section 4 provides a Table of Contents, an Index, and References.

2 Overall description

Here you are a course instructor preparing content for your new semester and it's time to create and upload your course Web page. Perhaps you've taught this course before and so you have an old Web page you can modify, but still you spend quite a bit of time editing HTML such that everything is updated properly. Then, as you further make course decisions you realize you're going to need to add a lecture to your schedule, which requires quite a bit of HTML refactoring, again, it's time consuming and error prone. When it's all said and done you've spent a large amount of time authoring a site that doesn't look so great and will be difficult to keep updated without errors as the semester goes along. This is a job for our ***Course Site Generator***.

2.1 Product perspective

Our site generator will automate the process of creating our course Web site to greatly reduce the amount of time needed in planning a semester and update it as it progresses. In order to do that it will require the user provide a Web site template with some of the course content that is unlikely to change from semester to semester, and then a series of JSON files containing the data that will vary.

2.1.1 System Interfaces

The ***Course Site Generator*** will be a traditional workspace-type application that will let the user create a new course site and export it as needed. The first important thing to understand is what may go into a course site. Not all courses use the same content, or even the same pages for their sites, but some things, like the navigation bar structure, will be uniform. The following two course sites provide examples of what a course site might look like. Note that they have some notable differences regarding how many pages and the content of each page:

CSE 219

http://www3.cs.stonybrook.edu/~cse219/Section02/index.html

This site has a minimal Home (index.html) page but has dynamic content loaded on its Syllabus, Schedule, and HWs pages.

CSE 308

http://www3.cs.stonybrook.edu/~cse308/Section01/index.html

This site has a Home (index.html) page with a dynamically loading list of teams. There is also a projects page with a dynamically loading list of projects.

The particularly important things this application will do is provide a convenient means for editing data that will end up in JSON files loaded for these pages. It is important to understand the structure of the various JSON files that need to be build. In this respect there are five types of JSON files that will need to be generated, and so our application will need to provide a means to edit the content destined for these files, with examples listed below:

- http://www3.cs.stonybrook.edu/~cse219/Section02/js/TAsData.json
- http://www3.cs.stonybrook.edu/~cse219/Section02/js/RecitationsData.json
- http://www3.cs.stonybrook.edu/~cse219/Section02/js/ScheduleData.json
- http://www3.cs.stonybrook.edu/~cse308/Section01/js/TeamsAndStudents.json
- http://www3.cs.stonybrook.edu/~cse308/Section01/js/ProjectsData.json

It is important that we understand these JSON file formats and understand the data they hold. Note that JavaScript files already exist for properly loading this data into their necessary pages, one of these JavaScript loading files, TAsBuilder.js, will need some modification, as will the TAsData.json format, in order to accommodate differentiating between Graduate and Undergraduate teaching assistants such that they may be listed in their proper places.

2.1.2 **User Interfaces**

Our site editing program will be a desktop application and so will make use of a mouse and keyboard for user input. Figure 2.2 below summarizes the ways with which the user will interact with our ***Course Site Generator*** application, which will be further detailed using UML Use Case diagrams. 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:

Use Case	UI Context	Use Case
2.1	File Toolbar	Create New Course Site
2.2	File Toolbar	Load Course Site
2.3	File Toolbar	Save Course Site
2.4	File Toolbar	Save As Course Site
2.5	File Toolbar	Export Course Site
2.6	File Toolbar	Exit
2.7	Edit Toolbar	Undo
2.8	Edit Toolbar	Redo
2.9	Edit Toolbar	About
2.10	Course Details Pane	Edit Course Info
2.11	Course Details Pane	Select Export Directory
2.12	Course Details Pane	Select Template Directory
2.13	Course Details Pane	Toggle Use Template Page
2.14	Course Details Pane	Select Branding Images
2.15	Course Details Pane	Select Style Sheet
2.16	TA Data Pane	Add Teaching Assistant
2.17	TA Data Pane	Edit Teaching Assistant

2.18	TA Data Pane	Remove Teaching Assistant
2.19	TA Data Pane	Toggle TA Undergrad
2.20	TA Data Pane	Toggle TA Office Hours
2.21	TA Data Pane	Change Start/End Office Hours
2.22	Recitations Pane	Add Recitation
2.23	Recitations Pane	Edit Recitation
2.24	Recitations Pane	Remove Recitation
2.25	Schedule Pane	Edit start and end dates
2.26	Schedule Pane	Add Schedule Item
2.27	Schedule Pane	Edit Schedule Item
2.28	Schedule Pane	Remove Schedule Item
2.29	Projects Pane	Add Team
2.30	Projects Pane	Edit Team
2.31	Projects Pane	Remove Team
2.32	Projects Pane	Add Student
2.33	Projects Pane	Edit Student
2.34	Projects Pane	Remove Student

Figure 2.1: Overview of Use-Case Diagrams

Use Case 2.1: Create New Course Site

Use-Case:	Create New Course Site
Primary Actor:	Instructor
Goal in Context:	The user wishes to make a new course Web site
Preconditions:	The application has been started and the user is viewing the user interface
Trigger:	The user clicks on the New button
Key Shortcut:	N/A
Scenario:	• User is viewing the Course Site Generator • User clicks on the New button • User is prompted to save first if work is unsaved • Data is initialized and the user interface is reset with default values for a new site
Exceptions:	This button should always be enabled.
Priority:	Essential, must be implemented
When available:	First Benchmark
Frequency of use:	Used every time the user makes a new site
Open Issues:	Size, location, and style of button should be finalized by UI designer

Use Case 2.2: Load Course Site

Use-Case:	Load Course Site
Primary Actor:	Instructor
Goal in Context:	The user wishes to load and edit a course Web site that has already been created
Preconditions:	The application has been started and the user is viewing the user interface
Trigger:	The user clicks on the Load button
Key Shortcut:	N/A
Scenario:	• User is viewing the Course Site Generator • User clicks on the Load button • User is presented contents of work directory and selects file to load • User presses Ok and program proceeds to load site and present it for editing
Exceptions:	Should the user select a file that is not loadable because it is in the incorrect format, the program should not crash, but instead should display a dialog message telling the user what happened
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Used every time the user edits and existing site
Open Issues:	Size, location, and style of button should be finalized by UI designer

Use Case 2.3: Save Course Site

Use-Case:	Save Course Site
Primary Actor:	Instructor
Goal in Context:	The user wishes to save the course Web site that is currently loaded and has been edited
Preconditions:	The user is currently editing an existing site and has made at least one change

Trigger:	The user clicks on the Save button
Key Shortcut:	N/A
Scenario:	• Site is saved to its current file (note that when a site is created for the first time the file should be created at that time)
Exceptions:	This button should only be enabled if the user has made a change since last loading or saving
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Used many times per editing session
Open Issues:	Size, location, and style of buttons should be finalized by UI designer

Use Case 2.4: Save As Course Site

Use-Case:	Save As Course Site
Primary Actor:	Instructor
Goal in Context:	The user wishes to save the current course site under another file
Preconditions:	The user is currently editing an existing site
Trigger:	The user clicks on the Save As button
Key Shortcut:	N/A
Scenario:	• Site is saved to its current file (note that when a site is created for the first time the file should be created at that time)
Exceptions:	This button should only be enabled if the user has made a change since last loading or saving
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Used many times per editing session
Open Issues:	Size, location, and style of buttons should be finalized by UI designer

Use Case 2.5: Export Course Site

Use-Case:	Export Course Web Site
Primary Actor:	Instructor
Goal in Context:	User wishes to export the site
Preconditions:	The user has already specified the Export directory
Trigger:	The user clicks on the Export button
Key Shortcut:	N/A
Scenario:	• User clicks on Export button, which should export full site to export directory
Exceptions:	If an export directory has not yet been specified this button should be disabled
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Used every time site generation and JSON file writing is to be verified
Open Issues:	Size, location, and style of stats display should be finalized by UI designer

Use Case 2.6: Exit Application

Use-Case:	Exit Application
Primary Actor:	Instructor
Goal in Context:	The user is done working and wishes to exit the program
Preconditions:	N/A
Trigger:	The user clicks on either the Exit button or the window close button
Key Shortcut:	N/A
Scenario:	• If the user has unsaved work the application will prompt the user to save, so the user will specify whether to save or not. Then the application will close.
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	First Benchmark
Frequency of use:	Used once per session
Open Issues:	Size, location, and style of stats display should be finalized by UI designer

Use Case 2.7: Undo

Use-Case:	Undo
Primary Actor:	Instructor
Goal in Context:	Lets the user Undo an action
Preconditions:	User is editing a site and has made at least one change
Trigger:	User clicks Undo button
Key Shortcut:	Ctrl-Z
Scenario:	• User edits site data • User presses Undo
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of button should be finalized by UI designer

Use Case 2.8: Redo

Use-Case:	Redo
Primary Actor:	Instructor
Goal in Context:	Lets the user Redo an Undone action
Preconditions:	User is editing a site and has made at least one change and then pressed Undo
Trigger:	User clicks Redo button
Key Shortcut:	Ctrl-Y
Scenario:	• User edits site data • User presses undo • User decides to keep previous version so presses Redo
Exceptions:	N/A

Priority:	Essential, must be implemented
When available:	Second Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of button should be finalized by UI designer

Use Case 2.9: About

Use-Case:	About
Primary Actor:	Instructor
Goal in Context:	User wishes to learn a little about the app
Preconditions:	Application has started
Trigger:	Clicking on About button
Key Shortcut:	N/A
Scenario:	• User is viewing user interface with any tab activated • User clicks About button • User views dialog that pops up that has information about the app • User clicks Ok to close dialog
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of button should be finalized by UI designer

Use Case 2.10: Edit Course Info

Use-Case:	Edit Course Info
Primary Actor:	Instructor
Goal in Context:	User wishes to edit basic course details
Preconditions:	A Course Web site is being edited
Trigger:	Any of the course details controls being edited
Key Shortcut:	N/A
Scenario:	• User is viewing the Course Details Tab • User enters course info into Course Info controls
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of button should be finalized by UI designer

Use Case 2.11: Select Export Directory

Use-Case:	Select Export Directory
Primary Actor:	Instructor
Goal in Context:	User wishes to select directory where site will be exported to
Preconditions:	User is editing a site
Trigger:	User clicks Change button
Key Shortcut:	N/A
Scenario:	• User is viewing the Course Details Tab • User clicks on Change button • User selects directory for exporting to • User clicks Ok, which will now be remembered and should enable Export button
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of UI should be finalized by UI designer

Use Case 2.12: Select Template Directory

Use-Case:	Select Template Directory
Primary Actor:	Instructor
Goal in Context:	User wishes to select template for site
Preconditions:	A site template with used pages and resources already exists in a subdirectory of work/template
Trigger:	User clicks on Select Template button
Key Shortcut:	N/A
Scenario:	• User is viewing Course Details Tab UI • User clicks on Select Template button • User navigates to directory with desired template and selects directory • User clicks Ok button, which should detect .html files in directory that are either index.html, syllabus.html, schedule.html, hws.html, or projects.html, and loads them into table
Exceptions:	Must be a directory containing a valid template.
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of dialog and button should be finalized by UI designer

Use Case 2.13: Toggle Use Template Page

Use-Case:	Toggle Use Template Page
Primary Actor:	Instructor
Goal in Context:	User may wish to include/exclude a particular page that is in the site template

Preconditions:	User has already specified the template directory
Trigger:	User clicks on checkbox in table
Key Shortcut:	N/A
Scenario:	• User is viewing UI on Course Details Tab • User clicks checkbox next to page to toggle and thus include/exclude from site, which would affect if it's exported and also included in navbar
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of dialog and button should be finalized by UI designer

Use Case 2.14: Select Branding Images

Use-Case:	Select Branding Images
Primary Actor:	Instructor
Goal in Context:	User wishes to select images for branding site
Preconditions:	User has placed nav bar and footer branding images for universities in work/brands directory
Trigger:	User selects image from file selector
Key Shortcut:	N/A
Scenario:	• User is viewing Course Details tab • User clicks on one of the three Change buttons • User navigates to image and selects it • User clicks Ok • Image loaded and drawn in place
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of dialog and button should be finalized by UI designer

Use Case 2.15: Select Stylesheet

Use-Case:	Select Stylesheet
Primary Actor:	Instructor
Goal in Context:	The user wishes to select the color/font stylesheet for the site
Preconditions:	Color/Font stylesheets like sea_wolf.css have been created and placed in work/css directory so they may be found by application and loaded into combo box
Trigger:	User selects css from combo box
Key Shortcut:	N/A
Scenario:	• User is viewing UI on Course Details Tab • User selects stylesheet from combo box
Exceptions:	This is only from color/font style. Note that layout is fixed and cannot be changed.

Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of dialog and button should be finalized by UI designer

Use Case 2.16: Add Teaching Assistant

Use-Case:	Add Teaching Assistant
Primary Actor:	Instructor
Goal in Context:	User wishes to add a new teaching assistant
Preconditions:	User is editing a Course Site
Trigger:	User clears teaching assistant controls of data
Scenario:	• User is viewing user interface on TAs Data Tab • If a teaching assistant is selected in table, user can press Clear button to start adding a new teaching assistant, otherwise user can just start editing teaching assistant properties • User enters new teaching assistant details in teaching assistant controls and presses Add TA button
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	First Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of button & screens should be finalized by UI designer

Use Case 2.17: Edit Teaching Assistant

Use-Case:	Edit Teaching Assistant
Primary Actor:	Instructor
Goal in Context:	User wishes to change details for a teaching assistant
Preconditions:	At least one teaching assistant has been added
Trigger:	User selects a teaching assistant
Scenario:	• User is viewing user interface • User clicks on a student in the teaching assistant table, which loads teaching assistant properties into controls • User edits teaching assistant properties in editing controls • User clicks Update button to commit changes
Exceptions:	Must have unique name and email
Priority:	Essential, must be implemented
When available:	First Benchmark
Frequency of use:	Many times per session
Open Issues:	Size, location, and style of button & screens should be finalized by UI designer

Use Case 2.18: Remove Teaching Assistant

Use-Case:	Remove Teaching Assistant
Primary Actor:	Instructor
Goal in Context:	User wishes to remove one of the teaching assistants
Preconditions:	At least one teaching assistant has been added
Trigger:	User presses the '-' button or Delete key
Scenario:	• User is viewing user interface • If not on the TAs Data tab, user must click on that tab to activate • User selects a Teaching Assistant from the list • User either presses '-' button or delete key • User verifies that they wish to remove teaching assistant and all that affects
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	First Benchmark
Frequency of use:	Many times per session
Open Issues:	Appropriate size, location, and style of buttons & Screens to be determined by UI designer

Use Case 2.19: Toggle TA Undergrad

Use-Case:	Toggle TA Undergrad
Primary Actor:	Instructor
Goal in Context:	User wishes to specify TA as either a grad or undergrad TA, which means they will be listed in different sections of the syllabus page
Preconditions:	At least one TA has been added
Trigger:	Clicking checkbox in TA table
Scenario:	• User is viewing the user interface on TAs Data Tab • User clicks checkbox next to one of TAs in table to toggle on (as undergrad) or off (as grad) TA, which will affect where in syllabus page TA will appear
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	First Benchmark
Frequency of use:	Many times per session.
Open Issues:	Size, location, and style of Screen should be finalized by UI designer

Use Case 2.20: Toggle TA Office Hours

Use-Case:	Toggle TA Office Hours
Primary Actor:	Instructor
Goal in Context:	Player wishes to add/remove a TA from a time slot
Preconditions:	At least one TA has been added
Trigger:	User clicks on office hours grid cell
Scenario:	• User is viewing the user interface on TAs Data Tab • User selects a TA from table

	• User clicks on a cell in grid to toggle office hours on or off
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	First Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.21: Change Start/End Office Hours

Use-Case:	Change Start/End Office Hours
Primary Actor:	Instructor
Goal in Context:	User wishes to change the start and end times for the office hours grid
Preconditions:	User is editing a Course Site
Trigger:	User updates one of the times
Scenario:	• User is viewing the user interface on TAs Data Tab • User changes either start time or end time via combo box • User verifies (or cancels) change if it affects other data
Exceptions:	Start time must be before end time
Priority:	Essential, must be implemented
When available:	First Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.22: Add Recitation

Use-Case:	Add Recitation
Primary Actor:	Instructor
Goal in Context:	User wishes to add a new recitation
Preconditions:	User is editing a Course Site
Trigger:	User clears recitation controls of data
Scenario:	• User is viewing user interface on Recitations Data Tab • If a recitation is selected in table, user can press Clear button to start adding a new recitation, otherwise user can just start editing recitation properties • User enters new recitation details in recitation controls and presses Add Recitation button
Exceptions:	Recitation must have section, time, and location. TAs can only be added to Recitation after they've been added to TA list.
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.23: Edit Recitation

Use-Case:	Edit Recitation
Primary Actor:	Instructor
Goal in Context:	User wishes to change details for a recitation
Preconditions:	At least one recitation has been added
Trigger:	User selects a recitation
Scenario:	• User is viewing user interface • User clicks on a recitation in the student table, which loads recitation properties into controls • User edits reitation properties in editing controls • User clicks Update button to commit changes
Exceptions:	Must ensure all required data (section, day, location) are required at time of update
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.24: Remove Recitation

Use-Case:	Remove Recitation
Primary Actor:	Instructor
Goal in Context:	User wishes to remove one of the recitations
Preconditions:	At least one recitation has been added
Trigger:	User presses the '-' button or Delete key
Scenario:	• User is viewing user interface • If not on the Recitations Data tab, user must click on that tab to activate • User selects a Recitation from the list • User either presses '-' button or delete key • User verifies that they wish to remove recitation and all that affects
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.25: Edit start and end dates

Use-Case:	Edit start and end dates
Primary Actor:	Instructor
Goal in Context:	User wishes to change the start and end dates for the course schedule
Preconditions:	User is editing a Course Site
Trigger:	User updates one of the dates
Scenario:	• User is viewing the user interface on Schedule Data Tab

	• User changes either start date or end date via date picker • User verifies (or cancels) change if it affects other data
Exceptions:	Start date must be before end date. Also, start date must be Monday, end date a Friday
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.26: Add Schedule Item

Use-Case:	Add Schedule Item
Primary Actor:	Instructor
Goal in Context:	User wishes to add a new schedule item
Preconditions:	User is editing a Course Site
Trigger:	User clears schedule item controls of data
Scenario:	• User is viewing user interface on Schedule Data Tab • If a schedule item is selected in table, user can press Clear button to start adding a new schedule item, otherwise user can just start editing schedule item properties • User enters new schedule item details in schedule item controls and presses Add Schedule Item button
Exceptions:	Must have legal data for type of item
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.27: Edit Schedule Item

Use-Case:	Edit Schedule Item
Primary Actor:	Instructor
Goal in Context:	User wishes to change details for a schedule item
Preconditions:	At least one schedule item has been added
Trigger:	User selects a schedule item
Scenario:	• User is viewing user interface • User clicks on a schedule item in the student table, which loads schedule item properties into controls • User edits schedule item properties in editing controls • User clicks Update button to commit changes
Exceptions:	Must have legal data for type of item
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.28: Remove Schedule Item

Use-Case:	Remove Schedule Item
Primary Actor:	Instructor
Goal in Context:	User wishes to remove one of the schedule items
Preconditions:	At least one schedule item has been added
Trigger:	User presses the '-' button or Delete key
Scenario:	• User is viewing user interface • If not on the Schedule Data tab, user must click on that tab to activate • User selects a Schedule Item from the list • User either presses '-' button or delete key • User verifies that they wish to remove schedule item and all that affects
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.29: Add Team

Use-Case:	Add Team
Primary Actor:	Instructor
Goal in Context:	User wishes to add a new team
Preconditions:	User is editing a Course Site
Trigger:	User clears team controls of data
Scenario:	• User is viewing user interface on Projects Tab • If a team is selected in table, user can press Clear button to start adding a new team, otherwise user can just start editing team properties • User enters new team details in student controls and presses Add Team button
Exceptions:	Must have all data
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.30: Edit Team

Use-Case:	Edit Team
Primary Actor:	Instructor
Goal in Context:	User wishes to change details for a team
Preconditions:	At least one team has been added
Trigger:	User selects a team
Scenario:	• User is viewing user interface • User clicks on a team in the student table, which loads team properties into controls

	• User edits team properties in editing controls • User clicks Update button to commit changes
Exceptions:	Must have all data
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.31: Remove Team

Use-Case:	Remove Team
Primary Actor:	Instructor
Goal in Context:	User wishes to remove one of the teams
Preconditions:	At least one team has been added
Trigger:	User presses the '-' button or Delete key
Scenario:	• User is viewing user interface • If not on the Projects Data tab, user must click on that tab to activate • User selects a Team from the list • User either presses '-' button or delete key • User verifies that they wish to remove team and all that affects
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.32: Add Student

Use-Case:	Add Student
Primary Actor:	Instructor
Goal in Context:	User wishes to add a new student
Preconditions:	User is editing a Course Site
Trigger:	User clears student controls of data
Scenario:	• User is viewing user interface on Projects Tab • If a student is selected in table, user can press Clear button to start adding a new student, otherwise user can just start editing student properties • User enters new student details in student controls and presses Add Student button
Exceptions:	Must have unique name and email
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.33: Edit Student

Use-Case:	Edit Student
Primary Actor:	Instructor
Goal in Context:	User wishes to change details for a student
Preconditions:	At least one student has been added
Trigger:	User selects a student
Scenario:	• User is viewing user interface on Projects Tab • User clicks on a student in the student table, which loads student properties into controls • User edits student properties in editing controls • User clicks Update button to commit changes
Exceptions:	Must have unique name and email
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

Use Case 2.34: Remove Student

Use-Case:	Remove Student
Primary Actor:	Instructor
Goal in Context:	User wishes to remove one of the students
Preconditions:	At least one student has been added
Trigger:	User presses the '-' button or Delete key
Scenario:	• User is viewing user interface • If not on the Projects Data tab, user must click on that tab to activate • User selects a Student from the list • User either presses '-' button or delete key • User verifies that they wish to remove student and all that affects
Exceptions:	N/A
Priority:	Essential, must be implemented
When available:	Third Benchmark
Frequency of use:	Many times per session
Open Issues:	Precise size, location, and style of window should be finalized by UI designer.

i. Hardware Interfaces

The application should be runnable on any platform that supports Java, but would require a keyboard and mouse.

ii. Software Interfaces

Course Site Generatoro will be developed using the Java language. Note that since it is a traditional workspace-type application, it may be best to use the Desktop Java Framework. Note that the site exported by this application should be able to be deployed to any Web server and work via any Web browser, namely Chrome and Firefox. Note that it is recommended that Web Server for Chrome and the Chrome browser be used in cooperation to test exporting.

iii. Communications Interfaces

Note that this editing application will operate locally. There will be no networking requirements.

iv. Memory Constraints

This application uses a manageable amount of user provided data so this should not be a concern.

v. Operations

We must make sure the data exported to JSON files use the proper format according to current Web page requirements per the example files given with the one noted exception (Grad/Undergrad TAs).

vi. Site Adaptation Requirements

N/A

b. Product functions

N/A.

c. User characteristics

The editor should aim to be as user friendly as possible, using the principles of foolproof design as well as sound UI design principles.

d. Constraints

N/A

e. Assumptions and dependencies

N/A

f. Apportioning of the Requirements

N/A

3 Specific requirements

The Course Site Generator application will require a number of user interface contexts and components including:

- File Toolbar
- Edit Toolbar
- Tabbed Pane
    - Course Details Pane
    - TA Data Pane
    - Recitation Data Pane
    - Schedule Data Pane
    - Project Data Pane

Note that the user should be free to navigate between these five panes as desired.

3.1 **External interfaces**

The following wireframe mockups provide a look at the types of controls and layout to be used for the User Interface. Note that the User Interface designer should select the appropriate icons for all buttons and should carefully choose color and font combinations that provide good contrast and attract the eye. There are five UI diagrams, one for each Tab. Note that the User Interface designer should also consider additional dialogs for providing adequate feedback to the user as well as for navigation through the file system to select files and directories as part of certain use cases.

Figure 3.1 Course Details Tab

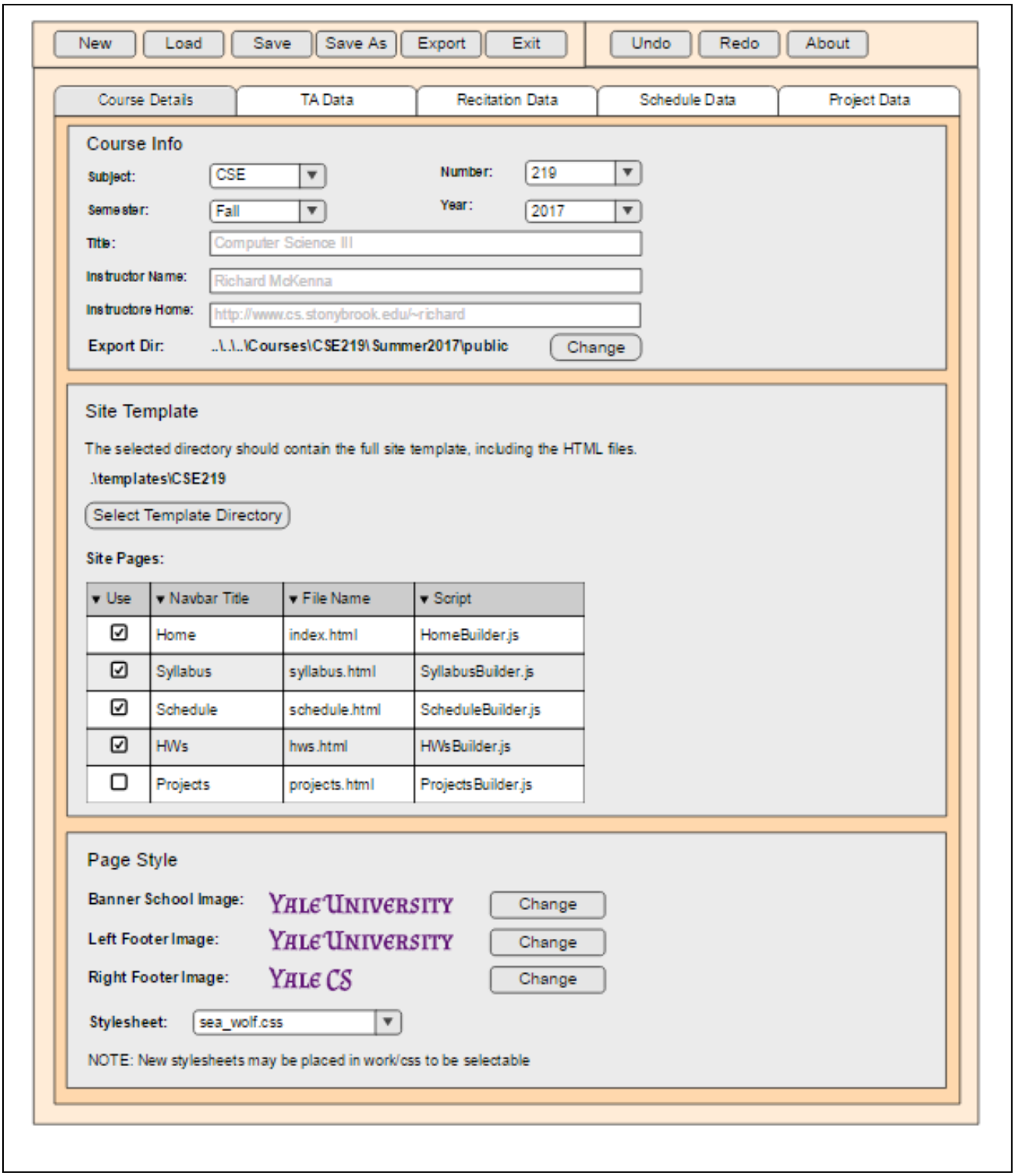


Figure 3.2 TA Data Tab

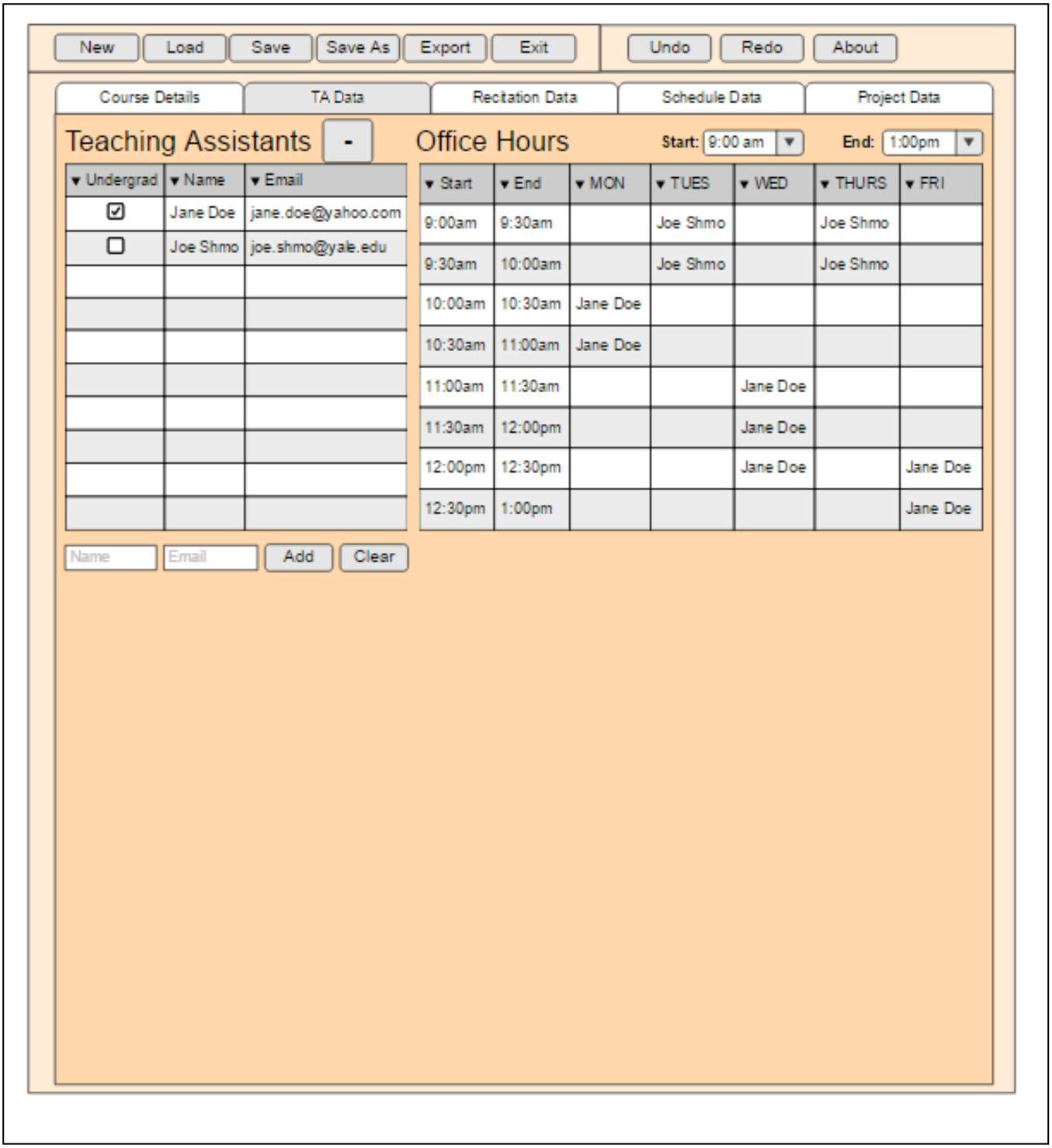


Figure 3.3 Recitation Data Tab

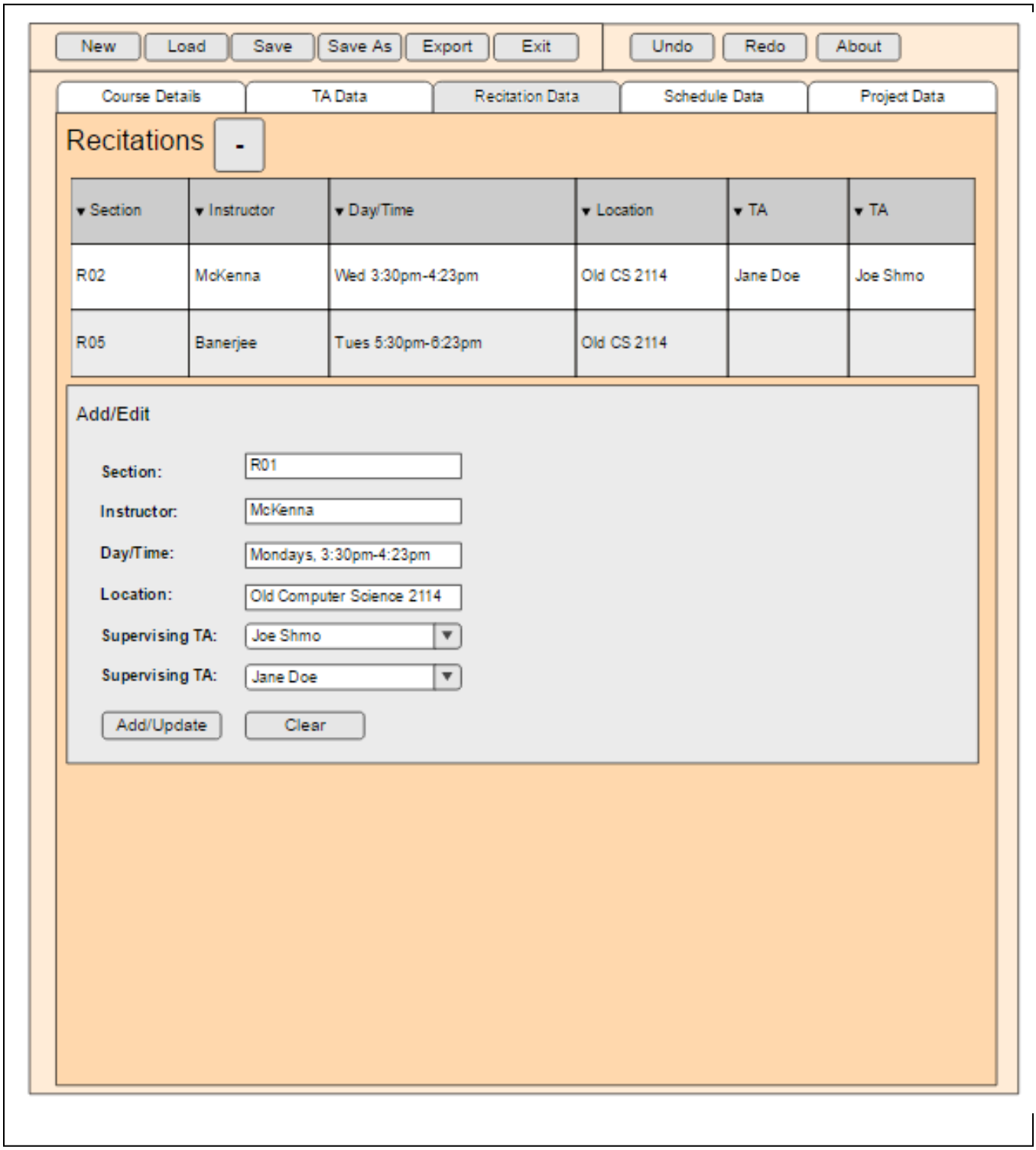


Figure 3.4 Schedule Data Tab

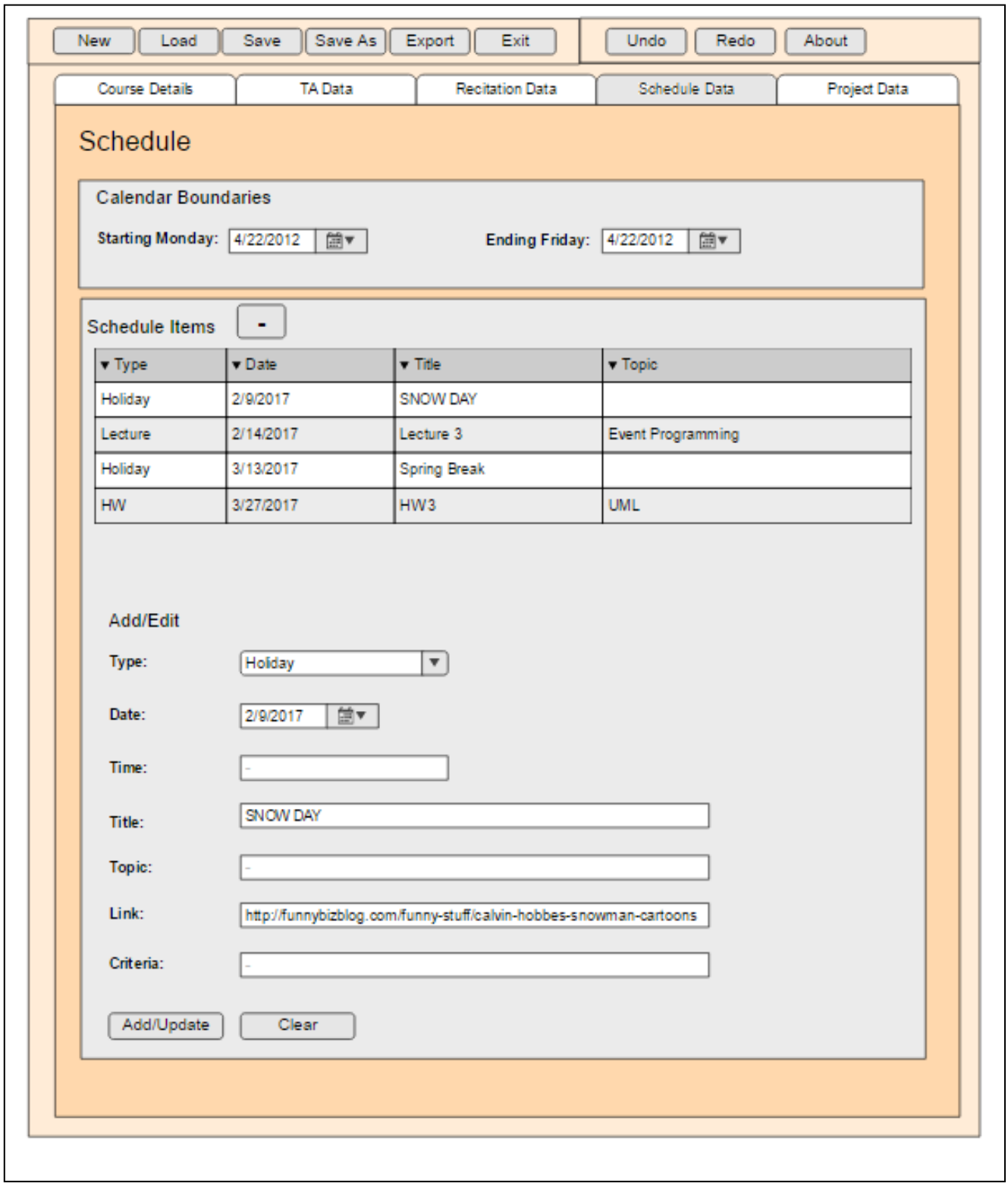
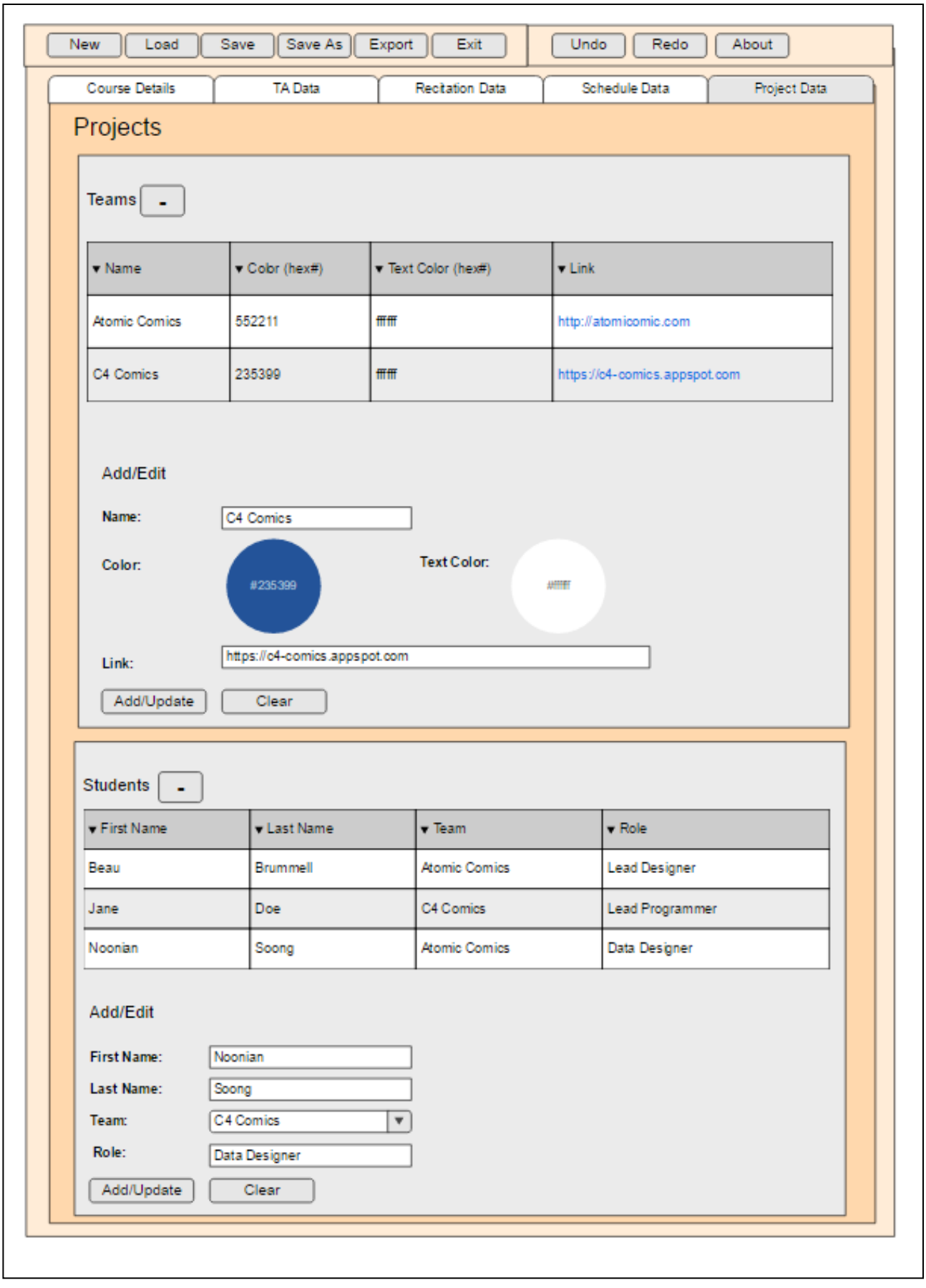


Figure 3.5 Project Data Tab



One of the important things to consider in our application is providing the appropriate feedback to the user. Users need feedback to enjoy their experience. This is typically done with visual cues like dialog boxes.

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.

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 for free.

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

3.6.4 Extensibility – It is possible that more JavaScript/JSON widgets might be added to course sites in the future, so by providing an additional tab with suitable data and making changes to exporting methods, this should be considered during this deaign.

1.6.5 Portability – To start with, the app will target desktop Java applications.

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 and instructors should make updates to the 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