The Course Site Generator™

Software Design Description

Author: Ajay Sarjoo
CSE 219 - L02
October 2018
Version 1.0

Based on IEEE Std 1016TM-2009 document format

1 Introduction

This is the Software Design Description (SDD) for the Course Site Generator™ website generating application. Note that this document format is based on the IEEE Standard 1016-2009 recommendation for software design.

1.1 Purpose

This document is to serve as the blueprint for the construction of the Course Site Generator application. This design will use UML class diagrams to provide complete detail regarding all packages, classes, instance variables, class variables, and method signatures needed to build the application. In addition, UML Sequence diagrams will be used to specify object interactions post-initialization of the application, meaning in response to user interactions or timed events.

1.2 Scope

For this project the goal is for instructors to easily make and update course websites. 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.

1.4 References

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

1.5 Overview

This Software Design Description document provides a working design for the Course Site Generator software application as described in the Course Site Generator Software Requirements Specification. Note that all parties in the implementation stage must agree upon all connections between components before proceeding with the implementation stage. Section 2 of this document will provide the Package-Level Viewpoint, specifying the packages and frameworks to be designed. Section 3 will provide the Class-Level Viewpoint, using UML Class Diagrams to specify how the classes should be constructed. Section 4 will provide the Method-Level System Viewpoint, describing how methods will interact with one another. Section 5 provides deployment information like file structures and formats to use. Section 6 provides a Table of Contents, an Index, and References. Note that all UML Diagrams in this document were created using the VioletUML editor.

2 Package-Level Design Viewpoint

As mentioned, this design will encompass both the Course Site Generator application and the Desktop Java Framework to be used in its construction. In building both we will

heavily rely on the Java API to provide services. Following are descriptions of the components to be built, as well as how the Java API will be used to build them.

2.1 Course Site Generator Overview

Figure 2.1 specifies all the components to be developed and places all classes in home packages.

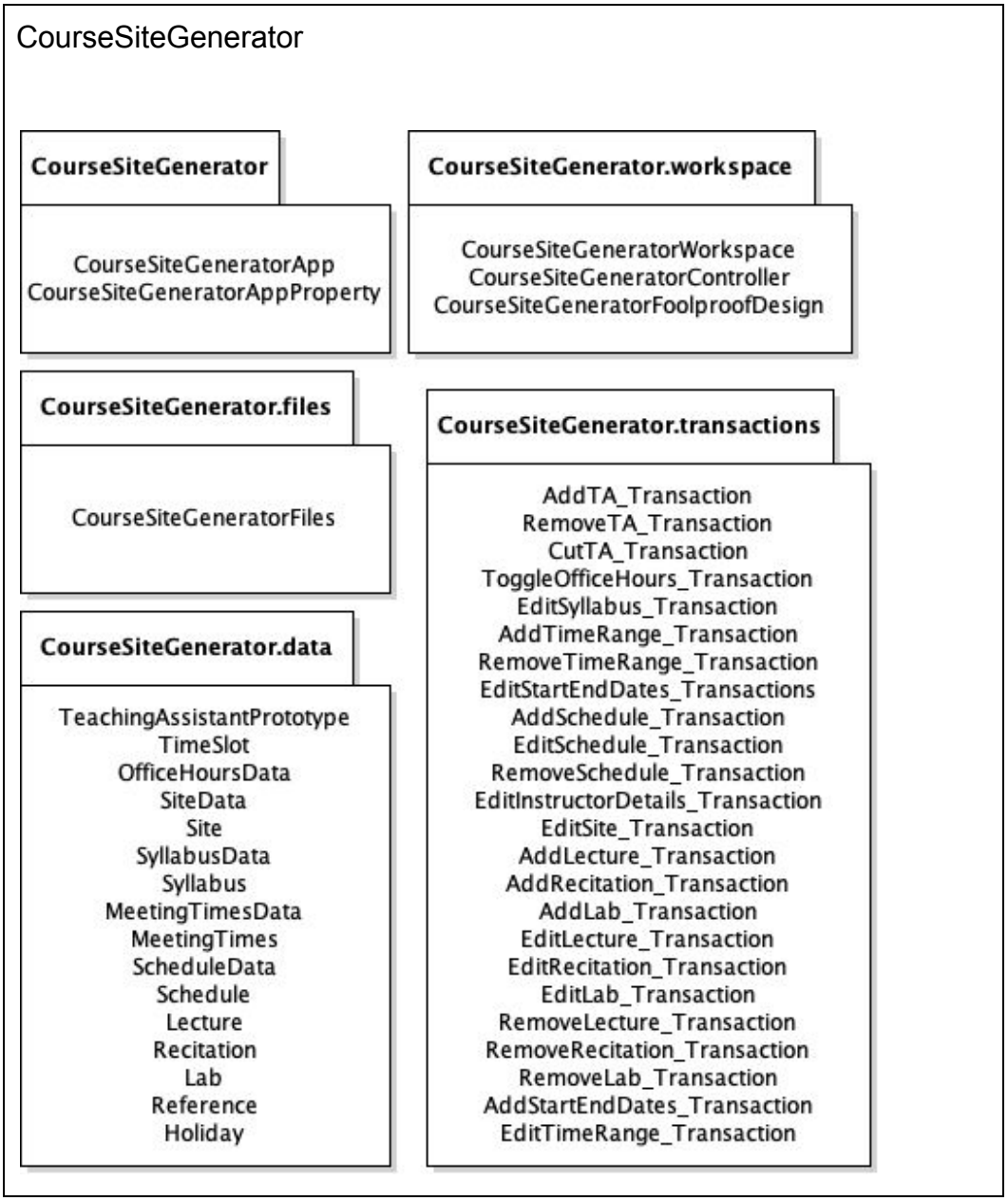

Figure 2.1: Design Packages Overview

2.2 Java API Usage

The application will be developed using the Java programming languages. As such, this design will make use of the classes specified in Figure 2.2.

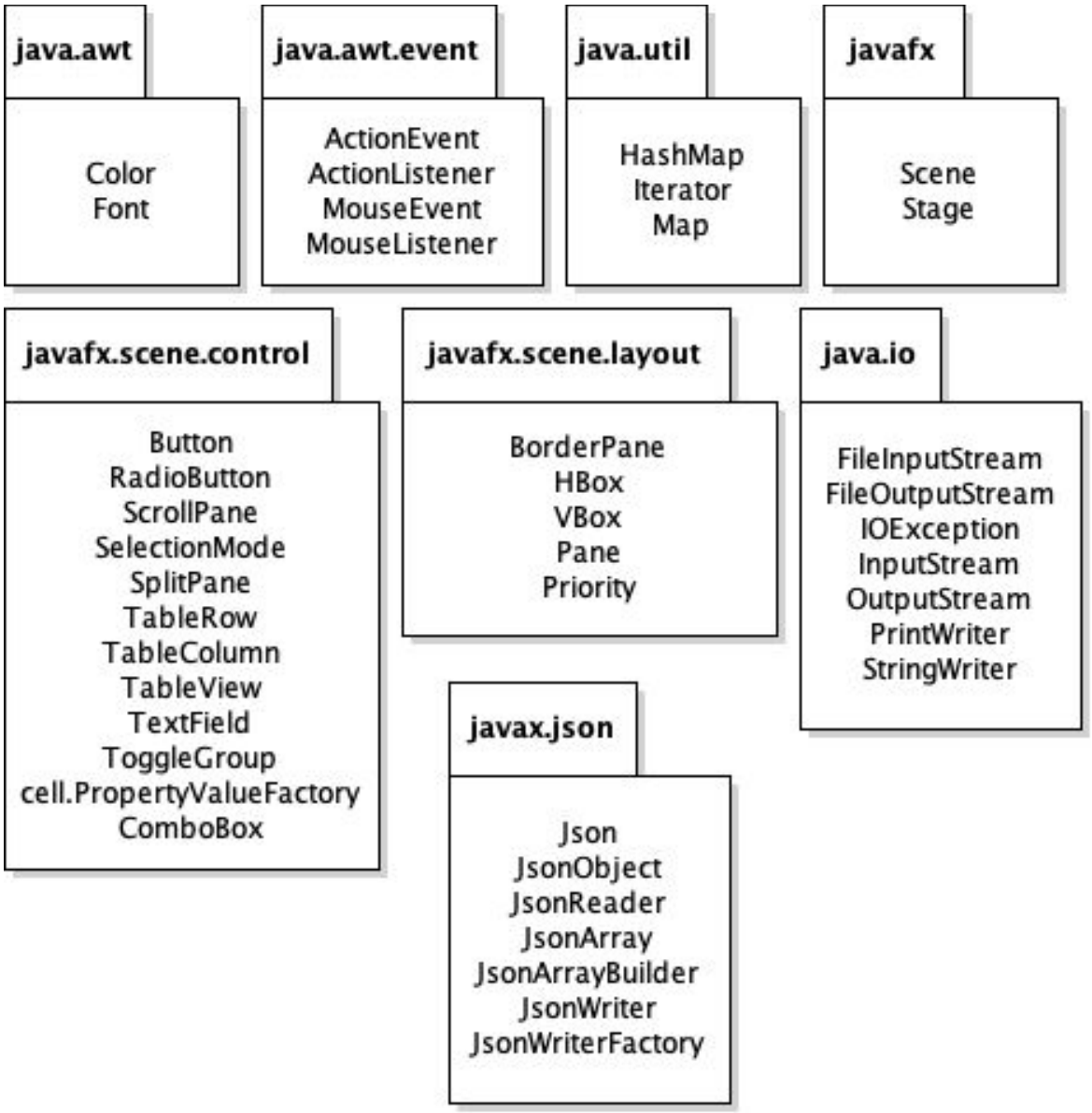

Figure 2.2: Java API Classes and Packages To Be Used

2.3 Java API Usage Descriptions

Tables 2.1-2.9 below summarize how each of these classes will be used. All descriptions comes from the Java and JavaFX API

Class/Interface	Use
Color	For setting the rendering colors for text and UI
Font	For setting the fonts for rendered text

Table 2.1: Uses for classes in the Java API's java.awt package

Class/Interface	Use
ActionEvent	For getting information about an action event like which button was pressed.
ActionListener	For responding to an action event, like a button press. We will provide our own custom implementation of this interface.
MouseEvent	For getting information about a mouse event, like where was the mouse pressed
MouseListener	For responding to a mouse event, like a mouse button press. We will provide our own custom implementation of this interface.

Table 2.2: Uses for classes in the Java API's java.awt.event package

Class/Interface	Use
HashMap	For storing (name,value) key pairs, we'll use it for storing our course information
Iterator	For iterating through a data structure during operations like rendering
Map	An interface that is the parent of HashMap

Table 2.3: Uses for classes in the Java API's java.util package

Class/Interface	Use
Scene	The container for all content in the app, we will use it to construct the UI
Stage	For iterating through a data structure during operations like rendering

Table 2.4: Uses for classes in the Java API's javafx package

Class/Interface	Use
Button	A simple button control. The button control can contain text and/or a graphic
RadioButton	RadioButtons create a series of items where only one item can be selected
ScrollPane	A Control that provides a scrolled, clipped viewport of its contents. It allows the user to scroll the content around either directly (panning) or by using scroll bars
SelectionMode	An enumeration used to specify how many items may be selected in a MultipleSelectionModel
SplitPane	A control that has two or more sides, each separated by a divider, which can be dragged by the user to give more space to one of the sides, resulting in the other side shrinking by an equal amount
TableRow	TableRow is an IndexedCell, but rarely needs to be used by developers creating TableView instances
TableColumn	Each TableColumn in a table is responsible for displaying (and editing) the contents of that column
TableView	The TableView control is designed to visualize an unlimited number of rows of data, broken out into columns

TextField	Text input component that allows a user to enter a single line of unformatted text
ToggleGroup	A class which contains a reference to all Toggles whose selected variables should be managed such that only a single Toggle within the ToggleGroup may be selected at any one time
cell.PropertyValueFactory	A convenience implementation of the Callback interface, designed specifically for use within the TableColumn cell value factory

Table 2.5: Uses for classes in the Java API's javafx.scene.control package

Class/Interface	Use
BorderPane	BorderPane lays out children in top, left, right, bottom, and center positions
HBox	HBox lays out its children in a single horizontal row. If the hbox has a border and/or padding set, then the contents will be layed out within those insets.
VBox	VBox lays out its children in a single vertical column. If the vbox has a border and/or padding set, then the contents will be layed out within those insets
Pane	Base class for layout panes which need to expose the children list as public so that users of the subclass can freely add/remove children
Priority	Enumeration used to determine the grow (or shrink) priority of a given node's layout area when its region has more (or less) space available and multiple nodes are competing for that space

Table 2.6: Uses for classes in the Java API's javafx.scene.layout package

Class/Interface	Use
FileInputStream	A FileInputStream obtains input bytes from a file in a file system. What files are available depends on the host environment
FileOutputStream	A file output stream is an output stream for writing data to a File or to a FileDescriptor. Whether or not a file is available or may be created depends upon the underlying platform
IOException	Signals that an I/O exception of some sort has occurred. This class is the general class of exceptions produced by failed or interrupted I/O operations
InputStream	This abstract class is the superclass of all classes representing an input stream of bytes
OutputStream	This abstract class is the superclass of all classes representing an output stream of bytes. An output stream accepts output bytes and sends them to some sink
PrintWriter	Prints formatted representations of objects to a text-output stream. This class implements all of the print methods found in PrintStream. It does not contain methods for writing raw bytes, for which a program should use unencoded byte streams
StringWriter	A character stream that collects its output in a string buffer, which can then be used to construct a string

Table 2.7: Uses for classes in the Java API's java.io package

Class/Interface	Use
Json	Factory class for creating JSON processing objects. This class provides

	the most commonly used methods for creating these objects and their corresponding factories. The factory classes provide all the various ways to create these objects
JsonObject	JsonObject class represents an immutable JSON object value (an unordered collection of zero or more name/value pairs). It also provides unmodifiable map view to the JSON object name/value mappings
JsonReader	Reads a JSON object or an array structure from an input source
JsonArray	JsonArray represents an immutable JSON array (an ordered sequence of zero or more values). It also provides an unmodifiable list view of the values in the array
JsonArrayBuilder	A builder for creating JsonArray models from scratch. This interface initializes an empty JSON array model and provides methods to add values to the array model and to return the resulting array. The methods in this class can be chained to add multiple values to the array
JsonWriter	Writes a JSON object or array structure to an output source
JsonWriterFactory	Factory to create JsonWriter instances. If a factory instance is configured with some configuration, that would be used to configure the created writer instances

Table 2.8: Uses for classes in the Java API's javax.json package

3 Class-Level Design Viewpoint

As mentioned, this design will encompass the entire Course Site Generator suite. The following UML Class Diagrams reflect this. Note that due to the complexity of the project, we present the class designs using a series of diagrams going from overview diagrams down to detailed ones.

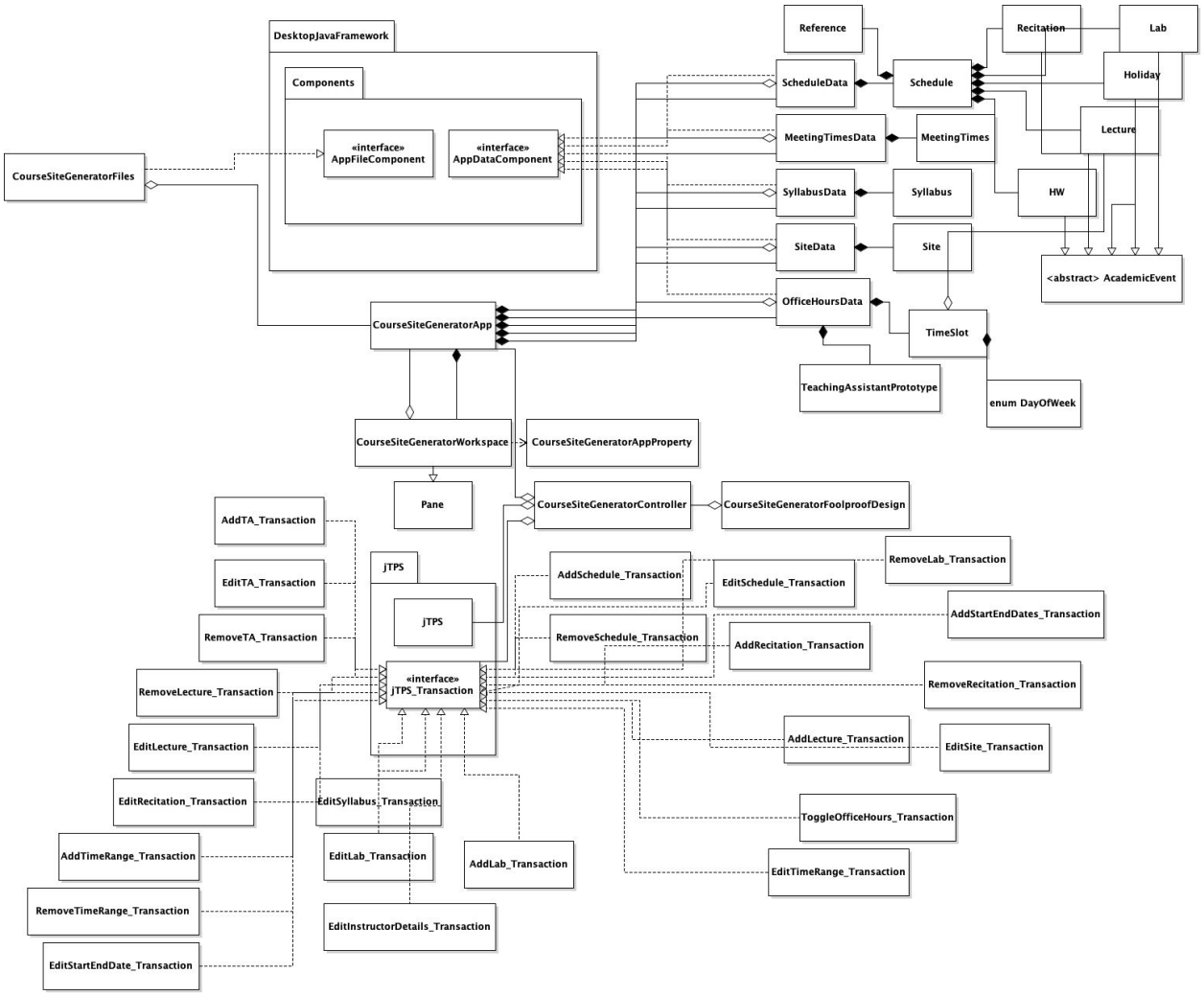

Figure 3.1: Course Site Generator UML Class Diagram

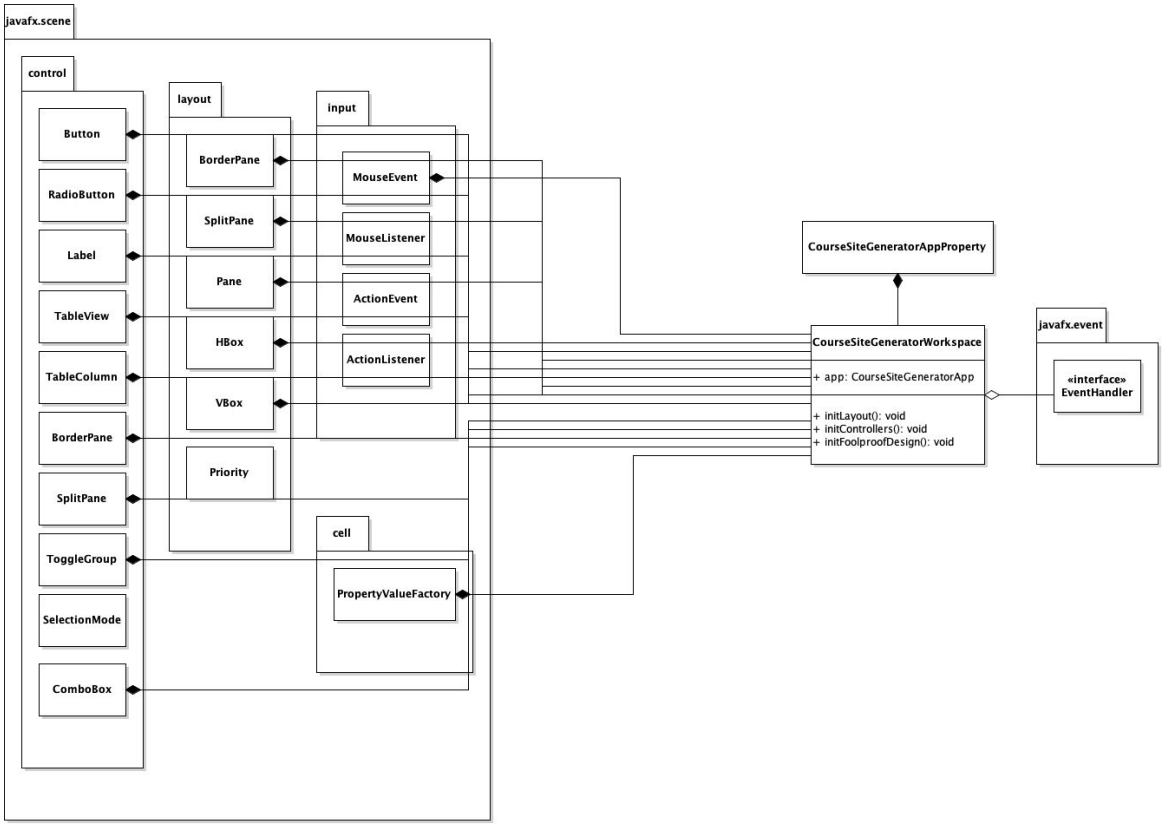

Figure 3.2: Course Site Generator Workspace UML Class Diagram

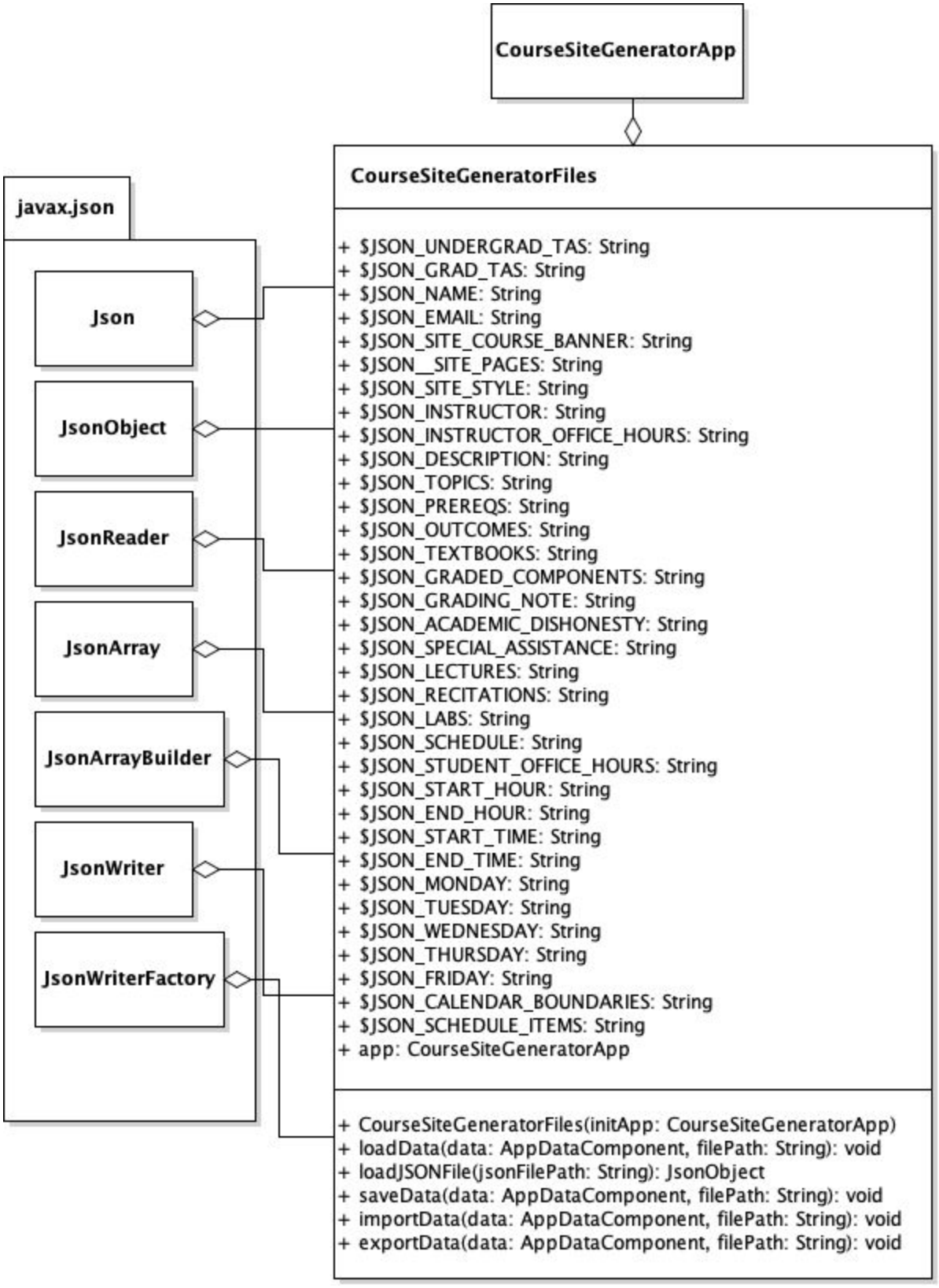

Figure 3.3: Course Site Generator Files UML Class Diagram

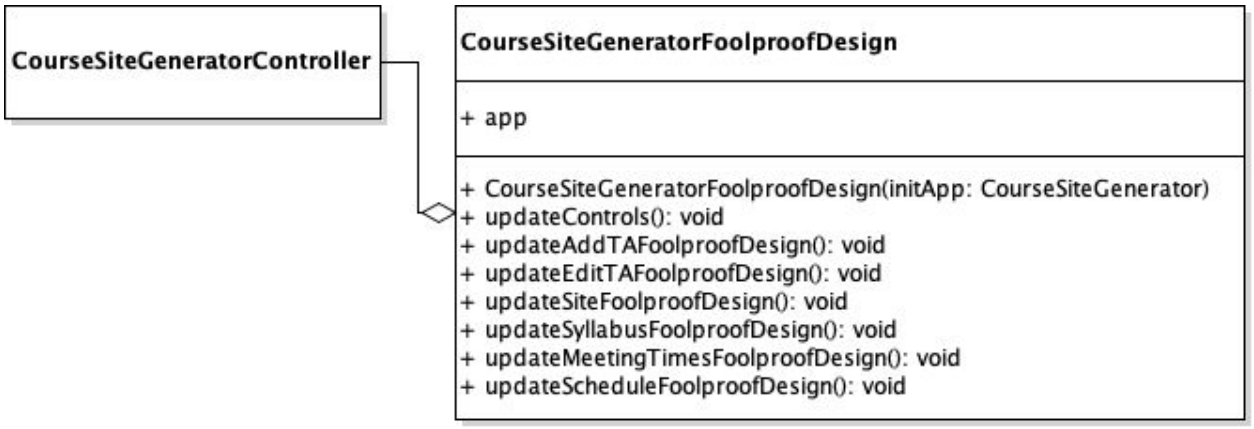

Figure 3.4: Course Site Generator Foolproof Design UML Class Diagram

Figure 3.5: Course Site Generator Controller UML Class Diagram

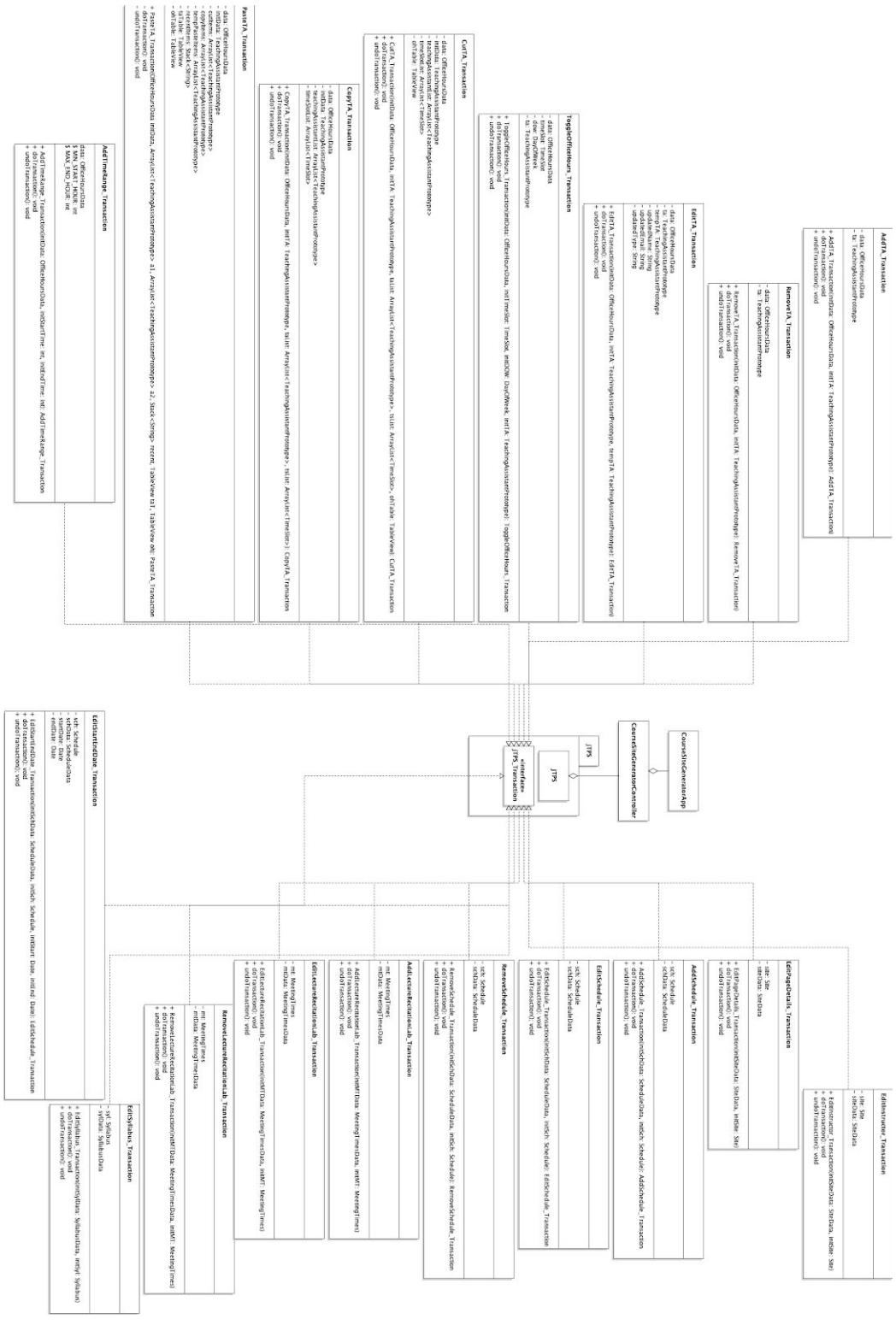

Figure 3.6: Course Site Generator Transactions UML Class Diagram

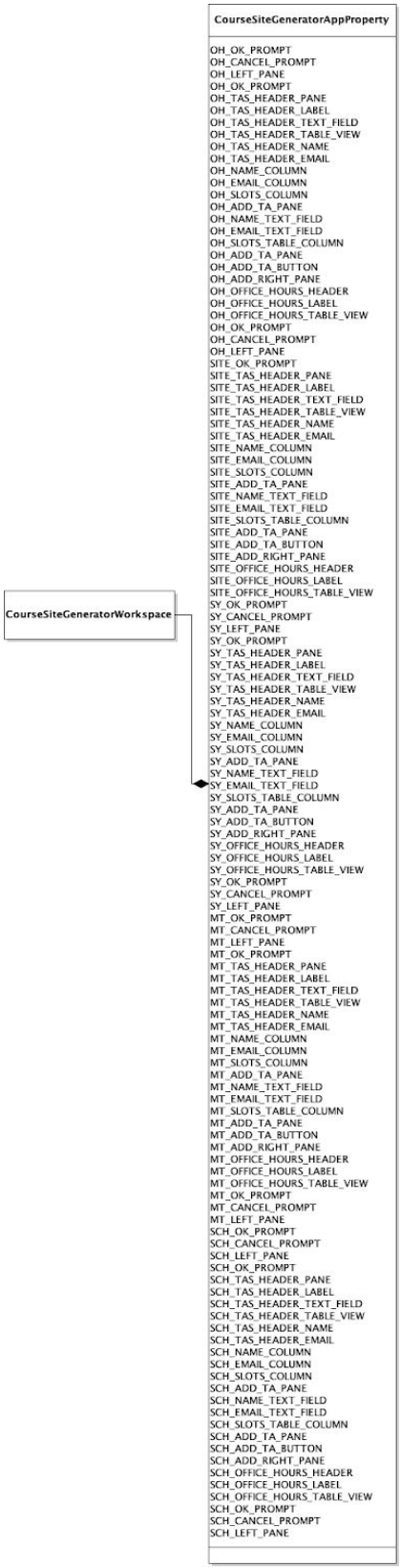

Figure 3.7: Course Site Generator App Properties UML Class Diagram

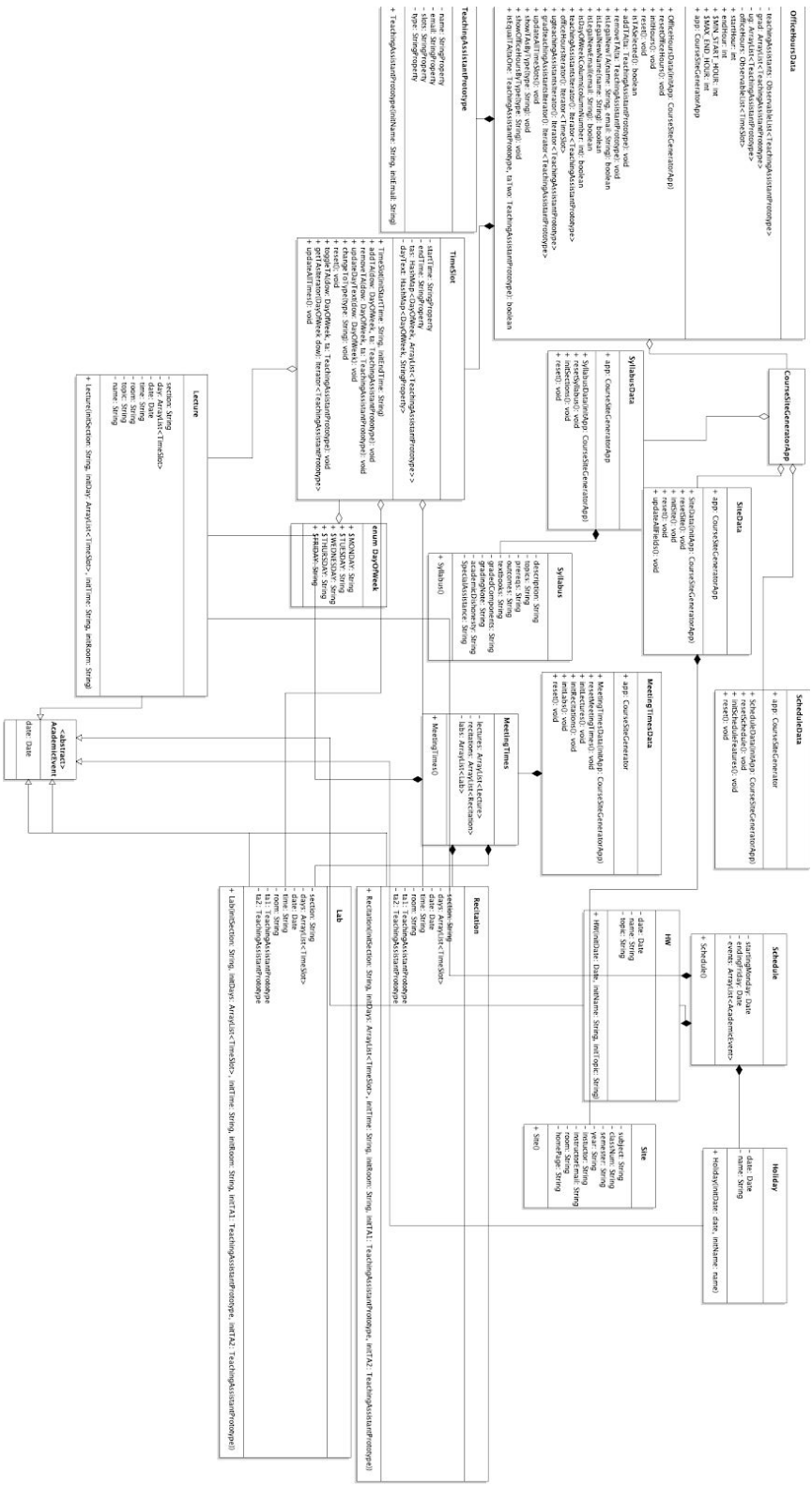

Figure 3.8: Course Site Generator Data UML Class Diagram

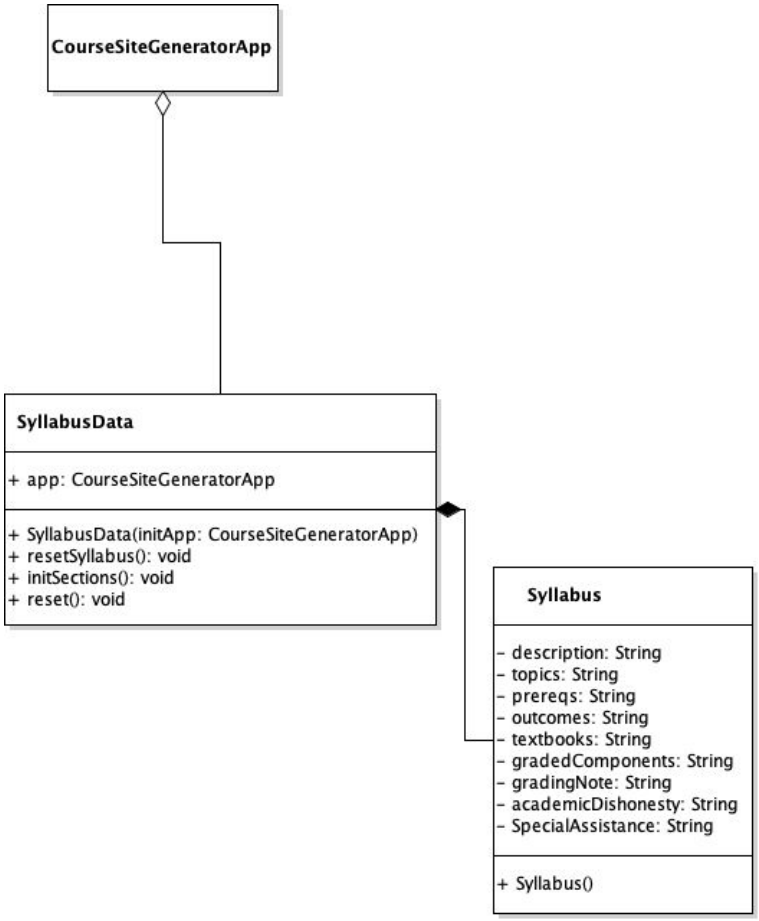

Figure 3.8.1: Course Site Generator Syllabus Data UML Class Diagram

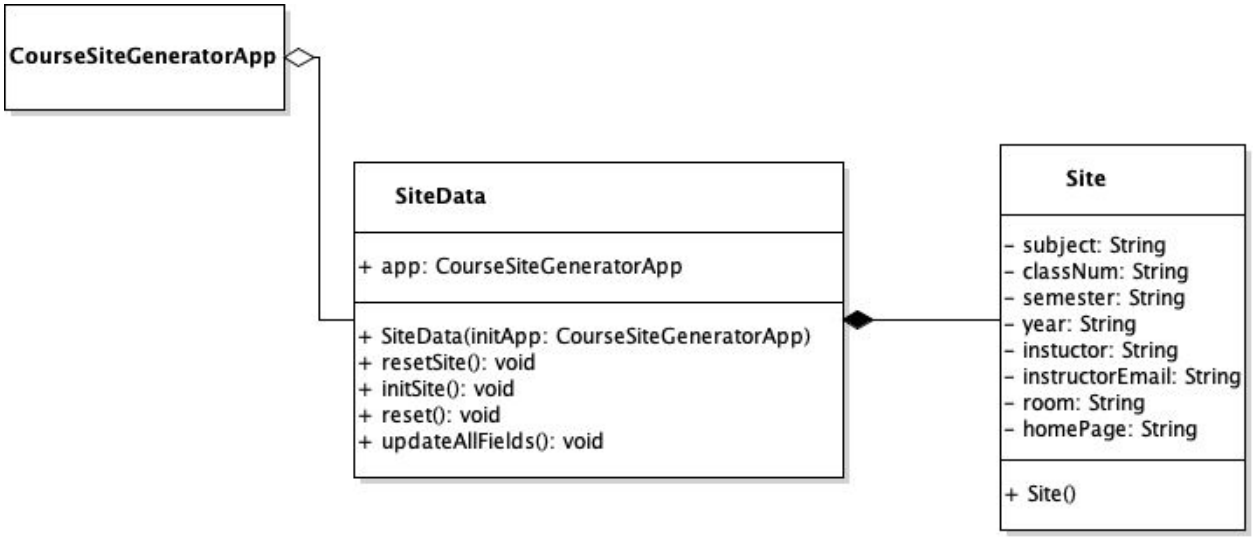

Figure 3.8.2: Course Site Generator Site Data UML Class Diagram

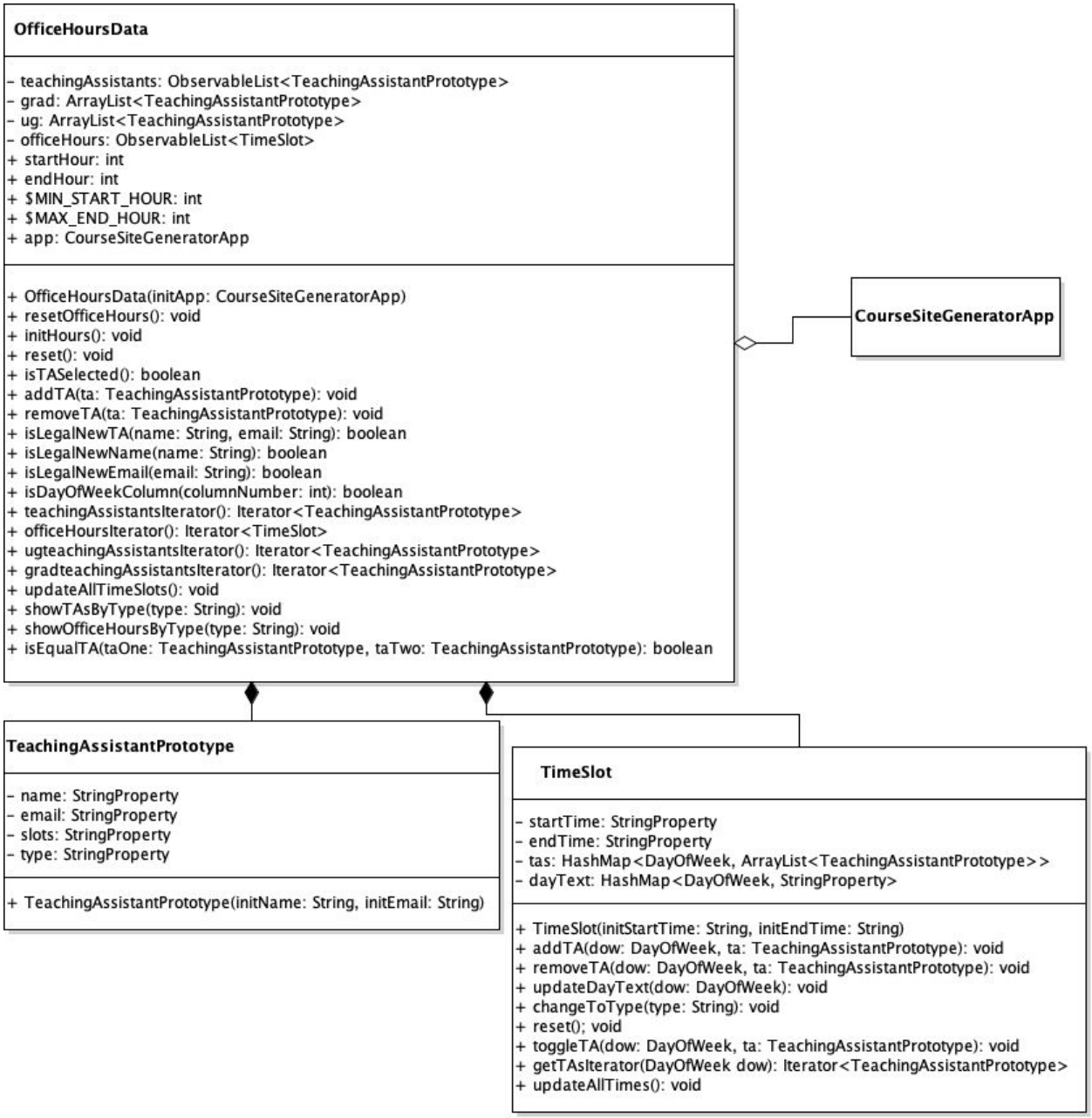

Figure 3.8.3: Course Site Generator Office Hours Data UML Class Diagram

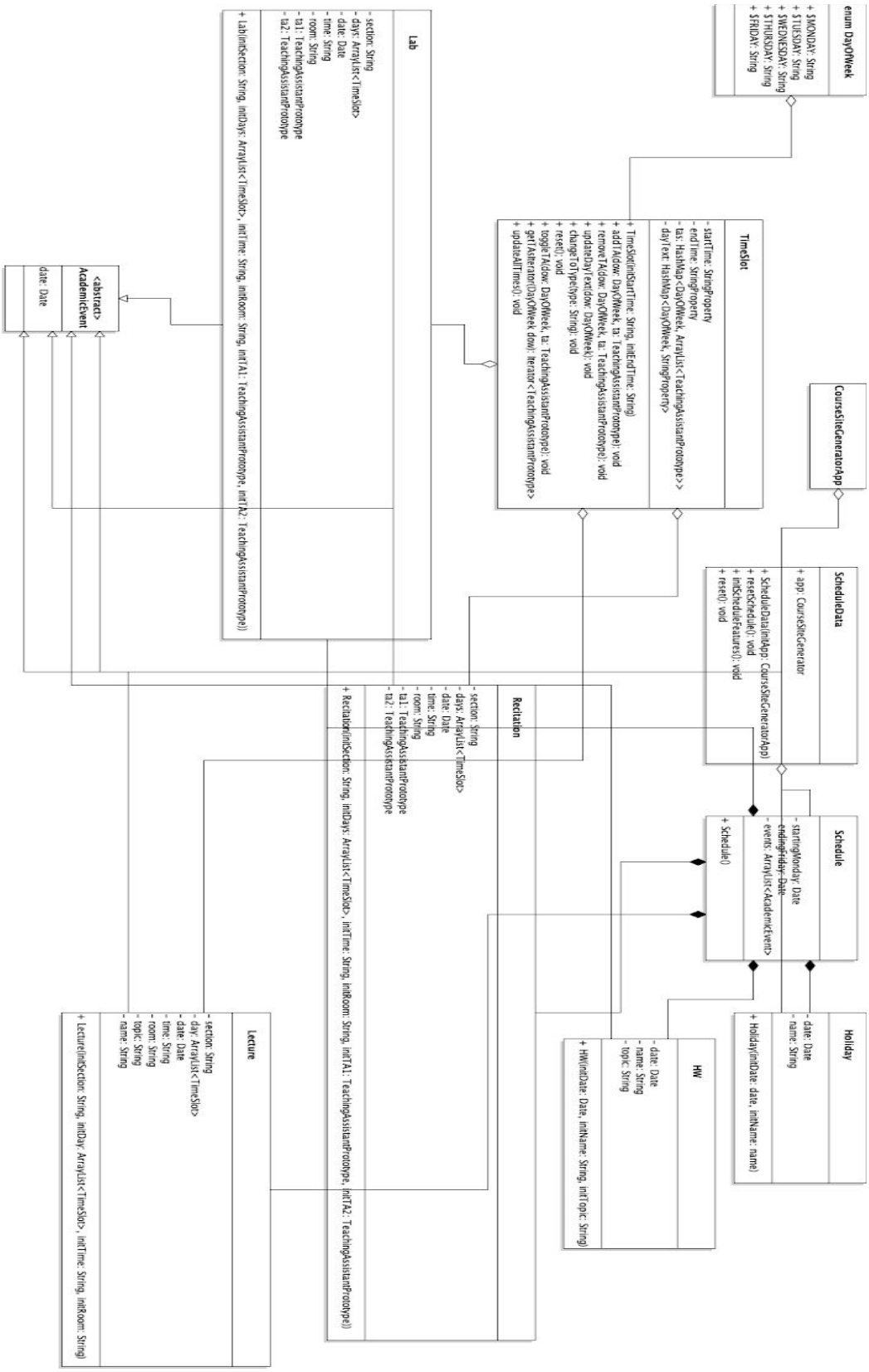

Figure 3.8.4: Course Site Generator Schedule Data UML Class Diagram

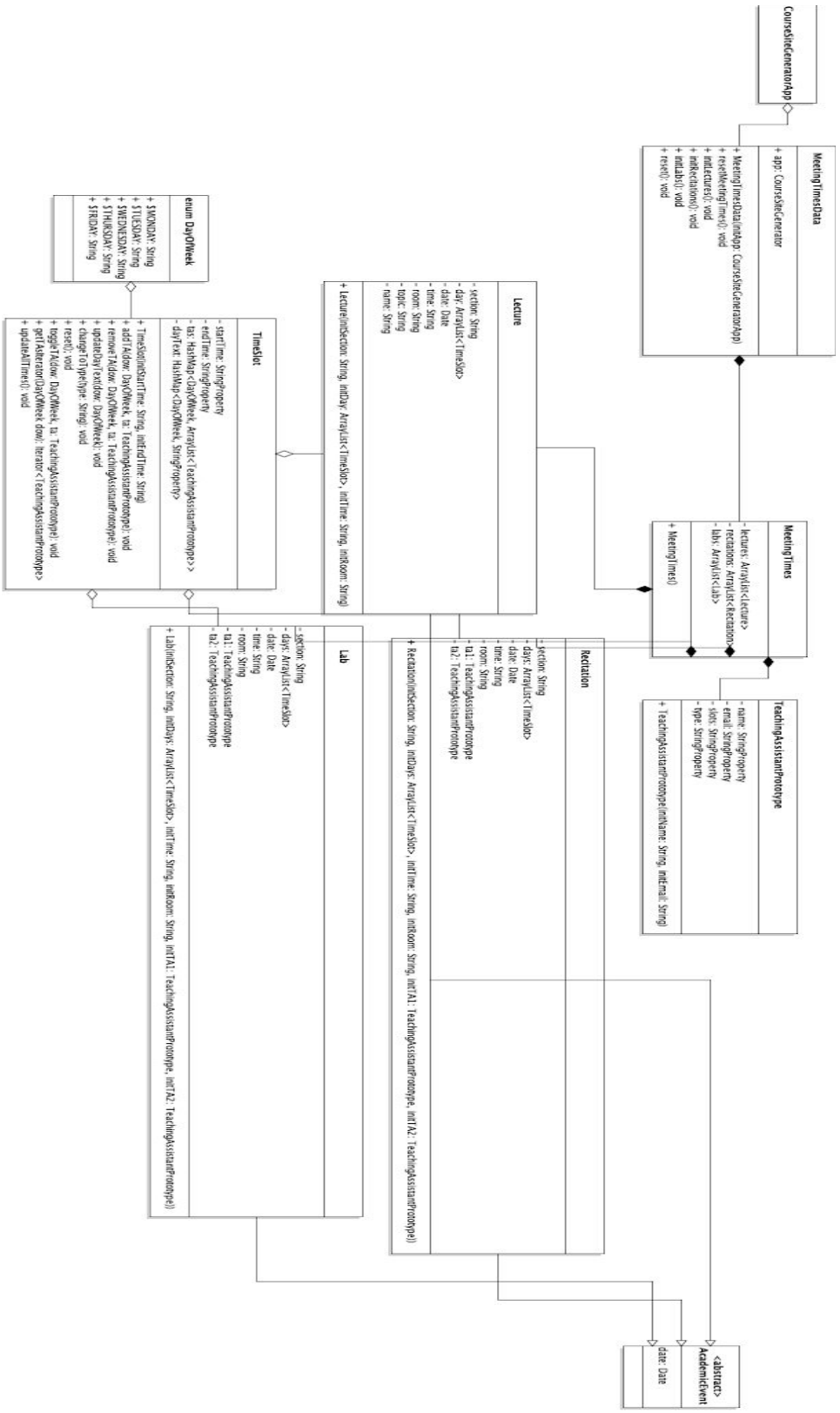

Figure 3.8.5: Course Site Generator Meeting Times Data UML Class Diagram

4 Method-Level Design Viewpoint

Now that the general architecture of the classes has been determined, it is time to specify how data will flow through the system. The following UML Sequence Diagrams describe the methods called within the code to be developed in order to provide the appropriate event responses.

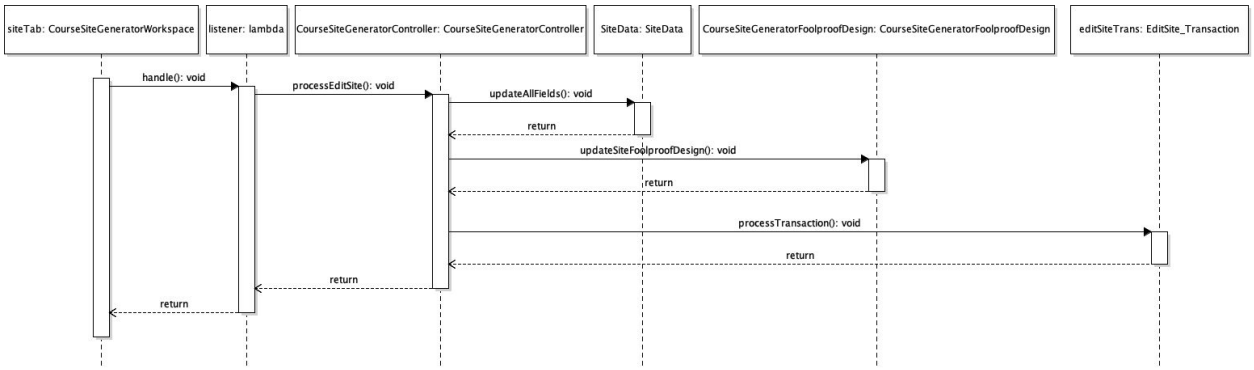

Figure 4.1 Edit Page Details UML Sequence Diagrams

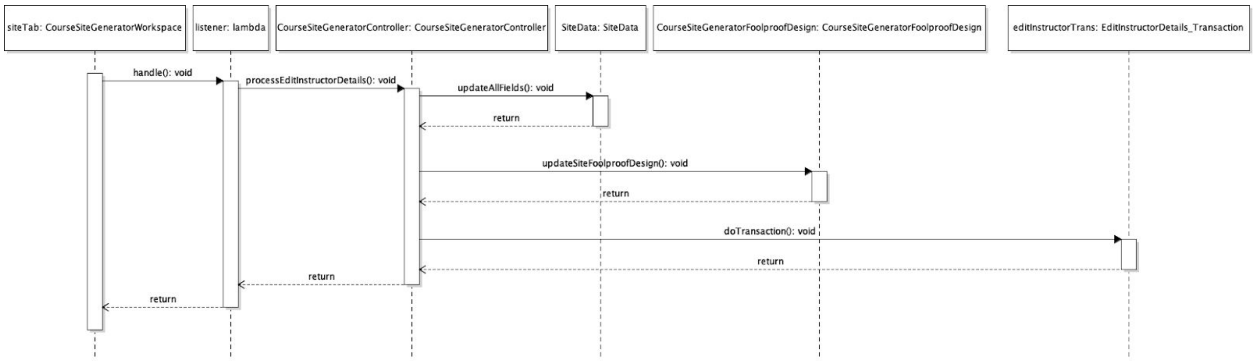

Figure 4.2 Edit Instructor Details UML Sequence Diagrams

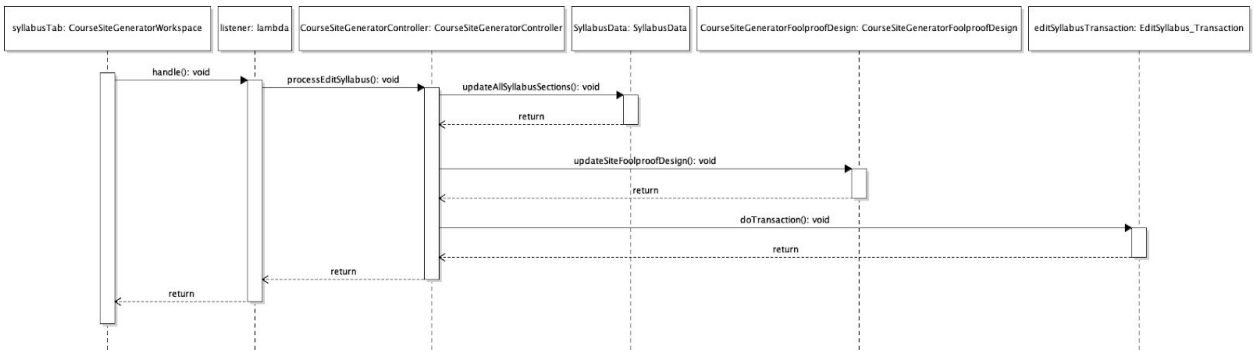

Figure 4.3 Edit Syllabus Details UML Sequence Diagrams

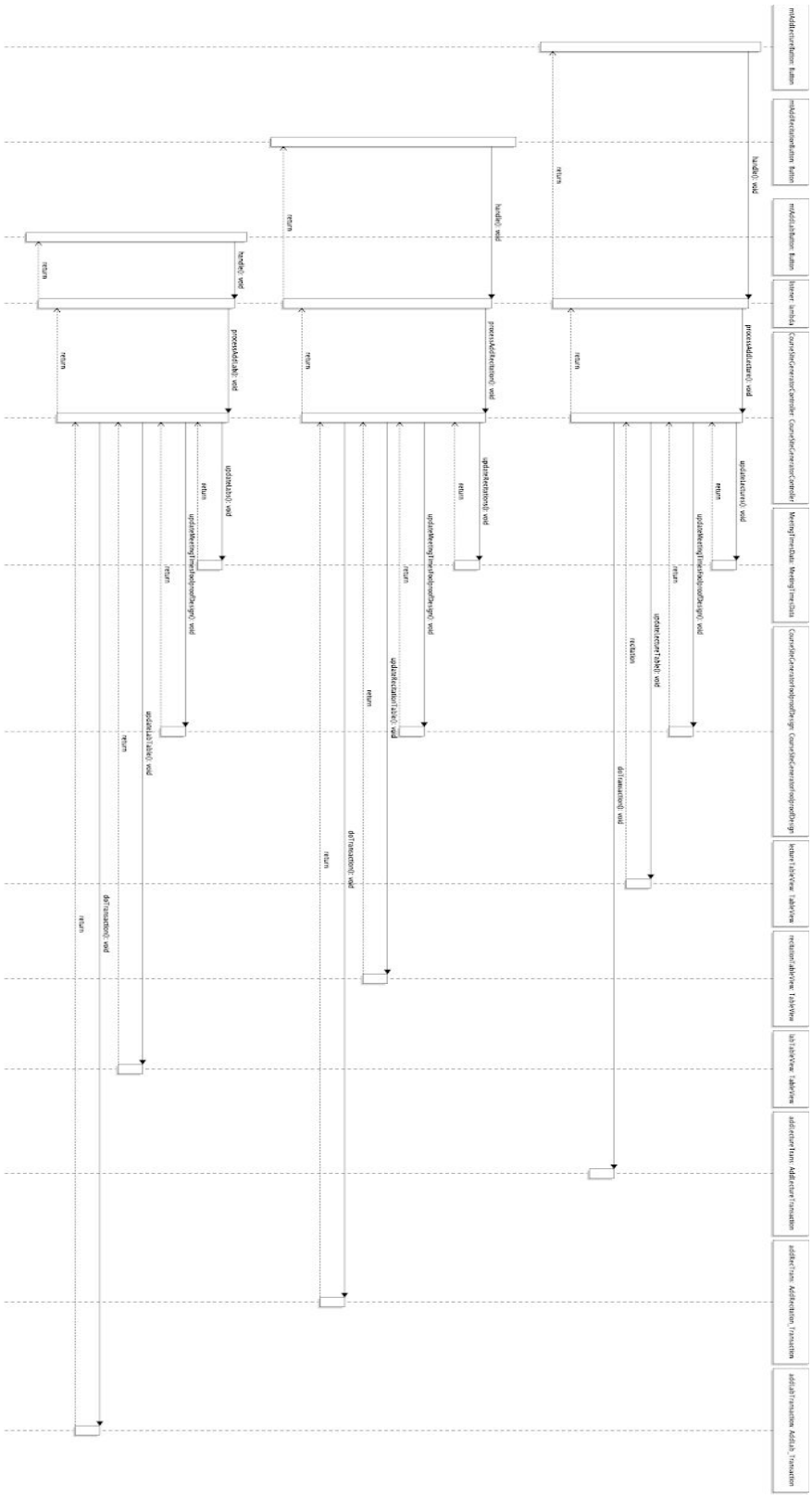

Figure 4.4 Add Lecture/Recitation/Lab UML Sequence Diagrams

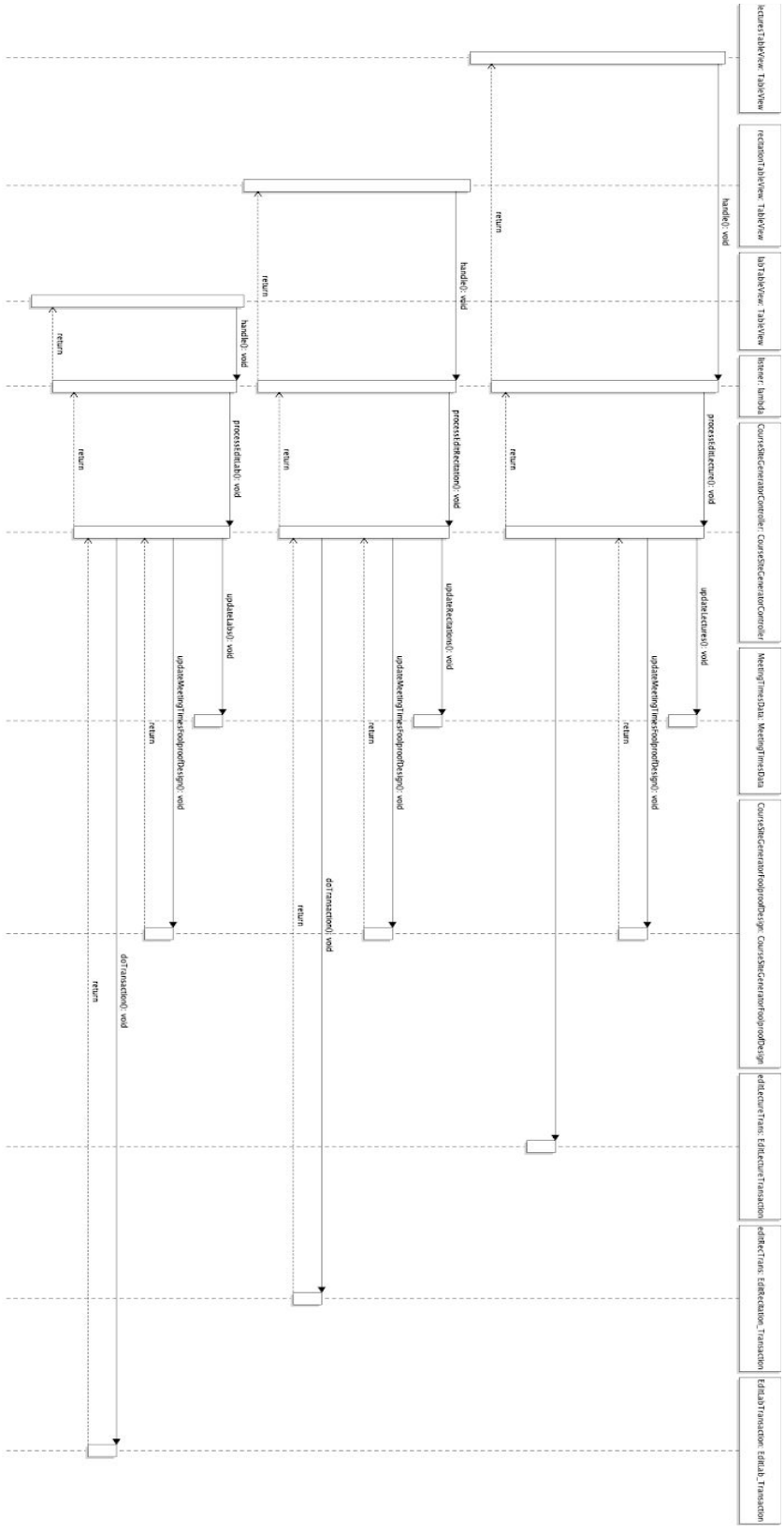

Figure 4.5 Edit Lecture/Recitation/Lab UML Sequence Diagrams

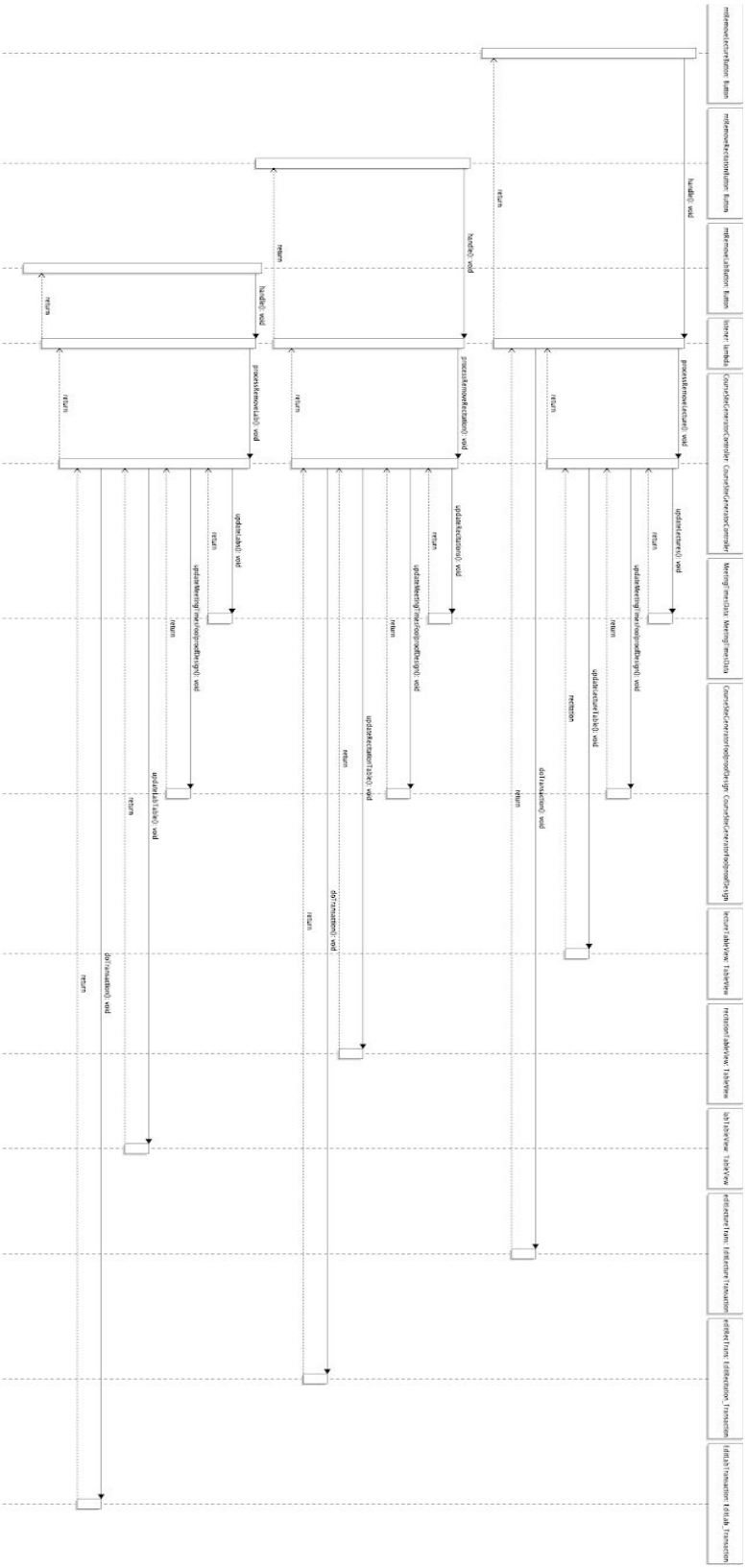

Figure 4.6 Remove Lecture/Recitation/Lab UML Sequence Diagrams

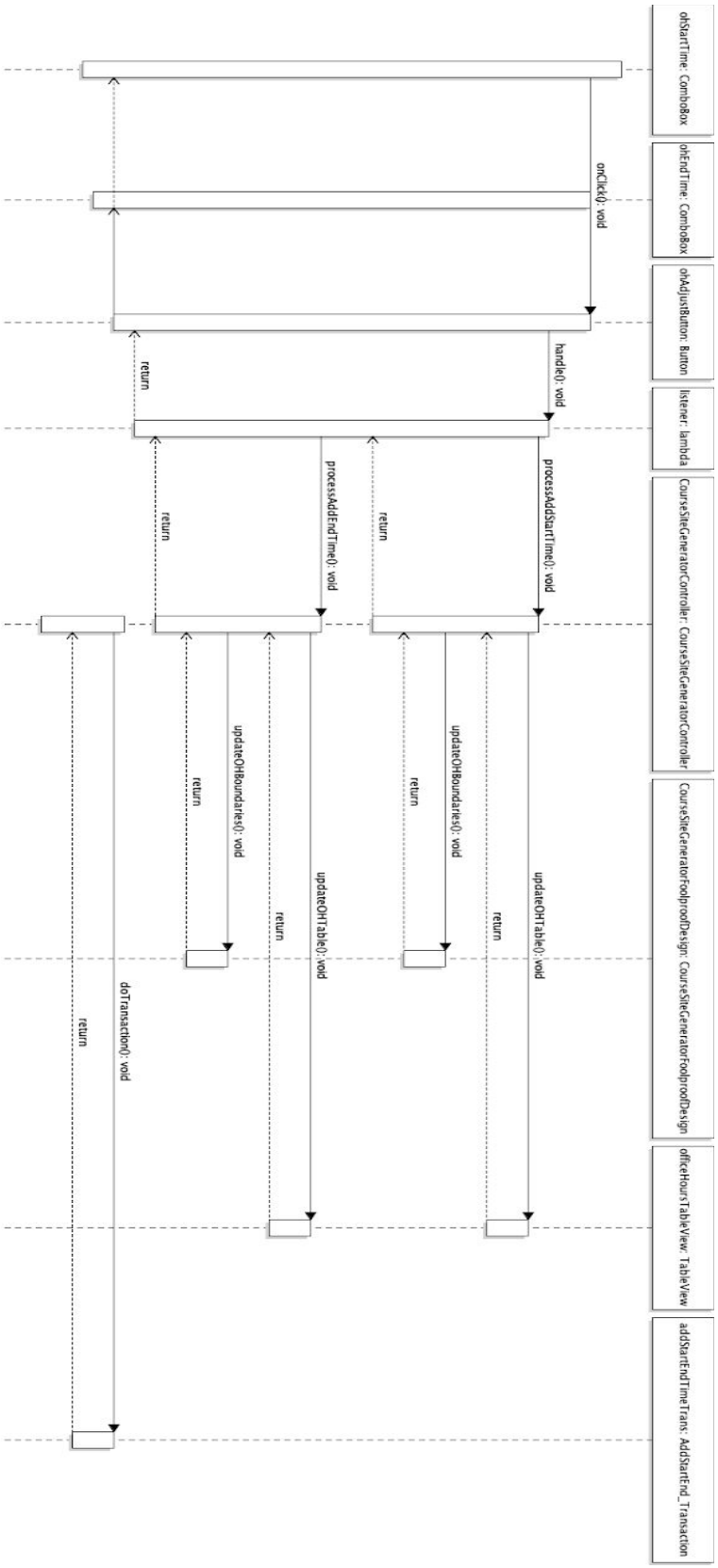

Figure 4.7 Select Time Range UML Sequence Diagrams

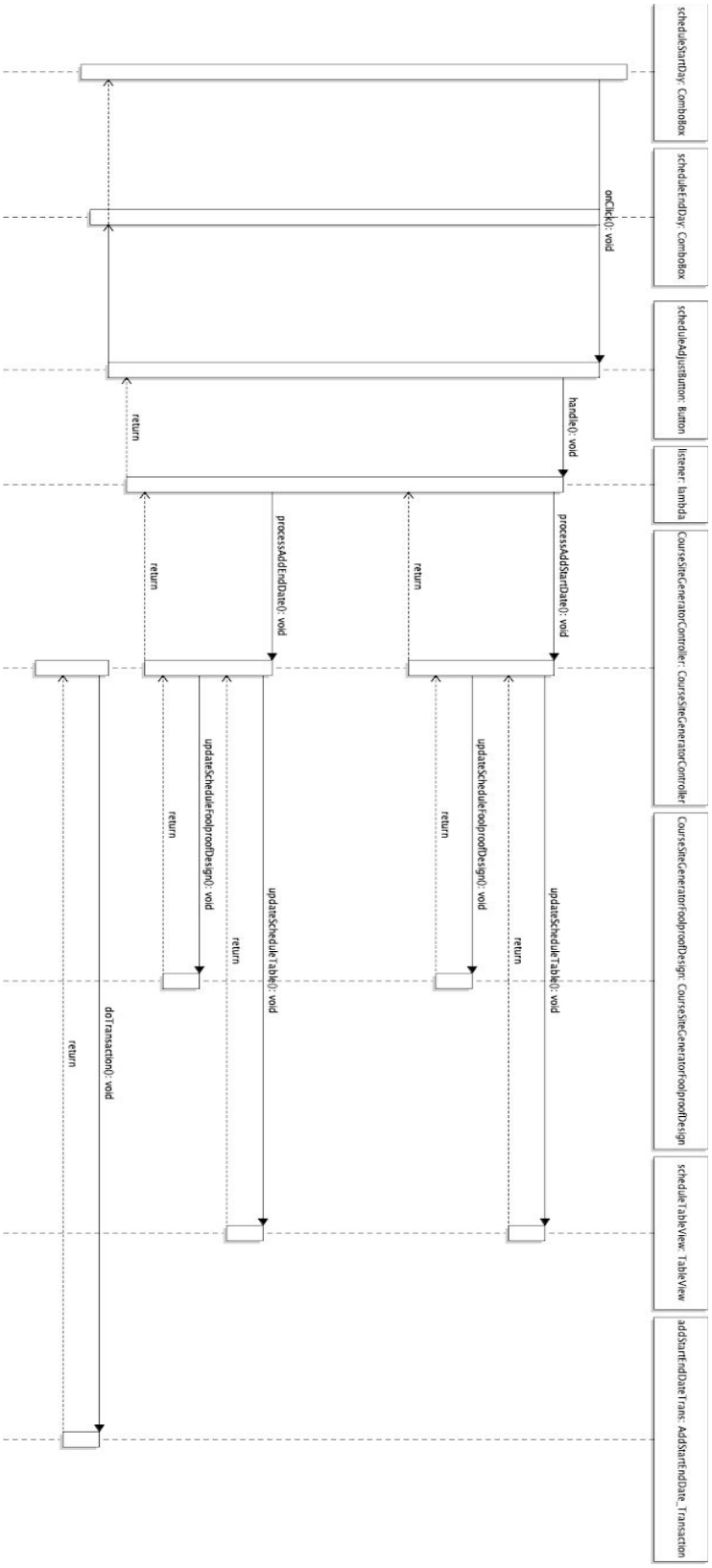

Figure 4.8 Edit Start and End Date UML Sequence Diagrams

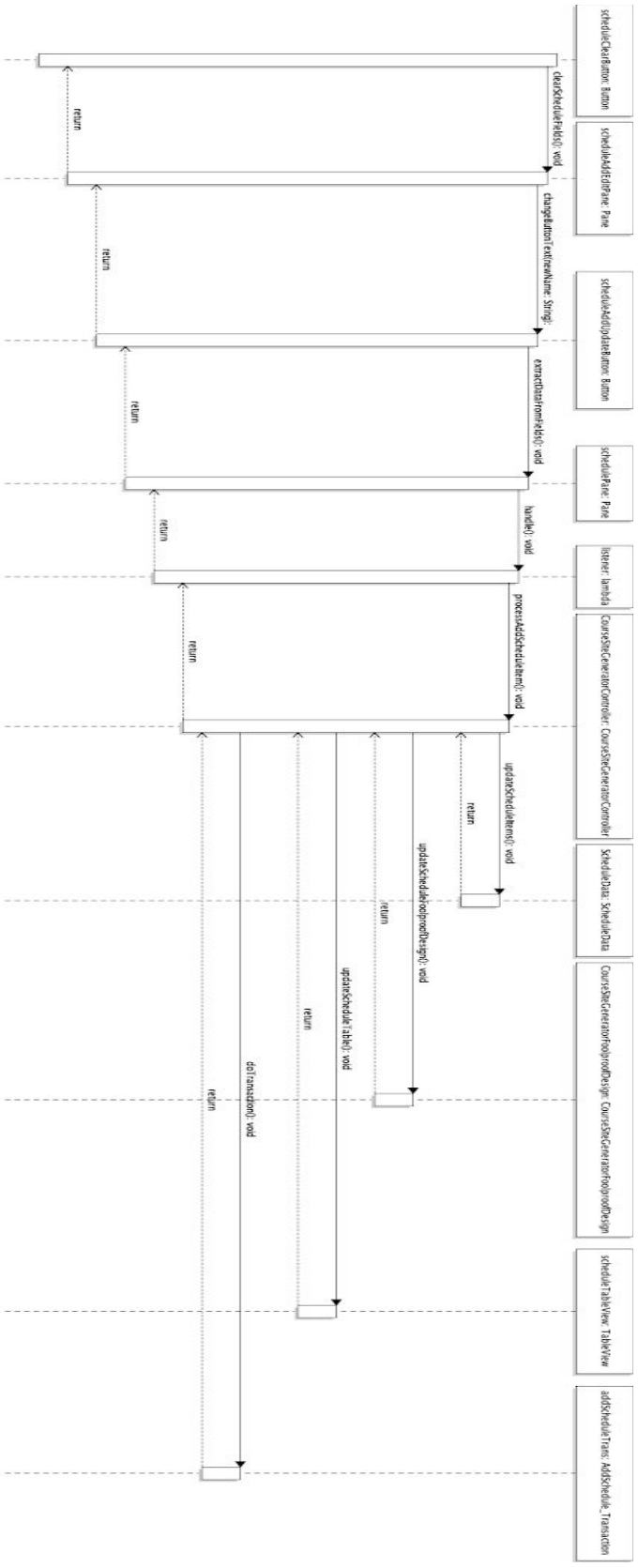

Figure 4.9 Add Schedule Item UML Sequence Diagrams

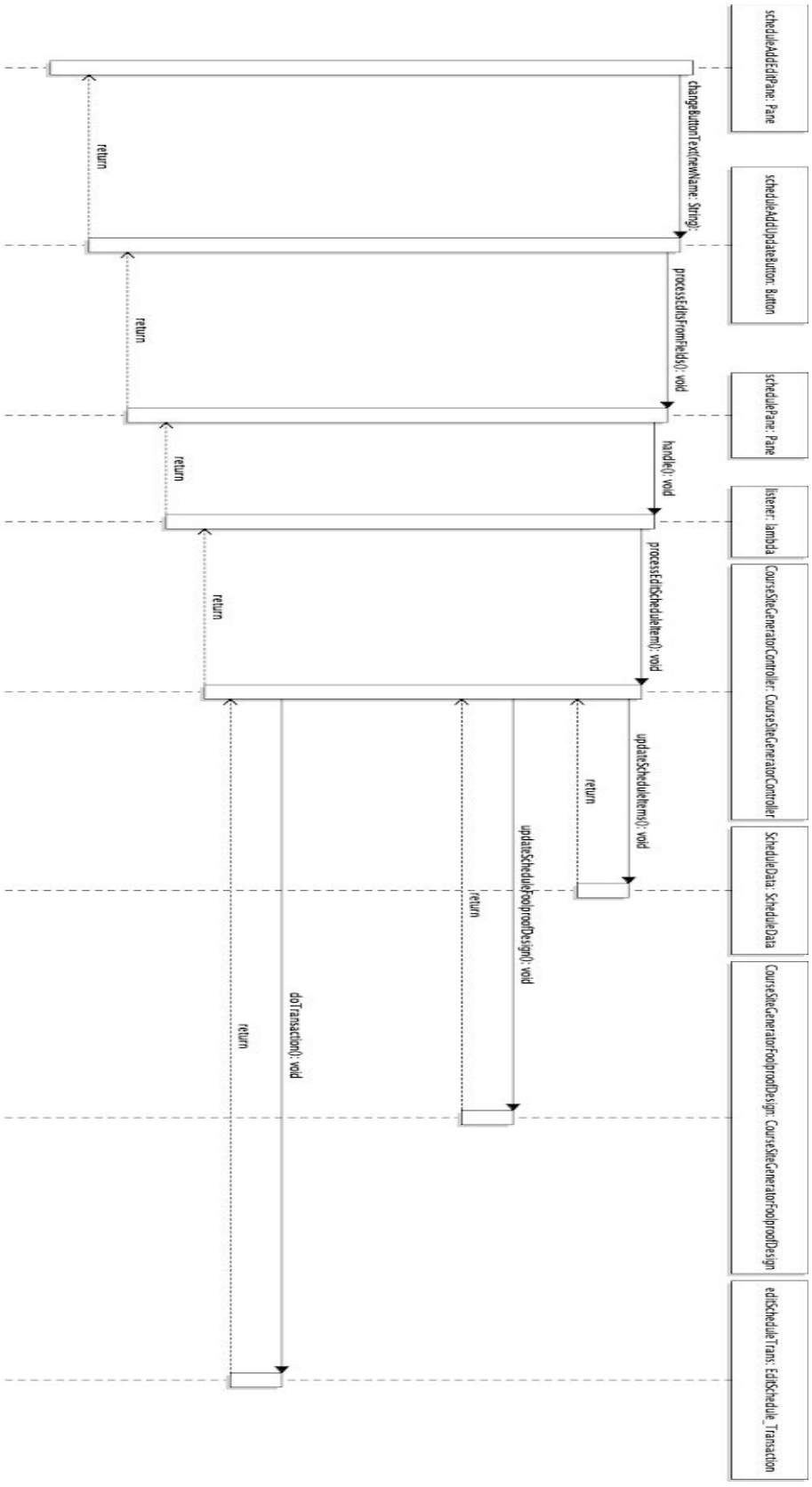

Figure 4.10 Edit Schedule Item UML Sequence Diagrams

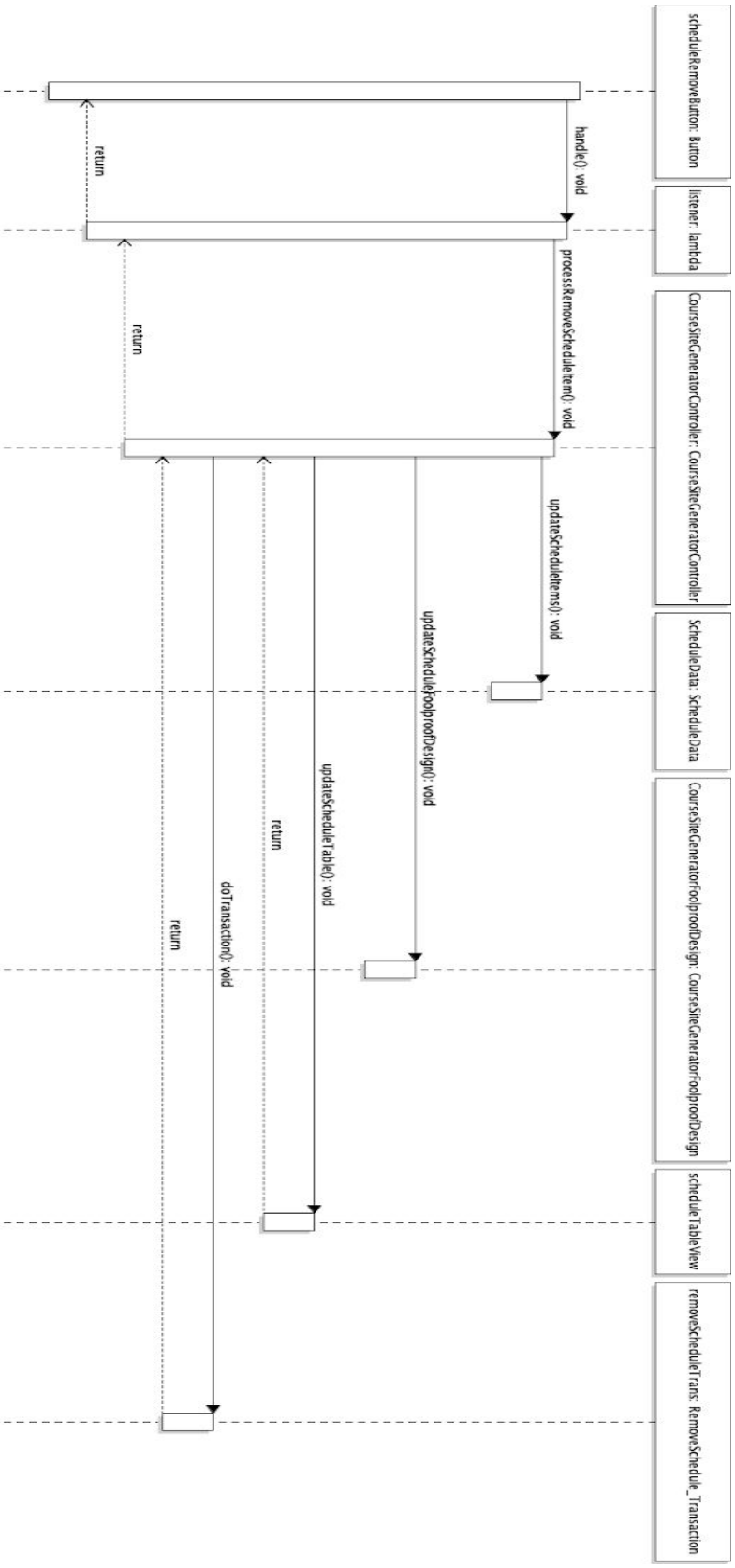

Figure 4.11 Remove Schedule Item UML Sequence Diagrams

5 File Structure and Formats

Note that all necessary data and art files must accompany this program. Figure 5.1 specifies the necessary file structure the launched application should use. Note that all necessary images should of course go in the image directory.

- App
  - CourseSiteGenerator
    - Data
    - Files
    - Transactions
    - Workspace
- Export
  - html
  - css
  - js
  - images
- Frameworks
  - DesktopJavaFramework
  - jTPS
  - PropertiesManager

Figure 5.1 Course Site Generator File Structure

5.1 Course Site Generator JSON File Structure

PageData.json

```
{
   "subject": "CSE",
   "number": "219",
   "semester": "Fall",
   "year": "2018",
   "title": "Computer Science III",
   "logos": {
      "favicon": {
         "href": "./images/SBUShieldFavicon.ico"
      },
      "navbar": {
```

```
        "href": "http://www.stonybrook.edu",
        "src": "./images/SBUDarkRedShieldLogo.png"
    },
    "bottom_left": {
        "href": "http://www.cs.stonybrook.edu",
        "src": "./images/SBUWhiteShieldLogo.jpg"
    },
    "bottom_right": {
        "href": "http://www.cs.stonybrook.edu",
        "src": "./images/SBUCSLogo.png"
    }
},
"instructor": {
    "name":     "Richard McKenna",
    "link":     "http://www.cs.stonybrook.edu/~richard",
    "email":    "richard@cs.stonybrook.edu",
    "room":     "New CS 216",
    "photo":    "./images/RichardMcKenna.jpg",
    "hours":    [
        { "day": "Tuesday",     "time": "1:00pm-2:20pm"    },
        { "day": "Wednesday",   "time": "2pm-3pm"       },
        { "day": "Thursday",      "time": "1:00pm-2:20pm"    }
    ]
},
"pages": [
    {
        "name": "Home",
        "link": "index.html"
    },
    {
        "name": "Syllabus",
        "link": "syllabus.html"
    },
    {
        "name": "Schedule",
        "link": "schedule.html"
    },
    {
        "name": "HWs",
```

```
        "link": "hws.html"
      }
   ]
}
```


This can be described as:

- Subject: the three letter code of the course
- Number: the three number code of course
- Semester: the semester of the course
- Title: the name of the course
- Logos: an array of images for the website
- Instructor: a description of the instructor
    - Name: name of the course
    - Link: link to the instructor's email
    - Email: email of the instructor
    - Room: office hours room of the instructor
    - Photo: an image of the instructor
    - Hours: office hours of the instructor
- Pages: the subsections of the course website, exported if selected
    - Name: name of the page
    - Link: link to the page

SyllabusData.json


```
{
   "description": "Development of the basic concepts and techniques from Computer Science I and II into practical programming skills that include a systematic approach to program design, coding, testing, and debugging. Application of these skills to the construction of robust programs of thousands of lines of source code. Use of programming environments and tools to aid in the software development process.",
   "topics": [
      "Programming style and its impact on readability, reliability, maintainability, and portability.",
      "Decomposing problems into modular designs with simple, narrow interfaces.",
      "Determining the proper objects in an object-oriented design.",
      "Selecting appropriate algorithms and data structures.",
      "Reusing code, including external libraries designed and built by others.",
      "Learning systematic testing and debugging techniques.",
      "Maintaining a repository of code during incremental development of a software
```

project.",
"Learning how to use threads to synchronize several tasks.",
"Improving program performance.",
"Making effective use of a programming environment, including:
Syntax-directed editor
Build tools
Debugging tools
Testing tools
Source code management tools
Profiling tools
"
],
"prerequisites": "You must have taken CSE 214 and received a grade of 'C' or better in order to take this course. In more detail, you are expected to have the following knowledge and skills at the beginning of the course:
Ability to write programs of a few hundred lines of code in the Java programming language.
Understanding of fundamental data structures, including lists, binary trees, hash tables, and graphs, and the ability to employ these data structures in the form provided by the standard Java API.
Ability to construct simple command-based user interfaces, and to use files for the input and output of data.
Mastery of basic mathematical and geometric reasoning using pre-calculus concepts.
",
"outcomes": [
"Ability to systematically design, code, debug, and test programs of about two thousand lines of code.",
"Sensitivity to the issues of programming style and modularity and their relationship to the construction and evolution of robust software.",
"Knowledge of basic ideas and techniques of object-oriented programming.",
"Familiarity with the capabilities and use of programming tools such as syntax-directed editors, debuggers, execution profilers, documentation generators, and revison-control systems."
],
"textbooks": [
{
"title": "Head First Object Oriented Analysis and Design",
"link": "http://proquestcombo.safaribooksonline.com.proxy.library.stonybrook.edu/0596008678",
"photo": "./images/HeadFirstOOAAD.jpg",
"authors": [
"Brett McLaughlin", "Gary Pollice", "David West"
],
"publisher": "O'Reilly Media, Inc.",

```
            "year": "2006"
        },
        {
            "title": "Head First Design Patterns",
            "link":
"http://proquestcombo.safaribooksonline.com.proxy.library.stonybrook.edu/0596007124
",
            "photo": "./images/HeadFirstDesignPatterns.gif",
            "authors": [
                "Eric T Freeman", "Elisabeth Robson", "Bert Bates", "Kathy Sierra"
            ],
            "publisher": "O'Reilly Media, Inc.",
            "year": "2004"
        }
    ],
    "gradedComponents": [
        {
            "name": "Recitations",
            "description": "Students will attend weekly recitations that will introduce use of
essential development tools and will require completion of an exercise for submission.",
            "weight": "10"
        },
        {
            "name": "Homework Assignments",
            "description": "The assignments will develop a students ability to design and
implement object-oriented systems. Grading will be based on functionality and proper
use specific tools. Submitted code that does not compile will receive no credit. Late
submissions will NOT be accepted. Programming assignments will be handed in
electronically, instructions for which will be provided early in the semester.",
            "weight": "20"
        },
        {
            "name": "Final Project",
            "description": "The assignments will build to the final project, which will be a fully
functioning application.",
            "weight": "20"
        },
        {
            "name": "Midterm Exam",
```

```
            "description": "The midterm will cover all lecture, quizzes, and homework materials covered during the first 1/2 of the semester.",
            "weight": "20"
        },
        {
            "name": "Final Exam",
            "description": "The final will be cumulative and will cover all lecture, reading, and homework material.",
            "weight": "30"
        }
    ],
    "gradingNote": "<strong>Note CEAS Policy:</strong> The Pass/No Credit (P/NC) option is not available for this course.",
    "academicDishonesty": "You may <em>discuss</em> the homework in this course with anyone you like, however each student's submissionmust be ones own work.<br /><br />The College of Engineering and Applied Sciences regards academic dishonesty as a very serious matter, and provides for substantial penalties in such cases. For more information, obtain a copy of the CEAS guidelines on academic dishonesty from the CEAS office.<br /><br /><strong>Be advised that any evidence of academic dishonesty will be treated with utmost seriousness. If you have a situation that may tempt you into doing something academically dishonest, resist the urge and speak with your instructor during office hours for help.</strong><br /><br />",
    "specialAssistance": "If you have a physical, psychological, medical or learning disability that may impact on your ability to carry out assigned course work, I would urge that you contact the staff at the <a href='https://www.stonybrook.edu/dss/'>Student Accessibility Support Center</a> (SASC) in the ECC building, 632-6748. SASC will review your concerns and determine with you what accommodations are necessary and appropriate. All information and documentation of disability are confidential.<br /><br /><br /><br />"
}
```


This can be described as:

- Description: a message describing what the course is
- Topics: the topics addressed in the course
- Prerequisites: described previous courses needed to take this course
- Outcomes: the goals described for the course
- Textbooks: an array of textbooks needed for the course
    - Title: The name of the textbook
    - Link: link to buy the textbook

        - Photo: image of the textbook
        - Authors: array of authors of the textbook
        - Publisher: publisher of the textbook
        - Year: the year the book was published
    - Graded Components: An array of syllabus percentages for the course
        - Name: name of the syllabus category
        - Description: description of the syllabus category
        - Weight: percentage of total grade
    - Grading Note: the grading note for the course
    - Academic Dishonesty: academic dishonesty statement for the course
    - Special Assistance: special assistance note for the course

SectionData.json


```
{
   "lectures": [
      {
         "section":  "L01",
         "days":     "Tuesdays &amp;amp; Thursdays",
         "time":     "10:00am-11:20pm",
         "room":     "Humanities 1006"
      }
   ],
   "labs": [

   ],
   "recitations": [
      {
         "section": "<strong>R06</strong> (McKenna)",
         "day_time": "Mondays, 10:00am-10:53am",
         "location": "New CS 115",
         "ta_1": "Lei Song",
         "ta_2": "Qihong Jiang"
      },
      {
         "section": "<strong>R07</strong> (McKenna)",
         "day_time": "Wednesdays, 10:00am-10:53am",
         "location": "New CS 115",
```

```
            "ta_1": "Mankirat Gulati",
            "ta_2": "Sammi Wu Leung"
        },
        {
            "section": "<strong>R08</strong> (McKenna)",
            "day_time": "Wednesdays, 2:30pm-3:23pm",
            "location": "New CS 115",
            "ta_1": "Sammi Wu Leung",
            "ta_2": "Qihong Jiang"
        }
    ]
}
```


This can be described as:

- Lectures: an array of course meeting times
    - Section: section of letter and number of the course
    - Days: days the course take place
    - Time: time the course takes place
    - Room: the place the course takes place
- Labs: an array of lab meeting times
    - Section: section of letter and number of the course
    - Day and Time: days the course take place and time the course takes place
    - Location: the place the course takes place
    - TA1: a TA for the course
    - TA2: a TA for the course
- Recitations: an array of recitation meeting times
    - Section: section of letter and number of the course
    - Day and Time: days the course take place and time the course takes place
    - Location: the place the course takes place
    - TA1: a TA for the course
    - TA2: a TA for the course

OfficeHoursData.json


```
{
    "startHour": "8",
    "endHour": "23",
```

```
  "instructor": {
    "name":     "Richard McKenna",
    "link":     "http://www.cs.stonybrook.edu/~richard",
    "email":    "richard@cs.stonybrook.edu",
    "room":     "New CS 216",
    "photo":    "./images/RichardMcKenna.jpg",
    "hours":    [
      { "day": "Tuesday",     "time": "1:00pm-2:20pm"    },
      { "day": "Wednesday",   "time": "2pm-3pm"       },
      { "day": "Thursday",      "time": "1:00pm-2:20pm"    }
    ]
  },
  "grad_tas":[
    {
      "name":"Aarin Chase",
      "email":"Aarin.Chase@stonybrook.edu"
    }
  ],
  "undergrad_tas":[
    {
      "name":"Abraham Rosloff",
      "email":"abraham.rosloff@stonybrook.edu"
    }
  ],
  "officeHours": [

    {
      "time":"3_30pm",
      "day":"WEDNESDAY",
      "name":"Abraham Rosloff"
    }
  ]
}
```

This can be described as:

- Start Hour: the earliest time office hours can occur
- End Hour: the latest time office hours can occur
- Instructor: The instructor details
    - The name of the instructor

        - Link: a url to the instructor's webpage
        - Email: the email address for the instructor
        - Room: the location of the instructor's office hours
        - Photo: an image of the instructor
        - Hours: an array of meeting times that the instructor is free
            - Day: the day of week it takes place
            - Time: the time that it takes place
    - GradTAs: an array of graduate TAs for the course
        - Name: name of the TA
        - Email: the email of the TA
    - UndergradTAs: an array of undergraduate TAs for the course
        - Name: name of the TA
        - Email: the email of the TA
    - OfficeHours: an array of office hours for each TA
        - Time: the time of the oh slot
        - Day: day of the oh slot
        - Name: name of the TA that occupies that slot

ScheduleData.json


```
{
   "startingMondayMonth":  "8",
   "startingMondayDay":    "27",
   "endingFridayMonth":    "12",
   "endingFridayDay":      "21",

   "holidays": [
      { "month": "9", "day": "3", "title": "LABOR DAY", "link":
"https://www.stonybrook.edu/commcms/registrar/calendars/academic_calendars" },
      { "month": "10", "day": "8", "title": "FALL BREAK?<br /><br /><br />", "link":
"https://www.stonybrook.edu/commcms/registrar/calendars/academic_calendars" },
      { "month": "10", "day": "9", "title": "FALL BREAK?", "link":
"https://www.stonybrook.edu/commcms/registrar/calendars/academic_calendars" },
      { "month": "11", "day": "21", "title": "TRAVEL DAY", "link":
"https://www.stonybrook.edu/commcms/registrar/calendars/academic_calendars" },
      { "month": "11", "day": "22", "title": "THANKSGIVING", "link":
"https://www.stonybrook.edu/commcms/registrar/calendars/academic_calendars" },
      { "month": "11", "day": "23", "title": "BLACK FRIDAY", "link":
"https://www.stonybrook.edu/commcms/registrar/calendars/academic_calendars" },
```

```
        { "month": "12", "day": "11", "title": "READING DAY", "link":
"https://www.stonybrook.edu/commcms/registrar/calendars/academic_calendars" },
        { "month": "12", "day": "12", "title": "", "link": "" },
        { "month": "12", "day": "13", "title": "", "link": "" },
        { "month": "12", "day": "14", "title": "", "link": "" },
        { "month": "12", "day": "17", "title": "", "link": "" },
        { "month": "12", "day": "18", "title": "", "link": "" },
        { "month": "12", "day": "19", "title": "", "link": "" },
        { "month": "12", "day": "20", "title": "", "link": "" },
        { "month": "12", "day": "21", "title": "", "link": "" }
    ],

    "lectures":[
        { "month": "8",     "day": "28",     "title": "Lecture 1",      "topic": "Software<br
/>Development<br />Lifecycle<br /><br />",    "link":
"https://blackboard.stonybrook.edu/bbcswebdav/pid-4583144-dt-content-rid-31807868_
1/xid-31807868_1" },
        { "month": "8",     "day": "30",     "title": "Lecture 2",      "topic": "Graphical<br
/>User<br />Interfaces<br /><br />",    "link":
"https://blackboard.stonybrook.edu/bbcswebdav/pid-4588421-dt-content-rid-31820795_
1/xid-31820795_1" },
        { "month": "9",     "day": "4",     "title": "Lecture 3",      "topic": "Event<br
/>Programming<br /><br />",    "link":
"https://blackboard.stonybrook.edu/bbcswebdav/pid-4591754-dt-content-rid-31835006_
1/xid-31835006_1" },
        { "month": "9",     "day": "6",     "title": "Lecture 4",      "topic": "Graphics<br />&amp;
GUIs<br /><br />",    "link":
"https://blackboard.stonybrook.edu/bbcswebdav/pid-4595769-dt-content-rid-31868074_
1/xid-31868074_1" },
        { "month": "9",     "day": "11",     "title": "Lecture 5",      "topic": "Threads<br /><br
/>",    "link":
"https://blackboard.stonybrook.edu/bbcswebdav/pid-4598618-dt-content-rid-31897815_
1/xid-31897815_1" },
        { "month": "9",     "day": "13",     "title": "Lecture 6",      "topic": "Threads<br
/>(continued)<br /><br />",    "link": "none" },
        { "month": "9",     "day": "18",     "title": "Lecture 7",      "topic": "Multithreading<br
/>Issues<br /><br /><br />",    "link":
"https://blackboard.stonybrook.edu/bbcswebdav/pid-4598619-dt-content-rid-31897816_
1/xid-31897816_1" },
```

{ "month": "9", "day": "20", "title": "Lecture 8", "topic": "Multithreading
(continued)

", "link": "none" },
{ "month": "9", "day": "25", "title": "Lecture 9", "topic": "Properties of
High Quality Software

", "link": "https://blackboard.stonybrook.edu/bbcswebdav/pid-4611824-dt-content-rid-32139187_1/xid-32139187_1" },
{ "month": "9", "day": "27", "title": "Lecture 10", "topic": "OOP++

", "link": "https://blackboard.stonybrook.edu/bbcswebdav/pid-4612756-dt-content-rid-32163006_1/xid-32163006_1" },
{ "month": "10", "day": "2", "title": "Lecture 11", "topic": "OOP++
(continued)

", "link": "none" },
{ "month": "10", "day": "4", "title": "Lecture 13", "topic": "OOP++
(continued)

", "link": "none" },
{ "month": "10", "day": "11", "title": "Lecture 13", "topic": "Object-Oriented Design
using UML

", "link": "https://blackboard.stonybrook.edu/bbcswebdav/pid-4618222-dt-content-rid-32348762_1/xid-32348762_1" },
{ "month": "10", "day": "16", "title": "Midterm Exam", "topic": "(Version 1 Solutions)
(Version 2 Solutions)

", "link": "none" },
{ "month": "10", "day": "18", "title": "Lecture 15", "topic": "Object-Oriented Design
using UML
(continued)
", "link": "none" },
{ "month": "10", "day": "23", "title": "Lecture 16", "topic": "Design Review

", "link": "https://blackboard.stonybrook.edu/bbcswebdav/pid-4626374-dt-content-rid-32609229_1/xid-32609229_1" },
{ "month": "10", "day": "25", "title": "Lecture 17", "topic": "Test-Driven
Development

", "link": "https://blackboard.stonybrook.edu/bbcswebdav/pid-4626392-dt-content-rid-32609832_1/xid-32609832_1" },
{ "month": "10", "day": "30", "title": "Lecture 18", "topic": "Designing
with Exceptions", "link": "https://blackboard.stonybrook.edu/bbcswebdav/pid-4626375-dt-content-rid-32609231_1/xid-32609231_1" },
{ "month": "11", "day": "1", "title": "Lecture 19", "topic": "Test-Driven
Development
(continued)

", "link": "none" },

```
        { "month": "11",     "day": "6",     "title": "Lecture 20",      "topic": "Creational<br
/>Design Patterns<br /><br />",    "link": "none" },
        { "month": "11",     "day": "8",     "title": "Lecture 21",      "topic": "Structural<br
/>Design Patterns<br /><br />",    "link": "none" },
        { "month": "11",     "day": "13",     "title": "Lecture 22",      "topic": "Behavioral<br
/>Design Patterns<br /><br /><br />",    "link": "none" },
        { "month": "11",     "day": "15",     "title": "Lecture 23",      "topic": "Behavioral<br
/>Design Patterns<br />(continued)<br /><br />",    "link": "none" },
        { "month": "11",     "day": "20",     "title": "NO LECTURE",      "topic": "(Project Work
Day)<br /><br />",    "link": "none" },
        { "month": "11",     "day": "27",     "title": "Lecture 25",      "topic": "Code Profiling<br
/><br /><br />",    "link": "none" },
        { "month": "11",     "day": "29",     "title": "Lecture 26",      "topic": "Optimization<br
/>Techniques<br /><br />",    "link": "none" },
        { "month": "12",     "day": "4",     "title": "Guest Lecture",      "topic": "TBA<br /><br
/>",    "link": "none" },
        { "month": "12",     "day": "6",     "title": "Lecture 28",      "topic": "Final<br
/>Review<br /><br />",    "link": "none" },
        { "month": "12",    "day": "20",    "title": "<a
href='https://www.stonybrook.edu/commcms/registrar/registration/_exams/fall18-finals.p
hp'>FINAL EXAM</a>",  "topic": "(8:00am-10:45am)<br />(Cumulative)<br /><br />",
"link": "none" }
    ],

    "references": [
        { "month": "8", "day": "30", "title": "Reference", "topic": "JavaFX Examples", "link":
"https://blackboard.stonybrook.edu/bbcswebdav/pid-4588427-dt-content-rid-31820796_
1/xid-31820796_1" },
        {
            "month": "9", "day": "11", "title": "Reference", "topic": "Progress Example", "link":
"https://blackboard.stonybrook.edu/bbcswebdav/pid-4598620-dt-content-rid-31897817_
1/xid-31897817_1"
        },
        { "month": "10", "day": "26", "title": "Drop Deadline", "topic": "Last Day to Drop<br />
with a 'W'", "link":
"https://www.stonybrook.edu/commcms/registrar/calendars/academic_calendars" },
        { "month": "10", "day": "15", "title": "Midterm Review", "topic": "5pm, Old CS
2311<br /><br />", "link": "none" }
    ],
```

```
   "recitations":[
            { "month": "8",     "day": "27",     "title": "NO RECITATION",      "topic": "<br
/><br />",    "link": "none" },
       { "month": "8",     "day": "29",     "title": "NO RECITATION",      "topic": "<br /><br
/>",    "link": "none" },
       { "month": "9",     "day": "5",     "title": "Recitation 1",      "topic": "Debugging<br
/><br />",    "link": "none" },
       { "month": "9",     "day": "10",     "title": "Recitation 1",      "topic": "Debugging<br
/><br />",    "link": "none" },
       { "month": "9",     "day": "12",     "title": "Recitation 2",      "topic": "Version
Control<br /><br />",    "link": "none" },
       { "month": "9",     "day": "17",     "title": "Recitation 2",      "topic": "Version
Control<br /><br />",    "link": "none" },
       { "month": "9",     "day": "19",     "title": "Recitation 3",      "topic": "Canvas<br /><br
/>",    "link": "none" },
       { "month": "9",     "day": "24",     "title": "Recitation 3",      "topic": "Canvas<br /><br
/>",    "link": "none" },
       { "month": "9",     "day": "26",     "title": "Recitation 4",      "topic": "Front-End<br
/>Web Technologies<br /><br /><br />",    "link": "none" },
       { "month": "10",     "day": "1",     "title": "Recitation 4",      "topic": "Front-End<br
/>Web Technologies<br /><br /><br />",    "link": "none" },
       { "month": "10",     "day": "3",     "title": "Recitation 5",      "topic": "Threads<br /><br
/><br />",    "link": "none" },
       { "month": "10",     "day": "10",     "title": "NO RECITATION",      "topic": "<br />",
"link": "none" },
       { "month": "10",     "day": "15",     "title": "Recitation 5",      "topic": "Threads<br /><br
/>",    "link": "none" },
       { "month": "10",     "day": "17",     "title": "Recitation 6",      "topic": "UML<br
/>Diagrams<br /><br />",    "link": "none" },
       { "month": "10",     "day": "22",     "title": "Recitation 6",      "topic": "UML<br
/>Diagrams<br /><br />",    "link": "none" },
       { "month": "10",     "day": "24",     "title": "NO RECITATION",      "topic": "<br /><br
/>",    "link": "none" },
       { "month": "10",     "day": "29",     "title": "Recitation 7",      "topic": "Unit Testing<br
/><br />",    "link": "none" },
       { "month": "10",     "day": "31",     "title": "Recitation 7",      "topic": "Unit Testing<br
/><br />",    "link": "none" },
       { "month": "11",     "day": "5",     "title": "Recitation 8",      "topic": "Build Tools<br
```

```
/><br />",    "link": "none" },
      { "month": "11",     "day": "7",     "title": "Recitation 8",      "topic": "Build Tools<br
/><br />",    "link": "none" },
      { "month": "11",     "day": "12",     "title": "Recitation 9",      "topic": "Unit Testing<br
/><br />",    "link": "none" },
      { "month": "11",     "day": "14",     "title": "Recitation 9",      "topic": "Unit Testing<br
/><br />",    "link": "none" },
      { "month": "11",     "day": "19",     "title": "NO RECITATION",      "topic": "<br /><br
/>",    "link": "none" },
      { "month": "11",     "day": "26",     "title": "Recitation 10",      "topic": "Mock-Ups<br
/><br />",    "link": "none" },
      { "month": "11",     "day": "28",     "title": "Recitation 10",      "topic": "Mock-Ups<br
/><br />",    "link": "none" },
      { "month": "12",     "day": "3",     "title": "Recitation 11",      "topic": "Code Profiling<br
/><br />",    "link": "none" },
      { "month": "12",     "day": "5",     "title": "Recitation 11",      "topic": "Code Profiling<br
/><br />",    "link": "none" },
      { "month": "12",     "day": "10",     "title": "NO RECITATION",      "topic": "<br /><br
/>",    "link": "none" }

   ],
   "hws": [
      { "month": "9", "day": "21",    "title": "HW 1",    "topic": "due @ 11:59pm<br />(GUIs
&amp;amp; Events)",      "link": "./hw/HW1.html",     "time": "",  "criteria": "none"     },
      { "month": "10", "day": "10",     "title": "HW 2",    "topic": "due @ 11:59pm<br
/>(GUIs &amp;amp; Events)",      "link": "./hw/HW2.html",     "time": "",  "criteria": "none"     },
      { "month": "10", "day": "26",    "title": "HW 3",    "topic": "due @ 11:59pm<br />(UML
Design)",      "link": "./hw/HW3.html",     "time": "",  "criteria": "none"     },
      { "month": "11", "day": "7",    "title": "HW 4",    "topic": "due @ 11:59pm<br
/>Implementation<br />Benchmark I",      "link": "none",     "time": "",  "criteria": "none"
},
      { "month": "11", "day": "20",    "title": "HW 5",    "topic": "due @ 11:59pm<br
/>Implementation<br />Benchmark II",      "link": "none",     "time": "",  "criteria": "none"
},
      { "month": "12", "day": "10",    "title": "Final Project",    "topic": "due @ 11:59pm<br
/>(JavaFX App)",      "link": "none",     "time": "",  "criteria": "none"     }
   ]
}
```

This can be described as:

- StartingMondayMonth: the starting month of the course
- StartingMondayDay: the starting day of the course that falls on a monday
- EndingFridayMonth: the ending month of the course
- EndingFridayDay: the ending day of the course that falls on a friday
- Holidays: an array of holidays for the course
    - Month: the month of the holiday
    - Day: the day of the holiday
    - Title: the name of the Holiday
    - Link: a link to a calendar
- Lectures: an array of lecture categories for the course
    - Month: the month of the lecture
    - Day: the day of the lecture
    - Title: the name of the lecture
    - Topic: what the lecture is about
    - Link: a link to a lecture
- References: an array of references for the course
    - Month: the month of the reference
    - Day: the day of the reference
    - Title: the name of the reference
    - Topic: what the reference is about
    - Link: a link to a reference
- Recitations: an array of recitations for the course
    - Month: the month of the recitation
    - Day: the day of the recitation
    - Title: the name of the recitation
    - Topic: what the recitation is about
    - Link: a link to a recitation
- HWs: an array of homeworks for the course
    - Month: the month of the homework
    - Day: the day of the homework
    - Title: the name of the homework
    - Topic: what the homework is about
    - Link: a link to a homework

6 Supporting Information

Note that this document should serve as a reference for those implementing the code, so we'll provide a table of contents to help quickly find important sections.

6.1 Table of Contents

6.2 Appendixes

N/A