Project Plan

Cesar Lopez

Christopher Victores

David Chew

Vijay Patel

Steven Dickinson

CMSC 495, Section 6265

Submitted to: Professor Nicholas Duchon

Date: 12/11/2016

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Changes |
| Draft | Nov 5, 2016 | Everyone | Original Draft |
| 1.1 | Nov 7, 2016 | Steven Dickinson | Added User’s Guide |
| 1.2 | Nov 10, 2016 | Vijay Patel | Added Test Plan |
| 1.3 | Nov 10, 2016 | Cesar Lopez | Proof Read and Formatted. |
| 1.4 | Nov 13, 2016 | Steven Dickinson | Incorporated Peer Review comments |
| 1.5 | Nov 17, 2016 | Everyone | Added Design Plan |
| 1.6 | Nov 24, 2016 | Steven Dickinson | Updated User’s Guide |
| 1.7 | Nov 24, 2016 | David Chew | Syntax edits and formatting |
| 1.8 | Nov 27, 2016 | Cesar Lopez | Updated UML diagram |
| 1.9 | Nov 30, 2016 | Cesar Lopez | Added Phase Reports |
| 2.0 | Dec 4, 2016 | Cesar Lopez | Updated Phase 2 Report |
| 2.1 | Dec 7, 2016 | Cesar Lopez | Added Phase 3 Report |
| 2.2 | Dec 10, 2016 | Steve Dickinson | Updated User’s Guide |
| 2.3 | Dec 11, 2016 | Vijay Patel | Updated Test Plan |
| 2.4 | Dec 11, 2016 | Cesar Lopez | Merged Team Members Docs |
| 2.5 | Dec 15, 2016 | Vijay Patel | Added Chapter 4 Sections |
| 2.6 | Dec 15, 2016 | Cesar Lopez | Added Overview Chapter |
| 2.7 | Dec 15, 2016 | Steven Dickinson | Made edits to UserGuide |
|  |  |  |  |

Contents

[1. Requirements 7](#_Toc469678070)

[1.1 Product Requirement Introduction 7](#_Toc469678071)

[1.1.1 Purpose 7](#_Toc469678072)

[1.1.2 Scope 7](#_Toc469678073)

[1.1.3 Definitions 7](#_Toc469678074)

[1.1.4 References 7](#_Toc469678075)

[1.2 System Specification 7](#_Toc469678076)

[1.2.1 Software Requirements 7](#_Toc469678077)

[1.2.2 Hardware Requirements 7](#_Toc469678078)

[1.3 Project Description 8](#_Toc469678079)

[1.4 Requirement Priority Matrix 8](#_Toc469678080)

[2. Project Plan 9](#_Toc469678081)

[2.1 Work Plan 9](#_Toc469678082)

[2.1.1 Work Allocations and Activities 9](#_Toc469678083)

[2.1.2 Key Milestones 9](#_Toc469678084)

[3. User’s Guide 10](#_Toc469678085)

[3.1 HTML Tutorials 10](#_Toc469678086)

[3.2 Location 10](#_Toc469678087)

[3.3 HTML Editor Layout 10](#_Toc469678088)

[3.4 Creating a new HTML Document 11](#_Toc469678089)

[3.5 Opening a sample HTML Project 11](#_Toc469678090)

[3.6 Opening an existing HTML Project 11](#_Toc469678091)

[3.7 Selecting an existing URL 12](#_Toc469678092)

[3.8 Getting Help 12](#_Toc469678093)

[3.9 Project information 13](#_Toc469678094)

[3.10 Switching between Tab categories 14](#_Toc469678095)

[3.11 Tool tips 14](#_Toc469678096)

[3.12 Populating the editor with HTML tags 15](#_Toc469678097)

[3.13 Selecting and using attributes 15](#_Toc469678098)

[3.14 Navigating a large HTML Document. 16](#_Toc469678099)

[3.15 Using Global Attributes 16](#_Toc469678100)

[3.16 Previewing the HTML 17](#_Toc469678101)

[3.17 Exiting the HTML Editor 17](#_Toc469678102)

[4. TEST PLAN 18](#_Toc469678103)

[4.1 Introduction 18](#_Toc469678104)

[4.2 Plan Purpose 18](#_Toc469678105)

[4.3 Functional Test Plan 18](#_Toc469678106)

[4.4 Test Result Summary: 24](#_Toc469678107)

[4.4.1 Test Case-01 25](#_Toc469678108)

[4.4.2 Test Case-02 25](#_Toc469678109)

[4.4.3 Test Case-03 26](#_Toc469678110)

[4.4.4 Test Case-04 26](#_Toc469678111)

[4.4.5. Test Case-05 27](#_Toc469678112)

[4.4.6. Test Case-06 28](#_Toc469678113)

[4.4.7. Test Case-07 28](#_Toc469678114)

[4.4.8. Test Case-08 29](#_Toc469678115)

[4.4.9. Test Case-09 30](#_Toc469678116)

[4.4.10. Test Case-10 31](#_Toc469678117)

[4.4.11. Test Case-11 31](#_Toc469678118)

[4.4.12. Test Case-12 32](#_Toc469678119)

[4.4.13. Test Case-13 33](#_Toc469678120)

[4.4.14. Test Case-14 34](#_Toc469678121)

[4.4.15. Test Case-15 35](#_Toc469678122)

[4.4.16. Test Case-16 35](#_Toc469678123)

[4.2.17. Test Case-17 36](#_Toc469678124)

[4.4.18. Test Case-18 37](#_Toc469678125)

[4.4.19. Test Case-19 38](#_Toc469678126)

[4.4.20. Test Case- 20 38](#_Toc469678127)

[4.5. Test Case- Error 39](#_Toc469678128)

[4.5.1 Test Case- Error - Screen shot - 1 39](#_Toc469678129)

[4.5.2 Test Case- Error - Screen shot - 2 39](#_Toc469678130)

[4.5.3 Test Case- Error - Screen shot - 3 40](#_Toc469678131)

[4.5.4 Test Case- Error - Screen shot -4 40](#_Toc469678132)

[4.5.5 Test Case- Error - Screen shot -5 41](#_Toc469678133)

[4.6 Test Strategy 42](#_Toc469678134)

[4.6.1 Test Assumptions 42](#_Toc469678135)

[4.7 Test Plan Execution Strategy 42](#_Toc469678136)

[4.7.1 Defect tracking & Reporting 42](#_Toc469678137)

[4.8 Test Management Process 42](#_Toc469678138)

[4.8.1 Test Execution Process 42](#_Toc469678139)

[5. Design Plan 43](#_Toc469678140)

[5.1 Introduction 43](#_Toc469678141)

[5.1.1 User Interface 43](#_Toc469678142)

[5.2 Communication Diagram 43](#_Toc469678143)

[5.3 Class Diagram 44](#_Toc469678144)

[5.4 Basic User Interface 46](#_Toc469678145)

[6. Phase 1 Report 47](#_Toc469678146)

[6.1 Introduction 47](#_Toc469678147)

[6.2 Roles and Task for Phase 1 47](#_Toc469678148)

[6.3 Original and Current Schedule 47](#_Toc469678149)

[6.3.1 Original Schedule for Phase 1 47](#_Toc469678150)

[6.3.2 Current Schedule for Phase 1 48](#_Toc469678151)

[6.4 Problems Encountered 48](#_Toc469678152)

[6.5 Reevaluation of All Decision Made till Now 49](#_Toc469678153)

[6.5.1 Decision on Roles 49](#_Toc469678154)

[6.5.2 Communications 49](#_Toc469678155)

[6.5.3 Sharing Documents 50](#_Toc469678156)

[6.6 Goals for phase 2. 50](#_Toc469678157)

[6.7 Document Updates 50](#_Toc469678158)

[7. Phase 2 Report 51](#_Toc469678159)

[7.1 Introduction 51](#_Toc469678160)

[7.2 Roles and Task for Phase 2 51](#_Toc469678161)

[7.3 Original and Current Schedule 51](#_Toc469678162)

[7.3.1 Original Schedule for Phase 2 52](#_Toc469678163)

[7.3.2 Current Schedule for Phase 2 52](#_Toc469678164)

[7.4 Problems Encountered 52](#_Toc469678165)

[7.5 Reevaluations 53](#_Toc469678166)

[7.5.1 Graphical User Interface Layout 53](#_Toc469678167)

[7.5.2 Tag Description using Mouse Over Action Event 53](#_Toc469678168)

[7.5.3 Placing Phase Report in Main Documentation 53](#_Toc469678169)

[7.5.4 Updating User’s Guide and UML diagram. 54](#_Toc469678170)

[7.6 Goals for Phase 3 54](#_Toc469678171)

[7.7 Document Updates 54](#_Toc469678172)

[8. Phase 3 Report 55](#_Toc469678173)

[8.1 Introduction 55](#_Toc469678174)

[8.2 Roles and Task for Phase 3 55](#_Toc469678175)

[8.3 Original and Current Schedule 56](#_Toc469678176)

[8.3.1 Original Schedule for Phase 2 56](#_Toc469678177)

[8.3.2 Current Schedule for Phase 2 56](#_Toc469678178)

[8.4 Problems Encountered 56](#_Toc469678179)

[8.5 Reevaluations 57](#_Toc469678180)

[8.5.1 HTML Editor Final Functionalities 57](#_Toc469678181)

[8.5.2 The Finish Line/Clean Up Week 57](#_Toc469678182)

[8.5.3 Java Classes 58](#_Toc469678183)

[8.6 Goals for Final Week 58](#_Toc469678184)

[8.7 Updated Documents 58](#_Toc469678185)

[9. Overview 59](#_Toc469678186)

[9.1 Team Members’ Contributions 59](#_Toc469678187)

[9.2 Lessons Learned 60](#_Toc469678188)

[9.3 Design Alternative 63](#_Toc469678189)

[9.3.1 Java Swing 63](#_Toc469678190)

[9.3.2 CSS Inline 63](#_Toc469678191)

[9.4 Design Strengths 63](#_Toc469678192)

[9.5 Limitations 63](#_Toc469678193)

[9.5.1 Multiple HTML Files 64](#_Toc469678194)

[9.5.2 Images for Web View Feature 64](#_Toc469678195)

[9.6 Suggestions for Future improvement 64](#_Toc469678196)

[9.6.1 Cascade Style Sheets 64](#_Toc469678197)

[9.6.2 Tag Validation 64](#_Toc469678198)

# 1. **Requirements**

## 1.1 **Product Requirement Introduction**

### 1.1.1 Purpose

To provide developers and operators a GUI driven editor for HTML, *Aggravator* will develop tools that contain selectable, organized HTML tag options.

### 1.1.2 Scope

The HTML editor will support HTML5. The HTML editor allows the web application developer to select, insert desired tags, and attributes. HTML editor will have a standardized format.

### 1.1.3 Definitions

* HTML - Hypertext Markup Language
* IEEE - Institute of Electrical and Electronics Engineers
* JAR File - A package file format typically used to aggregate many Java class files and associated metadata and resources (text, images, etc.) into one file to distribute application software or libraries on the Java platform.
* JDK - Java Development Kit

### 1.1.4 References

* HTML Reference - W3Schools. (n.d.). Retrieved November 4, 2016, from http://www.w3schools.com/TAGS/default.asp
* HTML Element Reference. (n.d.). Retrieved November 04, 2016, from http://www.w3schools.com/tags/refbyfunc.asp
* IEEE 830-1998 Recommended Practice for Software Requirements Specifications. (n.d.). Retrieved November 04, 2016, from https://standards.ieee.org/findstds/standard/830-1998.html
* IEEE 1058-1998 Standard for Software Project Management Plans. (n.d.). Retrieved November 04, 2016, from http://standards.ieee.org/findstds/standard/1058-1998.html

## 1.2 System Specification

### 1.2.1 Software Requirements

* Windows 10, mac OS Sierra, Linux
* JDK 8.x

### 1.2.2 Hardware Requirements

* 1 GHz processor speed or faster
* 8 GB 375 MHz or higher
* Standard input/output devices

## 1.3 Project Description

This application will provide developers and users an efficient tool to enhance and effectively create webpages.

## 1.4 Requirement Priority Matrix

Below is the requirement priority matrix with High, Medium and Low categories.

|  |  |  |
| --- | --- | --- |
| Requirement | Function | Priority |
| Open JAR File | Double click, or right click and select "open | High |
| Create new HTML File | Click "File" select "new" | High |
| Name HTML File | enter filename in text window | High |
| Click Save | File is saved | High |
| Select "Basic Elements" button on toolbar | select the desired element/tag from the drop down | High |
| Click Undo | last action is deleted | High |
| Select "Formatting" button | select the desired element/tag from the drop down | High |
| Select "Lists" button | select the desired element/tag from the drop down | High |
| Select "Tables" button | select the desired element/tag from the drop down | High |
| Select "Forms and Input" button | select the desired element/tag from the drop down | Mid |
| Select "Frames" button | select the desired element/tag from the drop down | Mid |
| Select "Images" button | select the desired element/tag from the drop down | Mid |
| Select "Links" button | select the desired element/tag from the drop down | Mid |
| Select "Styles and Semantics" button | select the desired element/tag from the drop down | Low |
| Select "Audio Video" button | select the desired element/tag from the drop down | Low |
| Select "Meta Info" button | select the desired element/tag from the drop down | Low |
| Select "Programming" button | select the desired element/tag from the drop down | Low |

*Table 1 Requirement Priority (Use Cases)*

# 2. Project Plan

## 2.1 Work Plan

### 2.1.1 Work Allocations and Activities

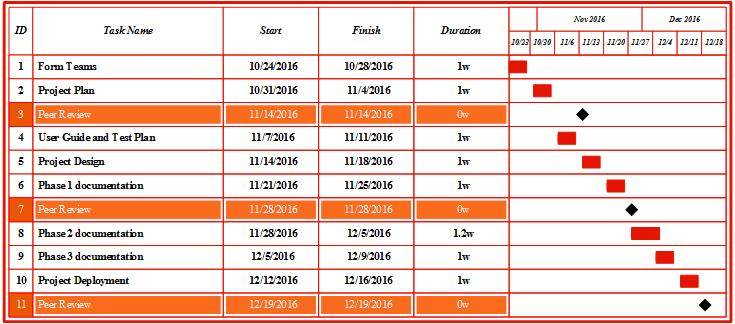
Table 2 presents the team members roles and what each role’s purpose is for developing this project.

|  |  |  |
| --- | --- | --- |
| **Role** | **Name** | **Activities** |
| Project Manager | Cesar Lopez | In charge of making sure everyone is communicating. He reports to the client. He puts every one’s work together. |
| Lead Design | David Chew | In charge of the design of the project. Thinks of the requirements and specification this application needs to be created. |
| Lead User’s Guide | Steve Dickinson | In charge of creating a user’s guide. He thinks about how this application will be used and coder will make it happen. |
| Lead Tester | Vijay Patel | In charge of testing. He tests the application with different scenarios to make sure everything work and to see what needs to be fixed. |
| Lead Coder | Chris Victores | In charge of the coding. He will use the design and user’s guide to understand how this application should be coded. |

*Table 2. Work allocations and Activities*

### 2.1.2 Key Milestones

This project milestone is illustrated below.



*Figure 1. Project Milestones*

# 3. User’s Guide

## 3.1 HTML Tutorials

a. [w3schools HTML5 Tutorial](http://www.w3schools.com/html/default.asp)

b. [codecademy](https://www.codecademy.com/learn/web)

## 3.2 Location

Once you have downloaded the HTML Editor, you’ll need to decide where to store the icon. You will simply double click or right click and select “open”.



Figure 2 – Opening your HTML Editor for the first time.

## 

## 3.3 HTML Editor Layout

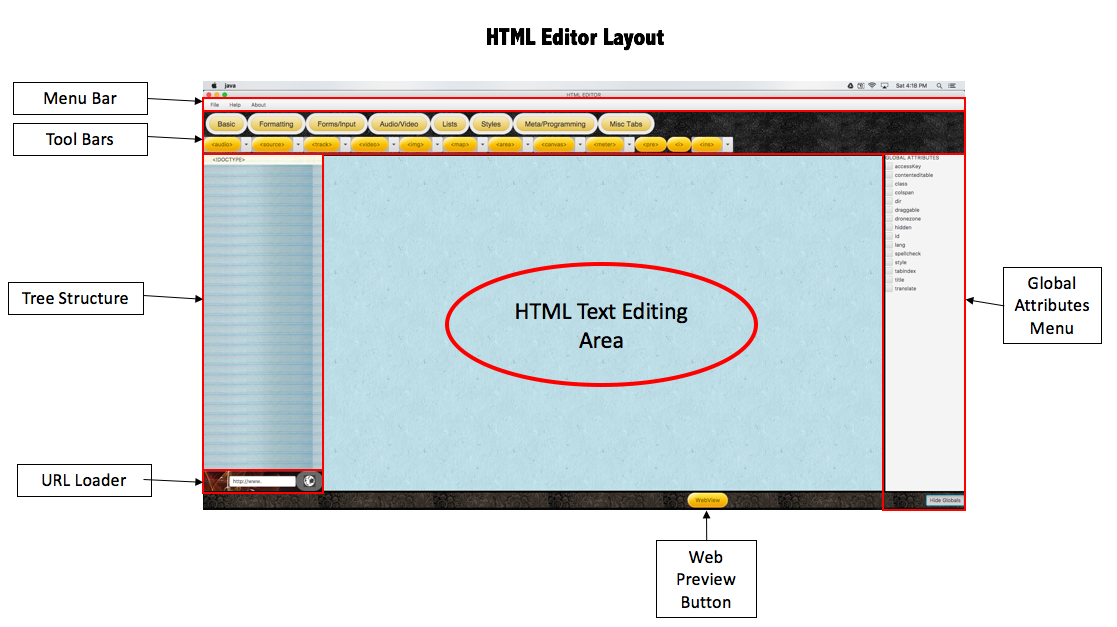


Figure 3 – HTML Editor Layout

## 3.4 Creating a new HTML Document

To create a new project, select “File” from the menu bar, then select “New Project” and “Add a project”. This begins a new HTML File that will need to be named. Enter the name of your project in the text box provided and select ok. You will then be able to being adding HTML to the text area.

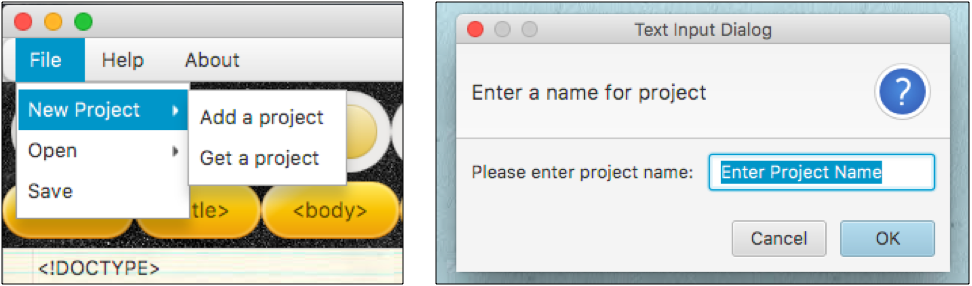


Figure 4 – Creating a new HTML File

## 

## 3.5 Opening a sample HTML Project

To open a sample project, select “File”, then “New Project”, “Get a project”. Then select one of the options in the “Select a Project” window that opens and click “Load Project”.

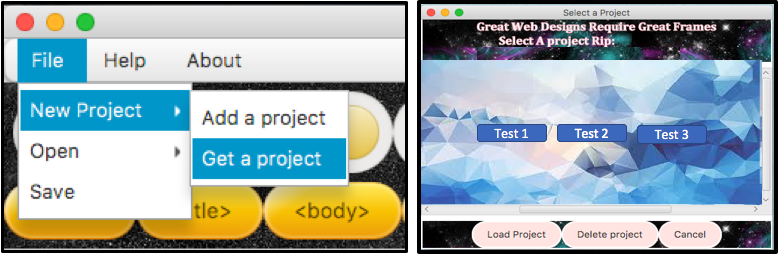


Figure 5 – Using Sample Project

## 3.6 Opening an existing HTML Project

To open a project, you can select either from a “.html” format or “.txt” format. This will open a file chooser window to navigate to the specific file you wish to import.

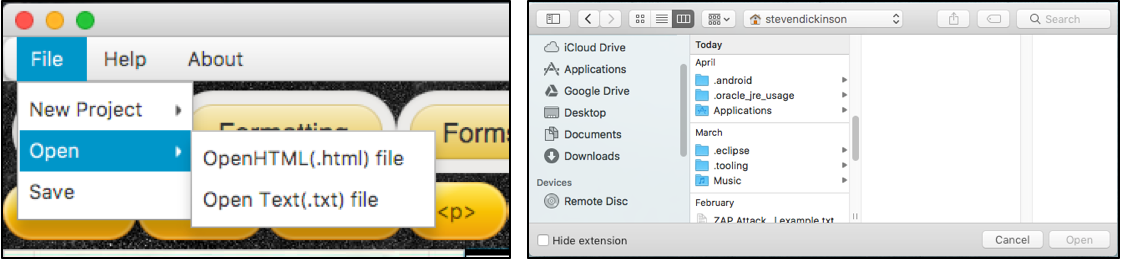


Figure 6 – Using an existing project.

## 3.7 Selecting an existing URL

To view the HTML of a particular site, enter the URL in the URL Loader and click the world icon. This will connect to the site and load the HTML into the HTML text editor.

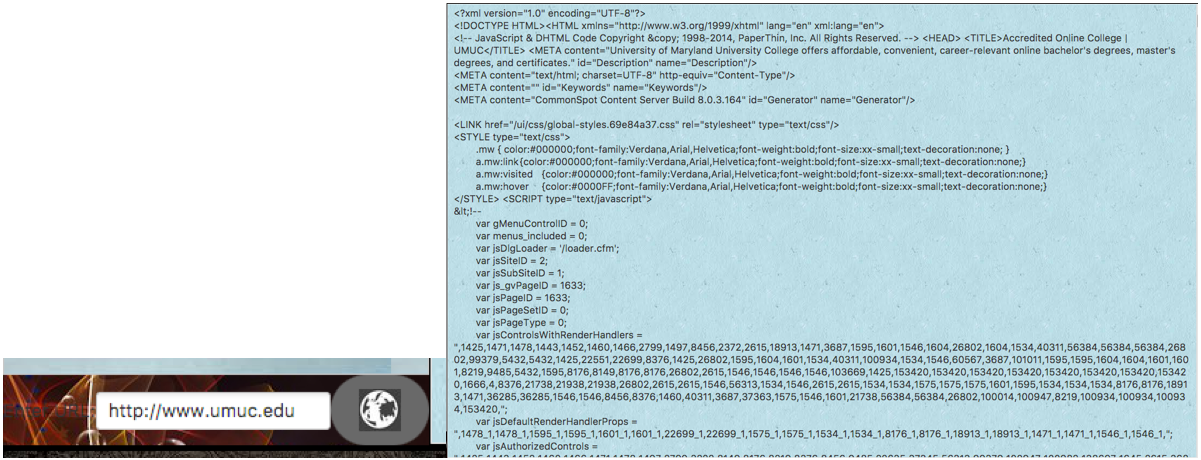


Figure 7 – Loading HTML from a website

## 3.8 Getting Help

To access this document or the design document select “Help” from the menu bar and select either User Guide or Design. This will open the User Guide in a new window (Figure 9)

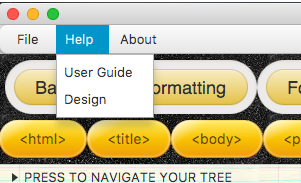


Figure 8 – Accessing user instructions

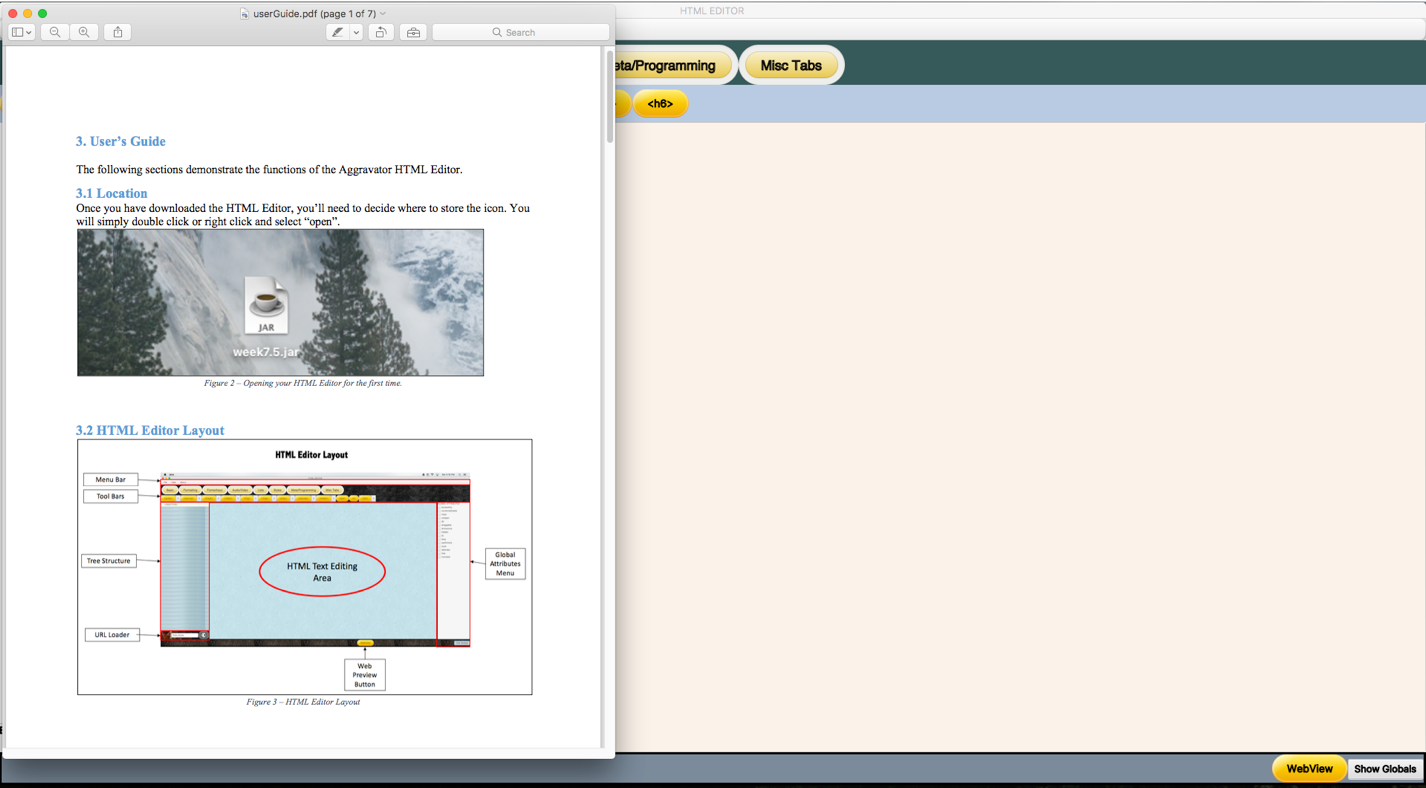
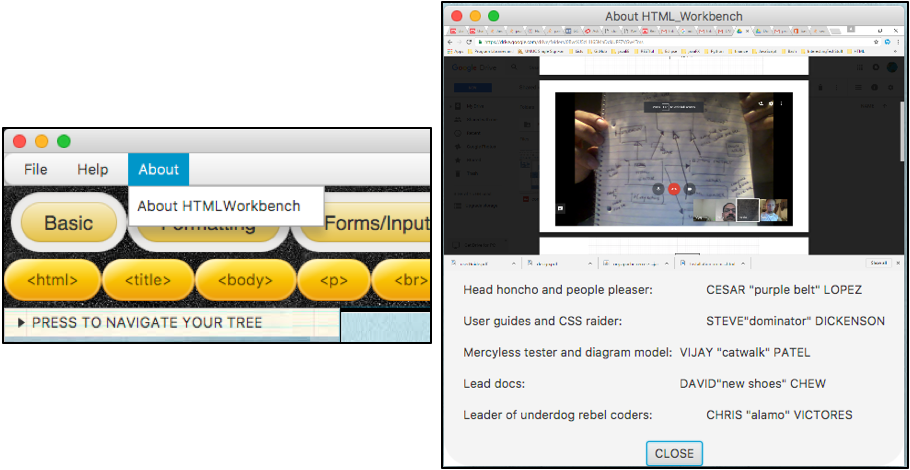


Figure 9

## 3.9 Project information

To access information about this project, select the “About” button from the menu bar and click “About HTML Workbench”. This will open the About HTML Workbench team as well as play a select tune.



*Figure 10 – Learning about the project team*

## 3.10 Switching between Tab categories

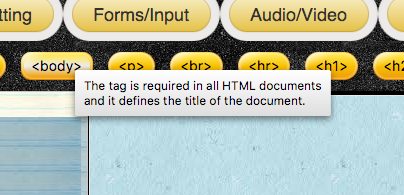
The top row of the tools bar provides tabs to switch between different categories of HTML Tags.



*Figure 11 – Tool bar layout*

## 3.11 Tool tips

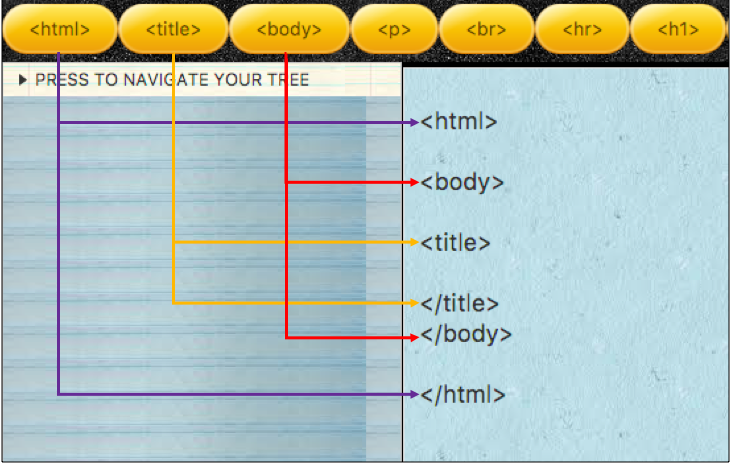
Hovering over a tag will show a small window popup explaining the purpose of each tag.



*Figure 12 – Tool Tips*

## 3.12 Populating the editor with HTML tags

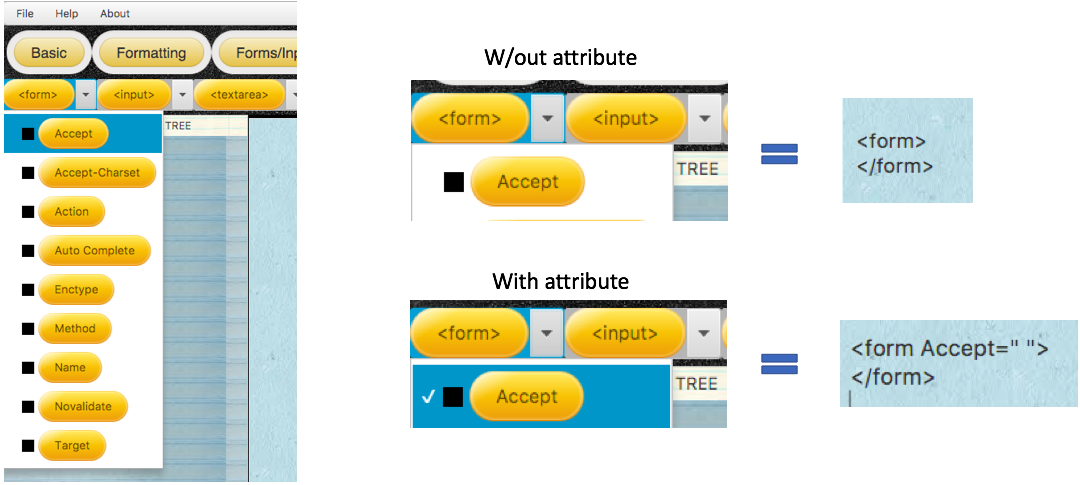
Each tag button clicked will add that tag to the HTML Editor area.



*Figure 13 – Adding HTML tags to the Editor Area*

## 3.13 Selecting and using attributes

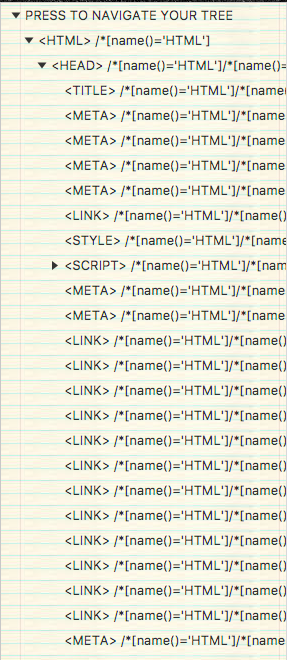
This editor also has additional options with each tag. Selecting the dropdown area will show the attribute options with each tag. Selecting an attribute will populate a check mark next to that option. Then, selecting the tag, <form> in this example will add that HTML tag along with the attribute selected.



*Figure 14 – Adding Tags with Attributes.*

## 3.14 Navigating a large HTML Document.

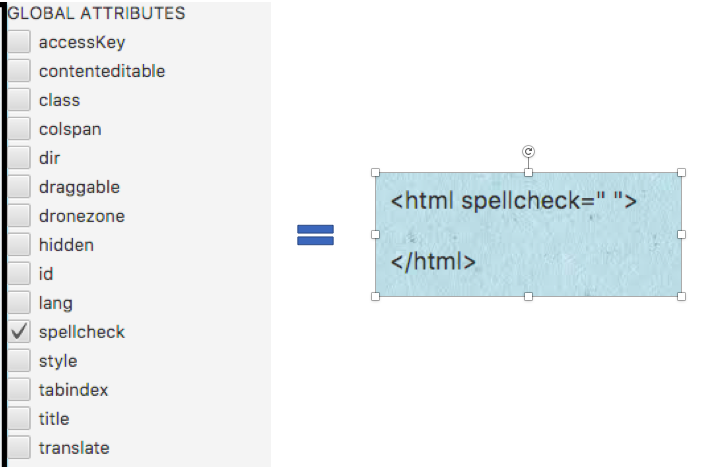
This example uses UMUC’s website. The HTML code is parsed into sections such as <HEAD>, <SCRIPT> and <BODY> and can be navigated to quickly by using the Tree pane.



*Figure 15 – Navigating through a large file with the Tree.*

## 3.15 Using Global Attributes

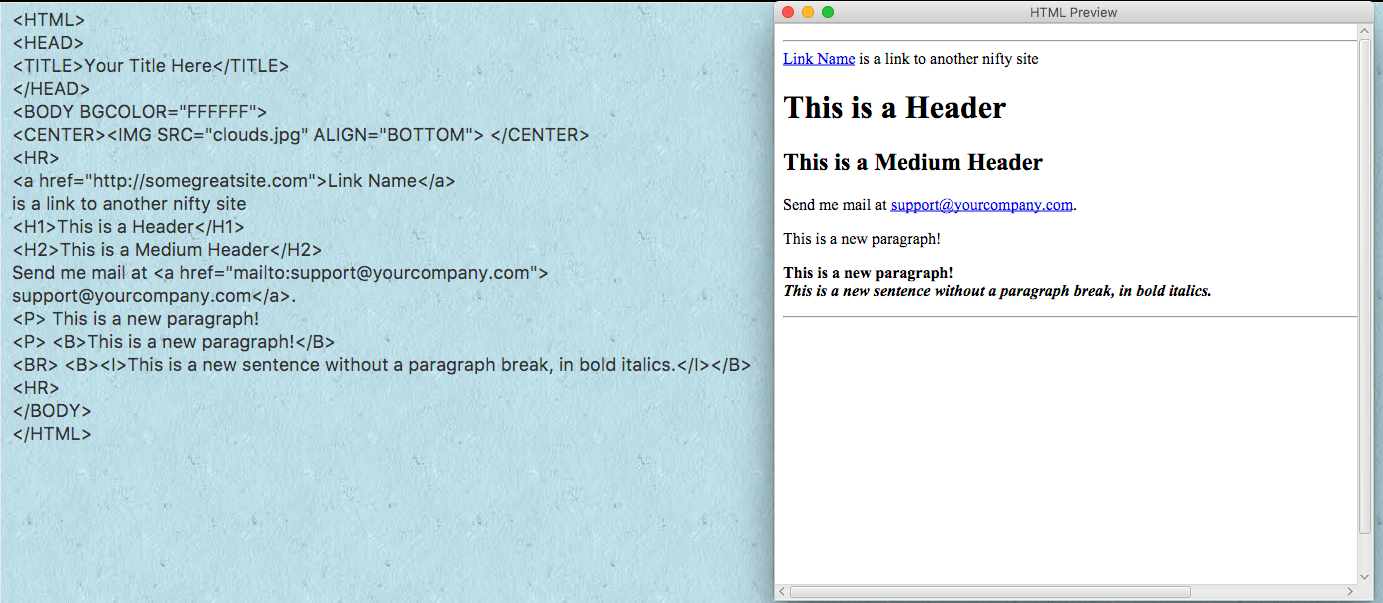
The ../../Screen%20Shot%202016-12-10%20at%208.31.55%20PM.png button is used to show the menu button below clicking ../../Screen%20Shot%202016-12-10%20at%208.31.44%20PM.png will make the menu pane disappear. The global attributes are used similar to the tag attributes but can be applied to any tag. The example below shows that when the “spellcheck” attribute is selected it will be applied to any HTML tags that are added to the HTML Editor area.



*Figure 16 – Adding Global Attributes.*

## 3.16 Previewing the HTML

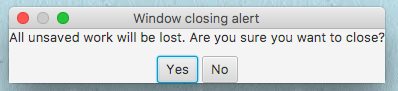
Selecting the ../../Screen%20Shot%202016-12-10%20at%208.45.49%20PM.png button will open a new window showing what the HTML will look in a browser window. The example below shows the HTML sample code in a new window.



*Figure 17 – Using the preview button*

## 3.17 Exiting the HTML Editor

The HTML Editor can be closed like any program. However, you will receive a popup window alerting you to save your work.



*Figure 18 – Closing out the Editor*

# 4. TEST PLAN

## 4.1 Introduction

The goal of this chapter is to organize and describe the architecture for building and testing this application. This application will be developed in multiple phases. Features and plan depicted in this document are subject to change. As the project progresses, application functionality will be revised as needed and additional functionality may be added or replaced with new functionality as needed.

## 4.2 Plan Purpose

This section illustrates the test plan’s approach and overall application testing framework for the GUI driven editor application for HTML. This document introduces:

• Test Strategy: the approach that testing element will use to define the process to set up a valid test such as testing criteria, creation of test cases, test task specification, scheduling, functional and unit testing strategy.

• Execution Strategy: These processes present how to perform testing, how to identify defects, how report defects, and how to fix defects and implement fixes.

• Test Management: This will outline the management of the testing including testing schedule, testing resource management, and events that come up during test plan execution.

## 4.3 Functional Test Plan

The objective of the test is to verify the overall functionality of HTML editor application works per the specifications. The test will execute and verify individual functionality.

To effectively test this application, we will implement unit testing, functional testing, and cross platform installation and functional testing.

Functional Test Plan will cover testing of each component of the HTML editor which will add HTML Tags and tag attribute. As per user’s requirement of the HTML editor tool, each requested functionality covered by providing tool bars, menus, and menu options/items. In this functional test plan, we will test following component functionality presented in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Description (Expected Result)** | **Test Results outcome** | **Comments** |
| Open JAR File | HTML editor successfully opens when double clicking icon (Installation test) | Pass/Fail |  |
| Create project | From the file menu create add project option allow to add new project. Second option Get project will allow load existing projects in the HTML Editor |  |  |
| Open Text file | Form the file menu open option🡪 open Text file sub menu will allow to open existing Text file from the drive in to the HTML Editor. | Pass/Fail |  |
| Open HTML File for edit | Form the file menu open option🡪 open HTML file sub menu will allow to open existing HTML file from the drive in to the HTML Editor. | Pass/Fail |  |
| Save option for files | Current HTML file in editor will saves successfully. This option will allow file to save as Text file or HTML file. | Pass/Fail |  |
| Select "Basic” button option on toolbar | Drop down button option displays HTML tag options in the submenu bar and populates following tag buttons options   * <html> tag * <title> tag * <body> tag * <p> tag * <br> tag * <hr> tag * <h1> tag * <h2> tag * <h3> tag * <h4> tag * <h5> tag * <h6> tag | Pass/Fail |  |
| Select "Formatting” button option on toolbar | Drop down button option displays HTML tag options in the submenu bar and populates following buttons formatting tag buttons:   * <b> tag * <abbr> tag * <address> tag * <bdi> tag * <bdo> tag * <cite> tag * <del> tag * <code> tag * <progress> tag * <rp> tag * <s> tag * <samp> tag | Pass/Fail |  |
| “Audio Video" button to add Audio video related tags | Adds selected Audio/Video element/tag from the drop down into the editor | Pass/Fail |  |
| Select " Audio/video” button option on toolbar | Drop down button option displays HTML tag options in the submenu bar and populates following buttons formatting tag buttons:   * <audio> tag * <source> tag * <tack> tag * <video> tag * <img> tag * <map> tag * <area> tag * <canvas> tag * <meter> tag * <pre> tag * <i> tag * <ins> tag | Pass/Fail |  |
|  |  |  |  |
| Select "Formatting" button to add formatting HTML tags | Adds selected Formatting element/tag from the drop down into the editor sub menu | Pass/Fail |  |
| Select "Lists” button option on toolbar | Drop down button option displays HTML tag options in the submenu bar and populates following buttons formatting tag buttons:   * <ul> tag * <ol> tag * <li> tag * <di> tag * <dt> tag * <dd> tag * <table> tag * <caption> tag * <coigroup> tag * <col> tag * <a> tag * <link> tag | Pass/Fail |  |
| Select "Lists" button to add List HTML tags | Adds selected List element/tag from the drop down into the editor | Pass/Fail |  |
| Select "Tables" button to add Table related HTML tags | Adds selected Table element/tag from the drop down into the editor | Pass/Fail |  |
| Select "Forms/Input” button option on toolbar | Drop down button option displays HTML tag options in the submenu bar and populates following buttons formatting tag buttons:   * <form> tag * <input> tag * <textarea> tag * <button> tag * <select> tag * <optgroup> tag * <option> tag * <label> tag * <fieldset> tag * <legend> tag * <datalist> tag * <keygen> tag | Pass/Fail |  |
| Select "Forms and Input" button to add Forms and Input related tags | Adds selected Form and Input element/tag from the drop down into the editor | Pass/Fail |  |
| Select "Frames" button to add Frames related tags | Adds selected Frames element/tag from the drop down into the editor | Pass/Fail |  |
| Select "Images" button to add Images related tags | Adds selected Images element/tag from the drop down into the editor | Pass/Fail |  |
| Select "Links" button to add Links related tags | Adds selected Links element/tag from the drop down into the editor | Pass/Fail |  |
| Select "Styles” button option on toolbar to added submenu related to Style tags | Drop down button option displays HTML tag options in the submenu bar and populates following buttons formatting tag buttons:   * <style> tag * <div> tag * <span> tag * <header> tag * <footer> tag * <main> tag * <section> tag * <article> tag * <field set> tag * <aside> tag * <details> tag * <dialog> tag * <summary> tag | Pass/Fail |  |
| Select "Styles and Semantics" button to add Style and Semantics related tags submenu | Adds selected Styles and Semantics element/tag from the drop down into the editor | Pass/Fail |  |
| “Meta/Programming" button to add meta data and programming related tags | Adds selected Meta/Programming element/tag from the drop down into the editor | Pass/Fail |  |
| Select "Meta/Programming” button option on toolbar to added submenu related to Meta/Programming tags | Drop down button option displays HTML tag options in the submenu bar and populates following buttons formatting tag buttons:   * <head> tag * <meta> tag * <base> tag * <script> tag * <noscript> tag * <embed> tag * <object> tag * <param> tag * <wbr> tag * <sub> tag * <sup> tag | Pass/Fail |  |
| cross platform installation | Cross platform Installation and functional testing will cover testing of application installation on different OS platforms such as Windows 10, Linux, and mac OS. | Pass/Fail |  |
| Select " Misc tabsg” button option on toolbar to added submenu related to Misc HTML tags | Drop down button option displays HTML tag options in the submenu bar and populates following buttons formatting tag buttons:   * <dfn> tag * <mark> tag * <small> tag * <time> tag * <output> tag * <iframe> tag * <u> tag * <nav> tag * < blockquote> tag * <q> tag * <kbd> tag | Pass/Fail |  |
| Select "Misc tabs" button to add Misc tag options | Adds selected Misc tabs element/tag from the drop down into the editor | Pass/Fail |  |
| Web search for entered web URL in search box. | Entering web URL in search box and Editor will web search web page and load into the editor to edit it. | Pass/Fail |  |
| Select Help menu will display user guide and help related to the tags | From the main menu after selecting Help option this will give two options one for user guide and second for design document and tags related information | Pass/Fail |  |
| Select About menu will display information about HMTL Editor | From the main menu after selecting About option this will give HTML Editor version and other related information. | Pass/Fail |  |
| Select "Show/Hide Global attribute" button to add global attributes with chosen tags | "Show/Hide Global attribute" button to select attributes to add with chosen tags in the Editor window. There are following global attributes:   * Accesskey * Conatinteditable * Class * Colspan * Div * Graggables * Dronzone * Hidden * Id * Lang * Spellcheck * Style * Tabindex * Title * Translate | Pass/Fail |  |
| Select "WebView" button to preview in progress or completed HTML document as webpage. | Will open another separate window in with document will display as HTML web page | Pass/Fail |  |
| Select "Close" button or close widow to close application | Select "Close" button or close widow to close application. This operation will display popup window to prompt to save and close or close without saving. | Pass/Fail |  |

*Table 3. Functional test plan*

**Note:**  As requirements will be revised, new functional test plan items will be added in List

## 4.4 Test Result Summary:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case No | Test case Description | Test Results | Tester Name | Error Assign To | Retesting Results |
| Test Case-01 | New Project - Add  project option | Pass | Vijay Patel | N/A | N/A |
| Test Case-02 | New Project - Get  project option | Pass | Vijay Patel | N/A | N/A |
| Test Case-03 | New Project - Open file option | Pass | Vijay Patel | N/A | N/A |
| Test Case-04 | Save File option | Pass | Vijay Patel | N/A | N/A |
| Test Case-05 | Help menu option | Pass | Vijay Patel | N/A | N/A |
| Test Case-06 | About menu option | Pass | Vijay Patel | N/A | N/A |
| Test Case-07 | WebView option | Pass | Vijay Patel | N/A | N/A |
| Test Case-08 | Navigation Tree option | Pass | Vijay Patel | N/A | N/A |
| Test Case-09 | Show Global button option | Pass | Vijay Patel | N/A | N/A |
| Test Case-10 | Basic menu button option | Pass | Vijay Patel | N/A | N/A |
| Test Case-11 | Formatting button option | Pass | Vijay Patel | N/A | N/A |
| Test Case-12 | Lists button option on toolbar | Pass | Vijay Patel | N/A | N/A |
| Test Case-13 | Forms/Input button option on toolbar | Pass | Vijay Patel | N/A | N/A |
| Test Case-14 | Styles button option on toolbar | Pass | Vijay Patel | N/A | N/A |
| Test Case-15 | Meta/Programming” button option | Pass | Vijay Patel | N/A | N/A |
| Test Case-16 | For "Misc tags” button option on toolbar | Pass | Vijay Patel | N/A | N/A |
| Test Case-17 | Web search functionality | Fail | Vijay Patel | Chris Victores/ Cesar Lopez | Pass |
| Test Case-19 | Test for <hr> tag button | Fail | Vijay Patel | Chris Victores | Pass |
| Test Case-20 | Test for <cite> tag button | Fail | Vijay Patel | Chris Victores | Pass |

### 4.4.1 Test Case-01

| HTML Editor Application | | |
| --- | --- | --- |
| **Interface Component Test - Case black-box test For the New Project option** | | |
| **Test Items:** This test will test New project option of File menu of the HTML Editor Application. | | |
| 1. Test case identifier: | Test Case-01 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test for the File menu 🡪 New Project🡪 New Project interface option as planed in Test plan. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Adding New Project in HTML Editor Application. | |
| **5. Test Output Specifications:**   * Enter New Project name dialog box will appear. * New Project added in Editor Application. | |
| **6. Steps to Reproduce this test:**  Click File menu🡪 New project 🡪 Add Project🡪 Enter New Project Name Dialog Box open-up 🡪Enter New Project Name🡪 Press OK button 🡪 New Project created in the HTML Editor Application. | |
| **7. Test Case Result: Pass** | |

### 4.4.2 Test Case-02

| HTML Editor Application | | |
| --- | --- | --- |
| **Interface Component Test - Case black-box test For the Get Project option** | | |
| **Test Items:** This test will test Get project option of File menu of the HTML Editor Application. | | |
| 1. Test case identifier: | Test Case-02 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test Get Project from the File menu 🡪 New Project🡪 Get Project interface option as planed in Test plan. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Open Exiting Project using list form open Project dialog box. | |
| **5. Test Output Specifications:**   * Select Project name from existing project dialog box. * Selected Project added in Editor Application. | |
| **6. Steps to Reproduce this test:**  Click File menu🡪 New project 🡪 Get Project🡪 Select Project Name from Dialog Box 🡪Enter Project Name dialog box open 🡪 Select Project form Project List -🡪Press OK button 🡪 Project will added the HTML Editor Application. | |
| **7. Test Case Result : Pass** | |

### 4.4.3 Test Case-03

| HTML Editor Application | | |
| --- | --- | --- |
| **Interface Component Test - Case black-box test For the File Open option** | | |
| **Test Items:** This test will test File Open option of File menu of the HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-03 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test File open option from the File menu 🡪 Open file from the drive🡪HTML and TXT file open interface option as planed in Test plan. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  File from user's drive and load HTML or TXT file using open file wizard to open file dialog box. | |
| **5. Test Output Specifications:**   * Select File name ( HTML or TXT) from the drive using File open dialog box. * Selected file ( HTML or TXT) open successfully in the in Editor Application. | |
| **6. Steps to Reproduce this test:**  Click File menu🡪 Open 🡪 Follow open file wizard 🡪 select HTML or TXT file from the drive🡪 Press OK 🡪File open successfully in the HTML Editor Application. | |
| **7. Test Case Result : Pass** | |

### 4.4.4 Test Case-04

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For the Save File option | | |
| **Test Items:** This test will test Save File option of File menu of the HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-04 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test File Save option from the File menu 🡪 Save file (HTML and TXT file) to the drive option as planed in Test plan. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Existing **open** File from the HTML | |
| **5. Test Output Specifications:**   * Save open File ( HTML or TXT) from the editor as name given by the user to the drive. | |
| **6. Steps to Reproduce this test:**  Click File menu🡪 Save option 🡪 Follow save file wizard 🡪 select HTML or TXT file type 🡪 select the drive🡪 Press OK 🡪File will store to user's drive | |
| **7. Test Case Result: Pass** | |

### 4.4.5. Test Case-05

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For the Help menu option | | |
| **Test Items:** This test will test Help menu option of the HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-05 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test Help menu option. This option will test two sub options of user guide and Design guide |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click to Help menu and select Help or Design option | |
| **5. Test Output Specifications:**   * After selecting Help menu user can see User guide or Design document guide. | |
| **6. Steps to Reproduce this test:**  Click Help menu🡪 Select Help sub option 🡪 User Guide document will open  Click Help menu🡪 Select Design sub option 🡪 Design Guide document will open | |
| **7. Test Case Result: Pass** (Remark 🡪 Make sure that latest document should be attach to this option.) | |

### 4.4.6. Test Case-06

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For the About menu option | | |
| **Test Items:** This test will test About menu option of the HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-06 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About menu option. This option will test provide information about HTML editor. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click to About menu | |
| **5. Test Output Specifications:**   * After selecting About menu user can see information document about HTML Editor . | |
| **6. Steps to Reproduce this test:**  Click About menu🡪 Information document about HTML Editor will open | |
| **7. Test Case Result: Pass** (original document need to replace. Remark 🡪 Make sure that original document should be attach to this option.) | |

### 4.4.7. Test Case-07

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For the WebView option of HTML Editor | | |
| **Test Items:** This test will test WebView option of the HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-07 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About WebVIew option. This option will test HTML document look as web page just Like Page view in browser. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Current Open document or open from drive using File🡪 Open option 🡪 select HTML document | |
| **5. Test Output Specifications:**   * Current open document will be view as web Pre view; preview as in web browser as web page. | |
| **6. Steps to Reproduce this test:**  Current Open document or open from drive using File🡪 Open option 🡪 select HTML document 🡪 Click WebView button 🡪 Information document about HTML Editor will open | |
| **7. Test Case Result : Pass** | |

### 4.4.8. Test Case-08

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For the Navigation Tree option of HTML Editor | | |
| **Test Items:** This test will test Navigation Tree option of the HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-08 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About Navigation Tree option. This option will test HTML document's Tag stack tree structure. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Current Open document or open from drive using File🡪 Open option 🡪 select HTML document🡪 select Press to Navigation Tree option | |
| **5. Test Output Specifications:**  Tags Tree Structure of the Current open document will be view so that user can easily fix error related to missing tag pair. | |
| **6. Steps to Reproduce this test:**  Current Open document or open from drive using File🡪 Open option 🡪 select HTML document 🡪 Click to open navigation tree option 🡪 Tags Tree Structure of the Current open document will be view | |
| **7. Test Case Result : Pass** | |

### 4.4.9. Test Case-09

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For the Show Global button option of HTML Editor | | |
| **Test Items:** This test will Show Global button of the HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-09 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About Show Global button. This option will test global button to add global attributes for selected TAGs from the menu option.. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on show attribute 🡪 select Global attribute check box to add attributes for Tags | |
| **5. Test Output Specifications:**   * Selected attribute will inserted with selected TAG into Editor's edit Pan | |
| **6. Steps to Reproduce this test:**  Click on show Global button🡪 Check box to select attributes🡪 select menu 🡪Press tag button from sub menu🡪 selected attribute inserted into editor Pan. **This Test case is repeated for all Attributes to test.** | |
| **7. Test Case Result: Pass** | |

### 4.4.10. Test Case-10

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For the Basic menu button option of HTML Editor | | |
| **Test Items:** This test will test Basic button from the menu of the HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-10 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About Basic button from the menu. This option will test Basic button to show sub menu of the Tags related to Basic menu option. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Basic menu button from the main menu list | |
| **5. Test Output Specifications:**  All tags related Basic HTML tags group will appear as submenus button group. This submenu group should have following tag buttons. <html> tag   * <title> tag * <body> tag * <p> tag * <br> tag * <hr> tag * <h1> tag * <h2> tag * <h3> tag * <h4> tag * <h5> tag * <h6> tag | |
| **6. Steps to Reproduce this test:**  Click on Basic menu button in the main menu**.** | |
| **7. Test Case Result : Pass** | |

### 4.4.11. Test Case-11

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For "Formatting” button option on toolbar of HTML Editor | | |
| **Test Items:** This test will test "Formatting” button option on toolbar HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-11 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About "Formatting” button option on toolbar. This option will test Formatting button to add submenu for of the Tags related to Formatting menu option. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Formatting button from the main menu Tool bar. | |
| **5. Test Output Specifications:**  All tags related Formatting HTML tags group will appear as submenus button group. This submenu group should have following tag buttons. <b> tag   * <abbr> tag * <address> tag * <bdi> tag * <bdo> tag * <cite> tag * <del> tag * <code> tag * <progress> tag * <rp> tag * <s> tag * <samp> tag | |
| **6. Steps to Reproduce this test:**  Click on Formatting menu button in the main menu Tool bar**.** | |
| **7. Test Case Result : Pass** | |

### 4.4.12. Test Case-12

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For " Lists” button option on toolbar of HTML Editor | | |
| **Test Items:** This test will test " Lists” button option on toolbar HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-12 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About " Lists ” button option on toolbar. This option will test Lists button to add submenu for of the Tags related to Lists menu option. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Lists button from the main menu Tool bar. | |
| **5. Test Output Specifications:**  All tags related Lists HTML tags group will appear as submenus button group. This submenu group should have following tag buttons. <ul> tag   * <ol> tag * <li> tag * <di> tag * <dt> tag * <dd> tag * <table> tag * <caption> tag * <coigroup> tag * <col> tag * <a> tag * <link> tag | |
| **6. Steps to Reproduce this test:**  Click on Lists menu button in the main menu Tool bar**.** | |
| **7. Test Case Result : Pass** | |

### 4.4.13. Test Case-13

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For " Forms/Input” button option on toolbar of HTML Editor | | |
| **Test Items:** This test will test " Forms/Input” button option on toolbar HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-13 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About " Forms/Input” button option on toolbar. This option will test Forms/Input button to add submenu for of the Tags related to Forms/Input menu option. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Forms/Input button from the main menu Tool bar. | |
| **5. Test Output Specifications:**  All tags related Forms/Input HTML tags group will appear as submenus button group. This submenu group should have following tag buttons   * <form> tag * <input> tag * <textarea> tag * <button> tag * <select> tag * <optgroup> tag * <option> tag * <label> tag * <fieldset> tag * <legend> tag * <datalist> tag * <keygen> tag | |
| **6. Steps to Reproduce this test:**  Click on Forms/Input menu button in the main menu Tool bar**.** | |
| **7. Test Case Result: Pass** | |

### 4.4.14. Test Case-14

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For " Styles” button option on toolbar of HTML Editor | | |
| **Test Items:** This test will test " Styles ” button option on toolbar HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-14 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About " Styles” button option on toolbar. This option will test Forms/Input button to add submenu for of the Tags related to Styles menu option. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Styles button from the main menu Tool bar. | |
| **5. Test Output Specifications:**  All tags related Styles HTML tags group will appear as submenus button group. This submenu group should have following tag buttons   * <head> tag * <meta> tag * <base> tag * <script> tag * <noscript> tag * <embed> tag * <object> tag * <param> tag * <wbr> tag * <sub> tag * <sup> tag | |
| **6. Steps to Reproduce this test:**  Click on Styles menu button in the main menu Tool bar**.** | |
| **7. Test Case Result: Pass** | |

### 4.4.15. Test Case-15

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For " Meta/Programming” button option on toolbar of HTML Editor | | |
| **Test Items:** This test will test " Meta/Programming” button option on toolbar HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-15 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About "Meta/Programming” button option on toolbar. This option will test Meta/Programming button to add submenu for of the Tags related to Meta/Programming menu option. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Meta/Programming button from the main menu Tool bar. | |
| **5. Test Output Specifications:**  All tags related Meta/Programming HTML tags group will appear as submenus button group. This submenu group should have following tag buttons   * <head> tag * <meta> tag * <base> tag * <script> tag * <noscript> tag * <embed> tag * <object> tag * <param> tag * <wbr> tag * <sub> tag * <sup> tag | |
| **6. Steps to Reproduce this test:**  Click on Meta/Programming menu button in the main menu Tool bar**.** | |
| **7. Test Case Result: Pass** | |

### 4.4.16. Test Case-16

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For " Misc tags” button option on toolbar of HTML Editor | | |
| **Test Items:** This test will test " Misc tags” button option on toolbar HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-16 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About " Misc tags” button option on toolbar. This option will test Misc tags button to add submenu for of the Tags related to Misc tags menu option. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Misc tags button from the main menu Tool bar. | |
| **5. Test Output Specifications:**  All tags related Misc tags HTML tags group will appear as submenus button group. This submenu group should have following tag buttons   * <dfn> tag * <mark> tag * <small> tag * <time> tag * <output> tag * <iframe> tag * <u> tag * <nav> tag * < blockquote> tag * <q> tag * <kbd> tag | |
| **6. Steps to Reproduce this test:**  Click on Misc tags menu button in the main menu Tool bar**.** | |
| **7. Test Case Result : Pass** | |

### 4.2.17. Test Case-17

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For " Meta/Programming” button option on toolbar of HTML Editor | | |
| **Test Items:** This test will test " Meta/Programming” button option on toolbar HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-17 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test About "Meta/Programming” button option on toolbar. This option will test Meta/Programming button to add submenu for of the Tags related to Meta/Programming menu option. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Meta/Programming button from the main menu Tool bar. | |
| **5. Test Output Specifications:**  All tags related Meta/Programming HTML tags group will appear as submenus button group. This submenu group should have following tag buttons   * <head> tag * <meta> tag * <base> tag * <script> tag * <noscript> tag * <embed> tag * <object> tag * <param> tag * <wbr> tag * <sub> tag * <sup> tag | |
| **6. Steps to Reproduce this test:**  Click on Meta/Programming menu button in the main menu Tool bar**.** | |
| **7. Test Case Result : Pass** | |

### 4.4.18. Test Case-18

## 

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For Web search functionality of HTML Editor | | |
| **Test Items:** This test will test Web search functionality option on toolbar HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-18 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test about Web search functionality. This option will test search for URL provided in text box area. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on into the text box 🡪enter URL to search. | |
| **5. Test Output Specifications:**  Entered URL will be search and HTML file will be loaded in the Editor | |
| **6. Steps to Reproduce this test:**  Click on into the text box 🡪enter URL to search 🡪 Press Go Button 🡪 search web will be loaded. | |
| **7. Test Case Result : FAIL : when I pressed go button without entering URL information unknown java exceptions error occurred** (Refer following figure screen shot for more information) | |

### 4.4.19. Test Case-19

## 

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For <hr> tag buttons of HTML Editor | | |
| **Test Items:** This test will test For <hr> tag buttons HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-19 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test about For <hr> tag buttons . This option will test <hr> </hr> tag pair in editor Pane area. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Basic button🡪 <hr> button tag . | |
| **5. Test Output Specifications:**  <hr> button tag option will Entre <hr> </hr> tag pair in editor Pane area. | |
| **6. Steps to Reproduce this test:**  Click on Basic button🡪 <hr> button tag🡪 <hr> </hr> tag pair in editor Pane area. | |
| **7. Test Case Result : FAIL : when I pressed <hr> tag button it entered <hr1> </hr1> tags pair which is wrong.** (Refer following figure screen shot for more information) | |

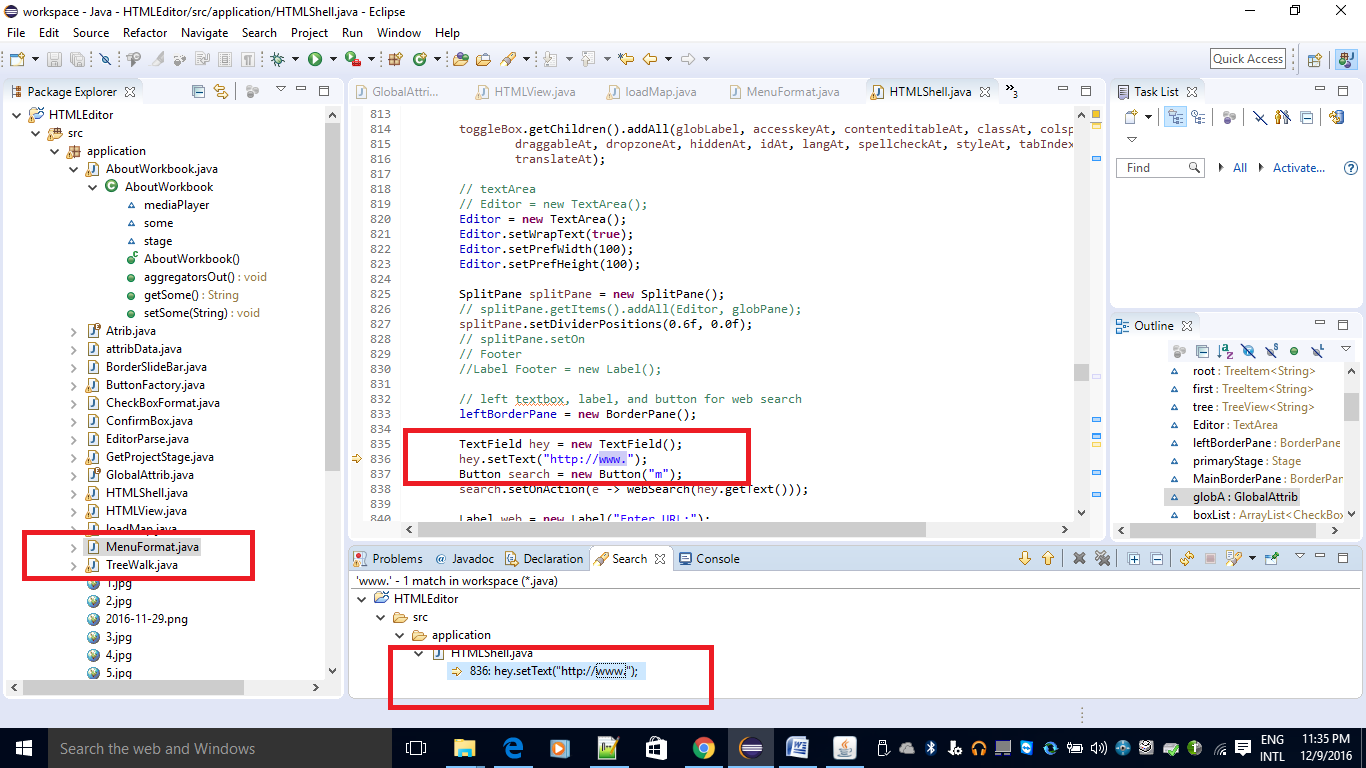
### 4.4.20. Test Case- 20

| HTML Editor Application | | |
| --- | --- | --- |
| Interface Component Test - Case black-box test For <cite> tag buttons of HTML Editor | | |
| **Test Items:** This test will test For <cite> tag buttons HTML Editor Application.. | | |
| 1. Test case identifier: | Test Case-20 | |
| **2. Test objective:** | The objective of this test case is to provide a black-box test about For <hr> tag buttons . This option will test <cite> </cite> tag pair in editor Pane area. |
| **3. Test** **performed** **by (Tester Name): Vijay Patel** | |
| **4. Test Input Specifications:**  Click on Formatting button🡪 <hr> button tag . | |
| **5. Test Output Specifications:**  <cite> button tag option will Entre <cite> </cite> tag pair in editor Pane area. | |
| **6. Steps to Reproduce this test:**  Click on Formatting button🡪 <cite> button tag🡪 <cite> </cite> tag pair in editor Pane area. | |
| **7. Test Case Result : FAIL : when I pressed <cite> tag button it entered <cite1> </cite1> tags pair which is wrong.** (Refer following figure screen shot for more information) | |

## 4.5. Test Case- Error

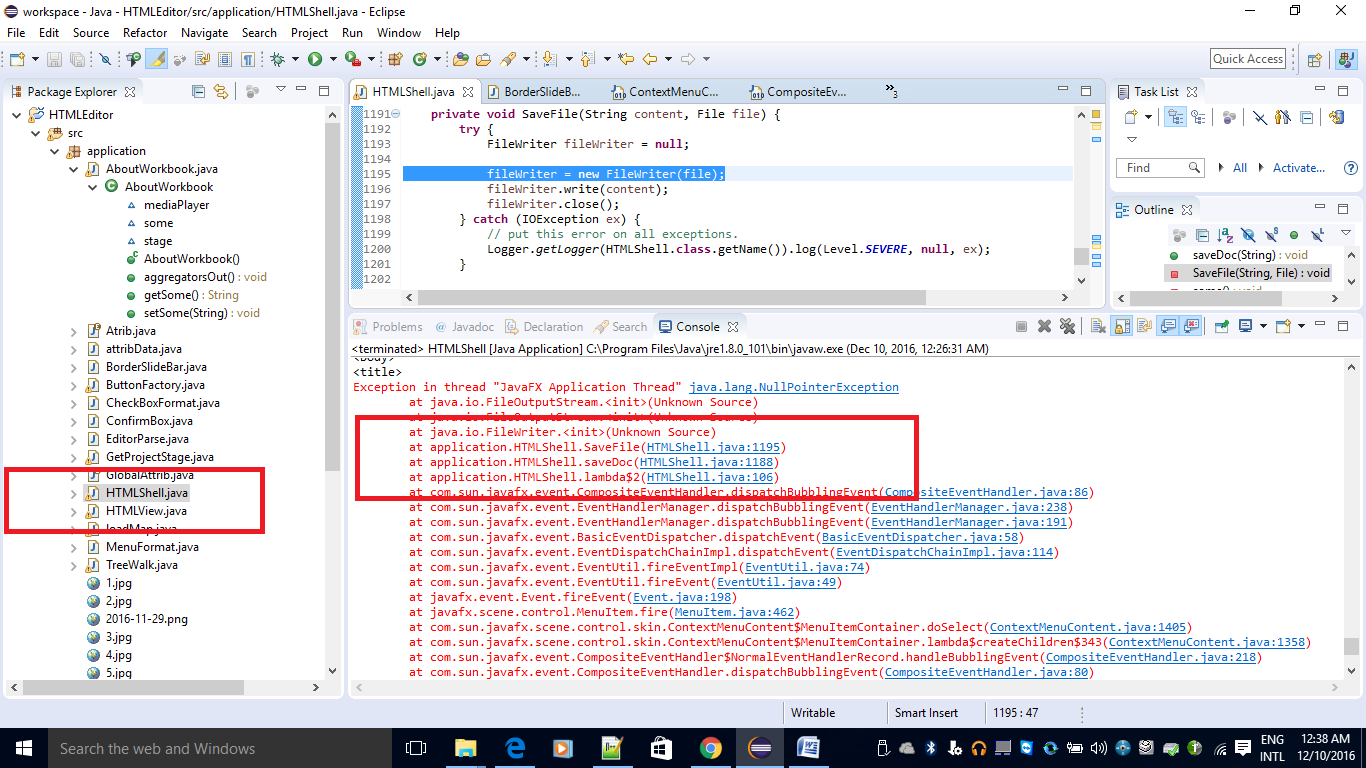
### 4.5.1 Test Case- Error - Screen shot - 1

This screen shot for Test case Id: Test Case-18



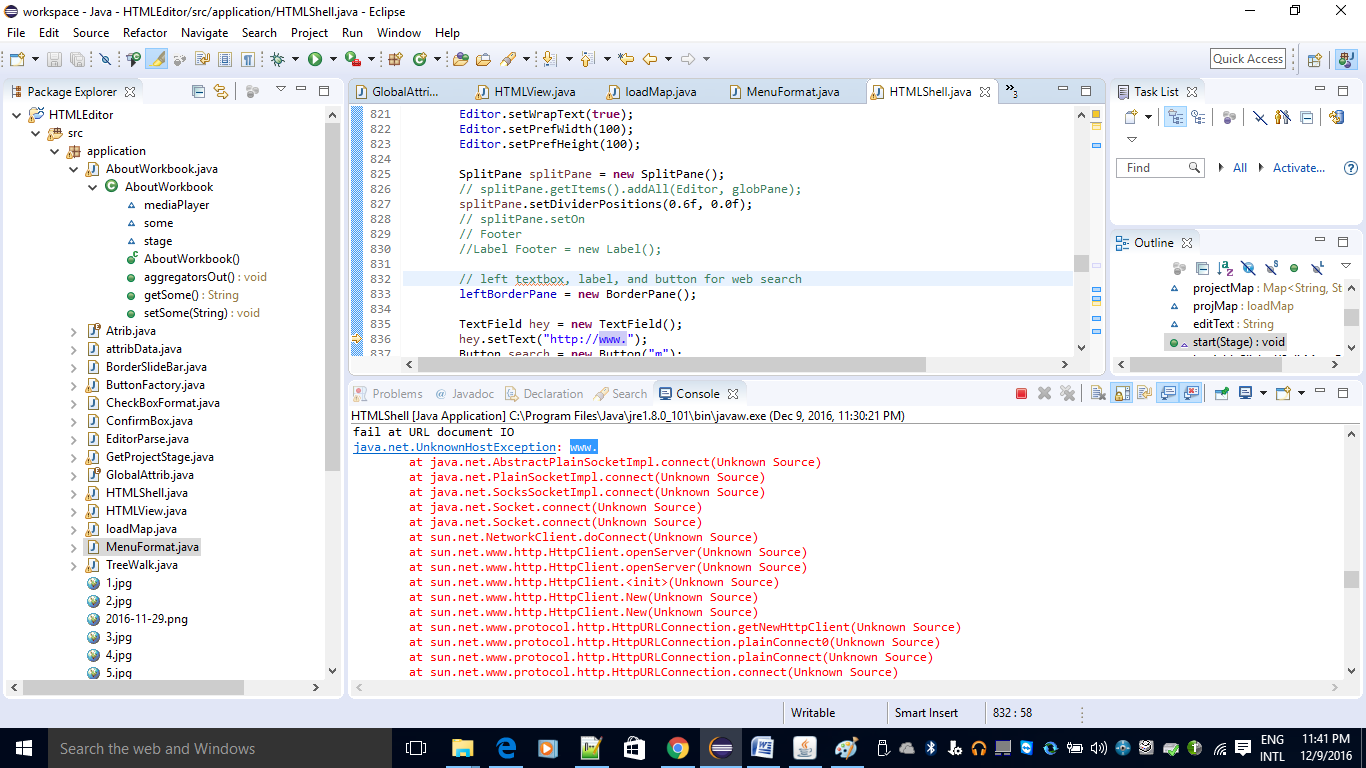
### 4.5.2 Test Case- Error - Screen shot - 2

This screen shot for Test Case Id: Test Case-18



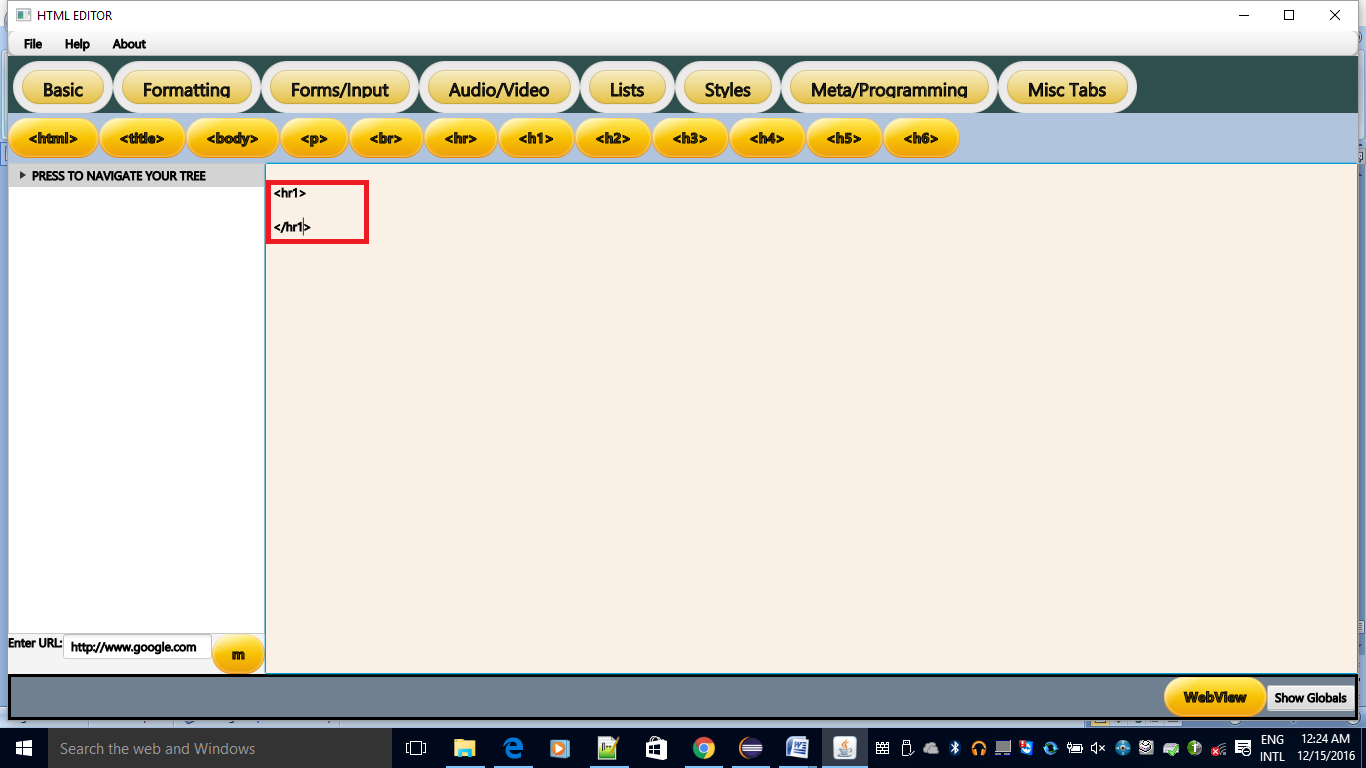
### 4.5.3 Test Case- Error - Screen shot - 3

This screen shot for Test case Id: Test Case-18



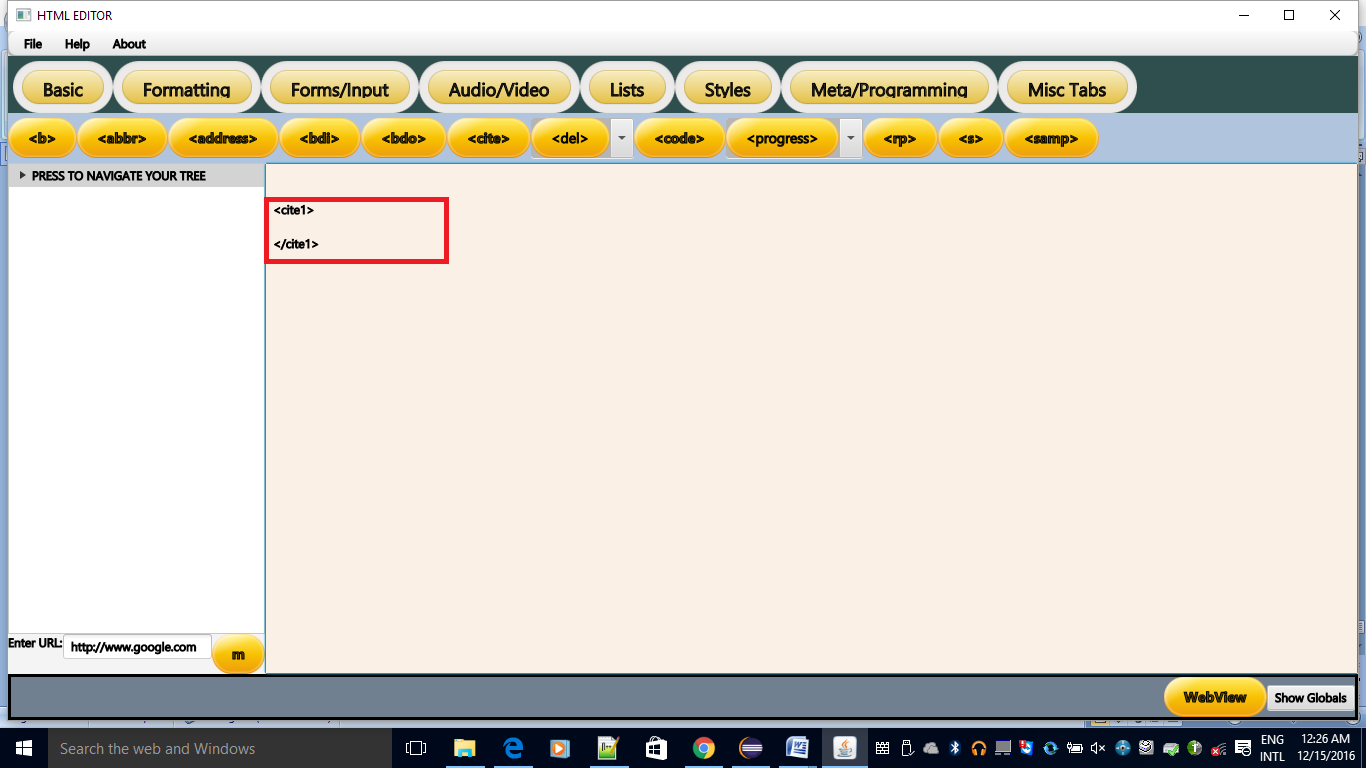
### 4.5.4 Test Case- Error - Screen shot -4

This screen shot for Test case Id: Test Case-19



### 4.5.5 Test Case- Error - Screen shot -5

This screen shot for Test Case Id: Test Case-20



## 4.6 Test Strategy

### 4.6.1 Test Assumptions

Key Assumptions are:

* The members of the testing team will run the application individually on their computers and document any errors to the google drive shared document.
* When a new functionality is added, we will perform all the testing again.
* Development Team will own test environment and preparation activities
* Development team will be provided plans to fix defect based on error log documentation.
* Bug fixing plan will be coordinated prior to applying the fixes on the Test environment.
* The project manager will provide test planning, test design, and test execution support.
* Project team has the knowledge and experience necessary, or has received adequate training in the system, the complete this project and the testing processes.

## 4.7 Test Plan Execution Strategy

### 4.7.1 Defect tracking & Reporting

It will be managed via using predefined versioned documentation template that is accessible by developers, testers, leads, and project manager from shared Google drive.

## 4.8 Test Management Process

To manage HTML editor application testing and reporting defects we will use version document template and define testing conditions in terms of software and hardware setup. Testing management process covers testing risk and mitigation factors as explain as below:

### 4.8.1 Test Execution Process

Once all Test cases are approved and the test environment is ready for testing, tester will start an exploratory test of the application to ensure the application is stable for testing.

# 5. Design Plan

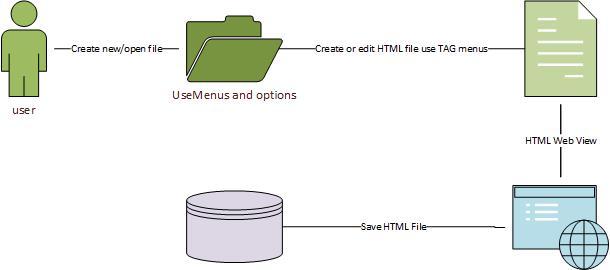
## 5.1 Introduction

The purpose of the Software design is to identify specific requirements necessary in the development of a management application the Aggravator HTML Editor. It provides essential description and detailed information of the application to meet the needs.

### 5.1.1 User Interface

The following User Interface diagram illustrates the flow of this application.

1. The user can create a new file or open an existing file.
2. This file then will be present in the editor’s main window for the user to edit.
3. User can then view the his or her work in the HTML Web View.
4. The user can then save the file and use it for whatever they like.



*Figure 18 User Interface Diagram*

## 5.2 Communication Diagram

The following diagram illustrates the communication between each module.

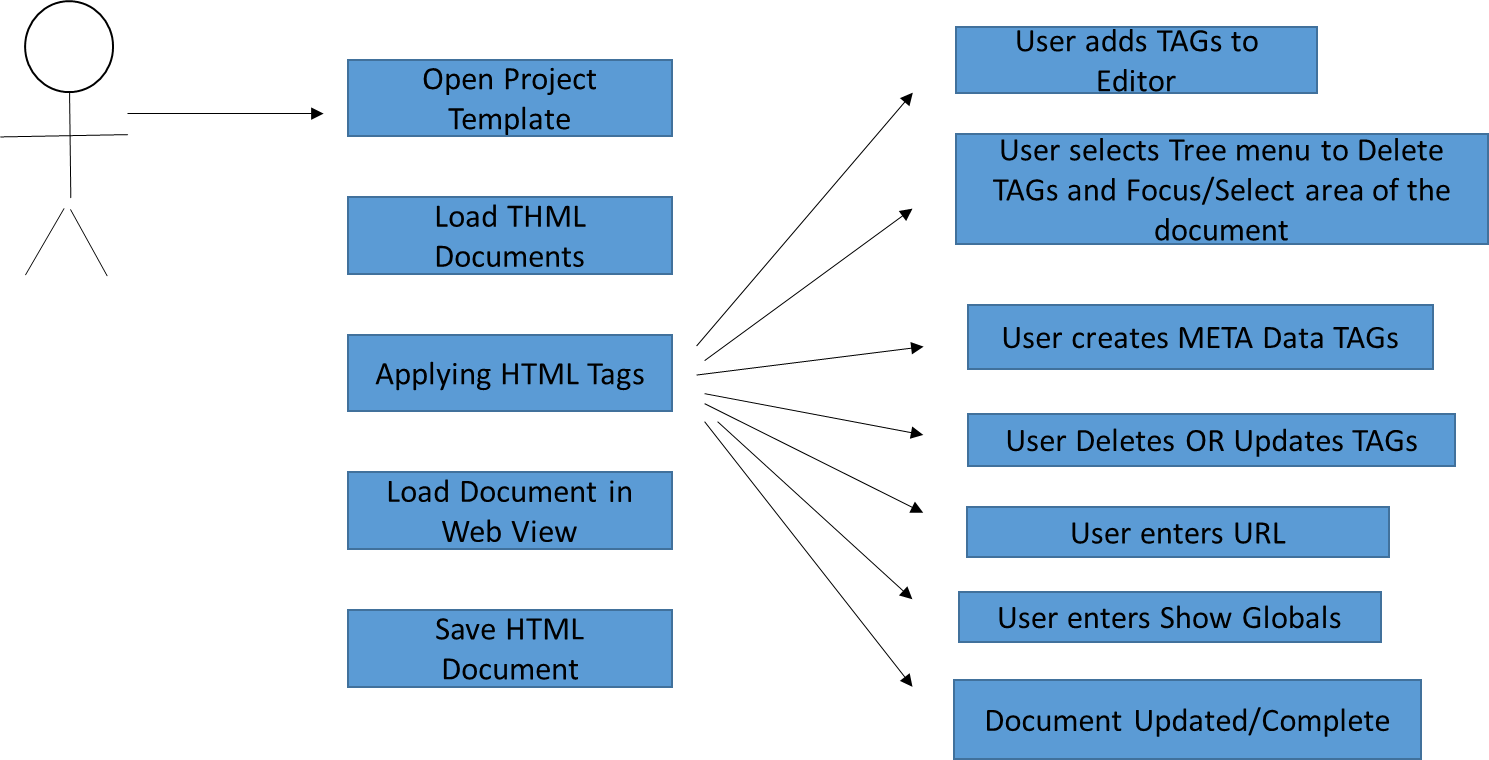


Figure 19. Communication Diagram

## 5.3 Class Diagram

There are eight classes created for this application so far. Pending additional classes. In addition, this application is projected to utilize a cascade style sheet file.

Java Class Files

1. HTMLShell (Driver Class)
2. ButtonFactory
3. buttonToEditorFilter
4. Atrib
5. attribData
6. HTMLView
7. MenuFormat
8. StructureClass
9. attribData
10. BorderSlideBar
11. CheckBoxFormat
12. Confirm Box
13. Editor Parse
14. GetProjectStage
15. GlobalAttrib
16. loadMap
17. MenuFomat
18. TreeWalk

Cascade Style Sheet

1. HTMLdesign

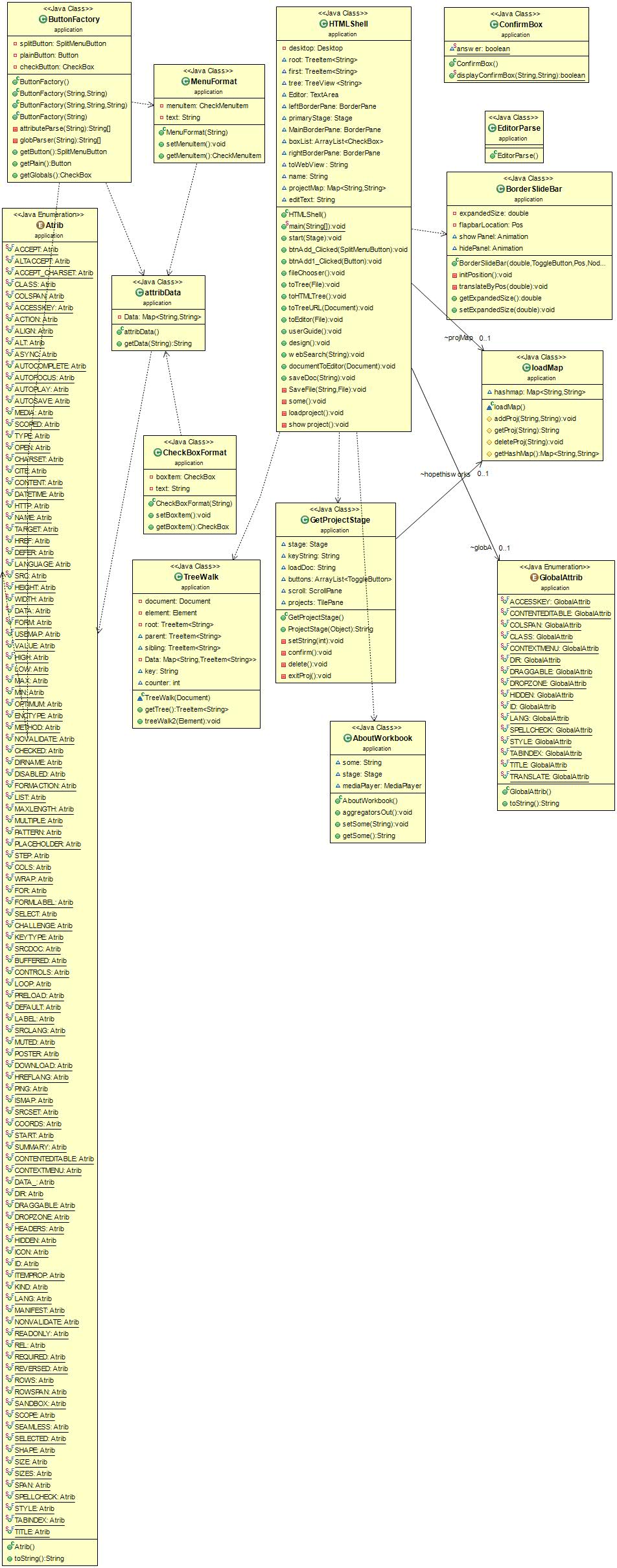


Figure 20. Class Diagram

## 5.4 Basic User Interface

The diagram below, presents the hierarchy of the items managed by the Graphical Users Interface.



Figure 21. Graphical Users Interface.

# 6. Phase 1 Report

## 6.1 Introduction

Phase 1 report will show activities for the week 11/21-11/27. Our phase 1 report, contains the following:

* Team member’s role and task for the phase 1 week
* Original expected and current schedule.
* Problems encountered
* Reevaluation
* Goals for Phase 2
* Document Updates

## 6.2 Roles and Task for Phase 1

|  |  |  |
| --- | --- | --- |
| **ROLE** | **NAME** | **This Week’s Activity** |
| Project Manager | Cesar Lopez | Coded all the buttons to display in the corresponding tabs and made the attributes display for the corresponding tag button |
| Lead Project Designer | David Chew | Reviewed Project Plan. Initiated the Phase 1 Report. |
| Lead User’s Guide | Steve Dickinson | Reviewed code. Updated User Guide to reflect GUI changes. |
| Lead Test Planner | Vijay Patel | Reviewed Test Plan. Planning to start testing as soon as targeted Phase I functionality code completes. |
| Lead Coder | Chris Victores | Worked on coding for the application and pulling external jars from a repository. |

## 6.3 Original and Current Schedule

In this section, we will discuss the original schedule for the application. Then will discuss our current schedule and if our progress is on schedule.

### 6.3.1 Original Schedule for Phase 1

The original milestones for our application is displayed on the table in section 2.1.2. This table displays that:

* Phase 1 report will be completed by 11/25.
* Peer Review 2 will be completed by 11/28

### 6.3.2 Current Schedule for Phase 1

Currently, we are on schedule and will submit the phase 1 report by the anticipated date. The peer review 2 schedule is also doing well. Peer review 2 will be submitted before the expected date.

Overall, we are on schedule with the entire project. We have a Graphical User’s Interface with better characteristic. Thus, we can make a more comprehensive user’s guide.

## 6.4 Problems Encountered

**Cesar Lopez**

I encountered two problems this week. As mentioned before, my task was to place buttons and split menu buttons in the correct tabs with the corresponding attributes. My first problem came when the split buttons were not displaying in the tabs. I had no problems with the first tab, “HTML Structure”. All the split buttons displayed for the first tab. It turned out, the problem was not with the actual split menu buttons, but with the attributes associated with them. I forgot to update the ENUM class that dealt with this. I was inserting attributes the application did not know. After I updated the ENUM class, all the split menu buttons displayed as intended. Figure14 below, illustrates the tabs, buttons, and split menu buttons.

The other problem I came across was the layout. Tabs like the “Common” tab, had many buttons with little space for them to be in. I had to widen the window to an inefficient width for all the buttons to display. Figure 22, illustrates the space problem.

Tabs



Figure 14: Tabs, buttons, and split menu buttons

Split Menu Button

Button



Figure 22: Button spatial issue

**David Chew**

I have not experienced any problems this week.

**Steve Dickinson**

I experienced minor issues getting the code to run on my machine. I found two lines on ButtonFactory.java that reference the CheckMenuItem class which has no code yet. I commented out the two statements found on line 56 “CheckMenuItem item = menuFormatObj.getMenuItem();” and 58 “splitButton.getItems().add(item);” in the file ButtonFactory.java and the program ran without errors.

**Vijay Patel**

I have not experienced any issue this week.

**Chris Victores**

I had some issues migrating the project build so we can pull external jars from a repository like GitHub or SourceForge. This is required for our need in a quality parser for our documents and editor. We looked at the HTMLEditor library in the swing package per teacher hint instead of creating our own parser from the ground up, which was proving to be quite a task, and was giving me an ulcer. We also found that swing is compatible with FX and vice versa. We found the XPATH language and DOM parsers to be more suitable with integration. I could not get GRADLE to work in the environment, but we were able to get Maven working correctly, which is more heavily used in industry anyway, but less modern than the UBER cool GRADLE. Now we can import DOM4J jars, have 1 less heartache, and some more marketable skills. We settled on using XPATH type parsers for future coding editor issues.

## 6.5 Reevaluation of All Decision Made till Now

### 6.5.1 Decision on Roles

The roles we decided to assign everyone is going well. Everyone is working on what they are supposed to do. Steve always does his work quick and early. David always contributes with good designs and shell documents for tasks due in the week. Vijay’s work is always very detailed. Chris is an awesome coder. He always tackles the most complex task in the code required for our application.

### 6.5.2 Communications

As a group, we decided to have a video conference every Monday at 7pm. We discuss:

* Update on what we have been working on.
* Tasks that need to be done by the end of the week
* Assign the task to the appropriate team member.
* Share ideas

We also communicate via text message on google hangout. Everyone updates their progress on google hangout. This communication decision has proven to be effective.

### 6.5.3 Sharing Documents

Our team program to use google drive to share documents with each other. After a shell document is uploaded, everyone updates it with their portion. The project lead, then puts everyone’s update in a single file and uploads it in a Folder named “Doc for Submission”. The team turns in the doc in this folder to the assignment folder in LEO.

## 6.6 Goals for phase 2.

By phase 2, we plan to have:

* Functioning tree in the user’s interface.
* Update the User’s Guide reflecting the updates made in the user’s interface.
* Revisit the test plan chapter for updates.
* Update the UML diagram to reflect the updates in the code.

## 6.7 Document Updates

* Chapter 5 section 5.3 was revised with an updated UML diagram.
* Chapter was updated with revised user’s guide.

# 7. Phase 2 Report

## 7.1 Introduction

Phase 2 report will show activities for the week 11/28-12/04. Our phase 2 report, contains the following:

* Team member’s role and task for the phase 2 week
* Original expected and current schedule.
* Problems encountered
* Reevaluation
* Goals for Phase 3
* Document Updates

## 7.2 Roles and Task for Phase 2

|  |  |  |
| --- | --- | --- |
| **ROLE** | **NAME** | **This Week’s Activity** |
| Project Manager | Cesar Lopez | Work on phase 2 report. Figure out how to code javaFX toggle button to add global attributes for application. Assign work to proper team members. Review the Project documentation before submitting. |
| Lead Project Designer | David Chew | Reviewed Project Plan. Reviewed functionality of  the application. |
| Lead User’s Guide | Steve Dickinson | Rearranged Tab/tag structure, and GUI default opening size |
| Lead Test Planner | Vijay Patel | Review test plan chapter for improvements. Researched HTML tag description for application. |
| Lead Coder | Chris Victores | Menu bar features, Opening and Saving documents, Handling file conversions. Parsing documents. Web and net capabilities. |

## 7.3 Original and Current Schedule

In this section, we will discuss the original schedule and what we planned to do this week. Then will discuss our current schedule and if our progress is on schedule.

### 7.3.1 Original Schedule for Phase 2

The original milestones for our application is displayed on the table in Chapter 2 section 2.1.2. This table indicates that we expected to have phase 2 complete and submitted by December 5. By this week, we originally expected to have a functioning graphical user interface with buttons, tabs, and split menu buttons.

### 7.3.2 Current Schedule for Phase 2

We are on schedule with the original schedule. Our phase 2 report will be completed by 12/3 and submitted by 12/4. The GUI is coming along well. The GUI is where the team expected it to be by this week.

## 7.4 Problems Encountered

**Cesar Lopez**

This week I worked on displaying the global attributes in the GUI. I had to think of how to do that. I did good amount of research. The goal was to create a toggle button. If the toggle button was selected, then a VBox will appear containing all the global attributes in check box format. When the button is deselected, the VBox would disappear. I found a person did something close to that online. I was able to change his logic to fit my needs. I took me long time to get this done, but I did it. I now have another problem. I need to figure out how to add the selected global attribute to the desired tag. I tried adding an extra class, but still have the same problem. The methods that inputs the attribute into the tag, in the split menu buttons, take a button as a constructor. A possible solution, is to create a button in the VBox containing the global attributes and use that somehow. Still thinking.

**Steve Dickinson:**

The goal for me is to update the GUI to resize dynamically as the window is resized. I have looked at doing this through JavaFX and CSS. So far, all my efforts have not been successful. I attempted to modify the CSS with VW which is supposed to resize based on the viewport. I don’t think it is compatible with JavaFX. However, I found a snippet of code that reads the size the users monitor and adjusts the size to open in full screen. This helps with not having buttons scrunched up. Additionally, there are some issue with reading the user guide from the help menu.

**Chris Victores**

The idea was to use XPATH parser because it has great future capabilities for sorting and future features for the editor. However, the web has its own parsers like JSOUP and NEKOHTML to correct malformed documents. Several tags such as <meta> are simply inserted with no closing tag. Parsing these documents has issues with our tree sorting algorithm because it has to be correctly formatted. Thus, more parsing jars must be injected into the program to handle malformed documents. There also seems to be some header issues when saving from one file to the other. The java text area as expected from week 2 is a piece of crap when handling large documents. That’s why I kept saying please, please somebody look at rich text area library, please study this like bible study the day before the apocalypse, and was promoting it like a girl scout cookie seller that needs to sale 10 more boxes to get that Hershey park field trip. Please with sugar on top this will help us later, but it has like 1000 classes and is all over the place. We might be able to be speed up with the font, caret, and putting basic text, but likely we just need to find a better text processor.

**David Chew**

I did not encounter any problems this week.

**Vijay Patel**

I encountered two problems this week. First problem was setting up the HTML editor application on my laptop. Since my team was using mostly eclipse, I had to install the eclipse IDE to my laptop. Here is where I ran into my first issue, my computer was very slow. It was taking an unreasonable amount of time to perform any tasks. I simply went to the task manager and disabled some of my start up application, then restarted my laptop. After that, Eclipse IDE was installed without any problem and could install and run the application without any problems.

The second challenge, was implementing tooltip for each button, representing a html tag, in our application. The goal here, was to make the description of the tag, the button represented, display when the mouse cursor is over it. I could display a static message, meaning, all the button had the same description. However, displaying dynamic message was challenging. I wanted the description to change to the appropriate description. Unfortunately, I was not able to resolve this conundrum. I will continue to work on it.

## 7.5 Reevaluations

### 7.5.1 Graphical User Interface Layout

Originally, we planned to have 8 tabs on the interface. Each tab has buttons or split menu buttons. Chris used cascade style sheet (CSS) to format the display of the GUI. We reevaluated some of these decisions:

* We are going to make appropriate extra tabs to deal is the number of buttons within the tabs.
* We are going change the CSS code for better layout for the GUI
* We are discussing whether to add a scroll pane to the HBox containing the buttons.

### 7.5.2 Tag Description using Mouse Over Action Event

Initially, we decided to use tool tip for the user to see the description of the desired tag. After some coding, done by lead coder, we concluded that it is best to use mouse over action event for tags and tool tip for attributes.

### 7.5.3 Placing Phase Report in Main Documentation

During the phase 1 report, we decided to make the report separate from the main doc. After some reevaluation, we think it is best to make each report a chapter in the main documentation for the project. During the phase 2 report week, we added phase 1 and phase 2 as part of the project documentation.

### 7.5.4 Updating User’s Guide and UML diagram.

Our team has reached a pivotal point in the project. This week, almost everyone did some coding. Updates were made to the code. We are in the middle of deciding whether to change the layout with CSS. The user’s guide may look completely different next week because of the amount of features the application will have by then. Thus, the team decided to wait till next week to update the User Guide.

There is a lot of code rearranging going on right now. Some classes were erased and some added. During the phase 3 week, all classes should be final. We believe it’s more time efficient to updated the UML diagram next week.

## 7.6 Goals for Phase 3

For phase3, we plan to:

* Implement Live View feature for our application.
* Put message boxes for all the catch statements.
* Fix any layout issues.
* Update user guide and test plan.
* Add and get project feature
* Web View
* XPath delete

## 7.7 Document Updates

During this phase, the following documents were updated:

* Chapter 6 and 7 were added.

# 8. Phase 3 Report

## 8.1 Introduction

Phase 3 report will show activities for the week 12/05-12/11. Our phase 3 report, contains the following:

* Team member’s role and task for the phase 2 week
* Original expected and current schedule.
* Problems encountered
* Reevaluation
* Goals for the final
* Document Updates

## 8.2 Roles and Task for Phase 3

|  |  |  |
| --- | --- | --- |
| **ROLE** | **NAME** | **This Week’s Activity** |
| Project Manager | Cesar Lopez | Worked on phase 3 report. Added code that allowed the global attributes to display within the tag the user selected. Added a confirm box to make sure the program is closed properly and the user saved their work. Assign work to proper team members. Review the Project documentation before submitting. Lend a hand when needed |
| Lead Project Designer | David Chew | Updated the Design Plan due to the additional feature was implemented. |
| Lead User’s Guide | Steve Dickinson | Updated users guide to reflect GUI changes and additional functionality. Added Web View functionality to preview the HTML code. |
| Lead Test Planner | Vijay Patel | Updated Test plan to reflect GUI changes and additional functionality. Added tooltip functionality for all tag option button. Performed functionality testing and reported two issues to the team about application errors. |
| Lead Coder | Chris Victores | Input new serialization feature, worked on menu Items and CSS. |

## 8.3 Original and Current Schedule

In this section, we will discuss the original schedule and what we planned to do this week. Then will discuss our current schedule and if our progress is on schedule.

### 8.3.1 Original Schedule for Phase 2

The original milestones for our application is displayed on the table in Chapter 2 section 2.1.2. This table indicates that we expected to have phase 3 complete and submitted by December 9. By this week, we originally expected to have all the Graphical User Interface’s functionalities working properly.

### 8.3.2 Current Schedule for Phase 2

We are on schedule with the original schedule. Our phase 3 report will be completed by 12/8 and submitted by 12/9. The GUI is almost complete with our original goals.

## 8.4 Problems Encountered

**Cesar Lopez**

The problem I had this week was getting the global attributes to display inside the tag the user selected. My first try at this, was creating a method that took a string as an argument, getGlobalSelected(String globals) and make a switch statement with each global as a case. If that case was selected, then global = that global. As I was doing this, I knew it was not an efficient way to solve my issue. This way did not work. I wanted to put the checkboxes in an observable list. That way, I can just iterate over the whole list and have it display if a global was selected. That wasn’t working either. Chris told me to try ArrayList<CheckBox>. I couldn’t believe I didn’t think of something so simply. Chris’ advice worked and the rest was downhill. I finished my coding task for the week.

**Steve Dickinson**

This week my two main goals were to add a Web View capabilities to the HTML Editor and updating the User’s Guide. Updating the User’s Guide is straight forward. The only limitation is ensuring I have the most up to date code. This team likes to code and if you wait five minutes the version you have will not be the most current. ☺ Adding the Web View capability took some time trying to decide the best place for the Web View button. I first tried placing it in the menu bar but it just didn’t seem right so we ended up moving it to the bottom. The Web View is somewhat limited unfortunately since when you import the HTML from a URL you get strict HTML without any CSS styling so it doesn’t always look how you might expect.

**Chris Victores**

Going into this week I was having some difficulty with the serialization issue, and also prioritizing what we need for a functional product. There is still a ton of stuff I want to do like navigating the document, and better editor functionality. Steve is in strong opposition to the bikini girls in the new project feature, and needs to understand why products sell--bikini girls, duh, look at any magazine that’s still in business. He even deleted it from the user manual, this kind of prude attitude is really bringing morale down, and bringing the Joseph Stalin/Adolf hitler type politically correct discipline into JavaFX stage management. It really dampens the marketing prospects.

**Vijay Patel**

This week my main goals were to add dynamic tooltip display on that all tag buttons and integrated latest code with my existing code; basically, get latest code from the developers and run the testing on functionality develop till phase 3. When I was integrating latest code from the Google drive I faced some problem. After taking my existing code when I clean up and load new jar file as instructed and build application it started giving errors in the all the classes. To fix this issue I took help from Cesar and after trying some option we figure out that my backup application folder setting in same workspace that is causing this issues. Then we cleanup everything from work space and re-installed application and it started working fine. While I was testing application, in some scenario application was caressing I not able to fine at what class has problem then debug it and found java classes and line numbers. I reported this issue to and specific scenario to team to catch exception.

**David Chew**

Additional features were added to the application. Therefore, the design plan was modified to reflect the application. The communication diagram has “User enters URL” and “User enters Globals” Try to assist coder by incorporate an Alert Box classes within Try Catch, however was not successful.

## 8.5 Reevaluations

### 8.5.1 HTML Editor Final Functionalities

We discussed how we will proceed with the final two weeks of the class. The team decided that by the end of phase 3 week, the application should have nearly all the complex features coded. This is to avoid working on time consuming code in the final week.

### 8.5.2 The Finish Line/Clean Up Week

Taking care of all the complex task on phase 3 week, team Aggrevators, will use the final week to clean everything up. The team will work on:

* Refining the project’s main documentation
* Cleaning up the code (erasing redundant commenting)
* Adding java doc
* Clean the try and catch statements.

Team Aggrevators will finish strong.

### 8.5.3 Java Classes

The coders of the team added the all tabs, buttons, and split menu buttons in the driver class. Thus, the driver class has a lot of code. The coding lead, Chris, has a great idea. We will make the tabs its own classes and put the button factory as a super class.

## 8.6 Goals for Final Week

By the end of the final week, we plan to:

* Get XPath parsers to provide functional features like delete and focus
* Work on the Web View function to handle errors and format headers.
* Put the tabs in their own classes where button factory is a superclass.
* Need to add some CSS code for global attributes.
* Add the results from functional tests
* add a search ctrl/f type feature.
* Complete Application.

## 8.7 Updated Documents

The following documents were updated:

* Chapter 3 user’s guide
* Chapter 4 section 4.3 (table)
* Chapter 5 section 5.3 was revised with an updated UML diagram.
* Chapter 8, phase 3 report, was added.

# 9. Overview

## 9.1 Team Members’ Contributions

|  |  |  |
| --- | --- | --- |
| **ROLE** | **NAME** | **Overall Contribution** |
| Project Lead | Cesar Lopez | * Setting up communication tools for the team. * Schedule conferences for team. * Code buttons, split menu buttons, and tabs for the application. * Code event handlers to deal with singleton tags. * Created Enum class for global attributes * Coded sliding bar panel for the global attributes and toggle button to activate it. * Wrote all Phase reports. * Merge everyone’s doc |
| Lead Coder | Chris Victores | * Used JavaFX for project code * Integrated CSS in project code * Used XML and HTML parsers for coding features * Utilized tools like GitHub and Maven for code management * Implemented recursion and algorithms for document navigation * Used io tools for load, save, and file management * Implemented Serialization features * Added to the “Problem encountered” sections to all Phase reports |
| Lead User’s Guide | Steve Dickinson | * Updated User’s guide to reflect updated code every week. * Organized the buttons, split menu button, and tabs for greater efficient layout. * Added to the “Problem encountered” sections to all Phase reports. * Coded the web view logic and added the button to the GUI. * Help code to fix a pdf issue * Added code to read screen size and open GUI to corresponding size. * Helped with tables for test plan |
| Lead Test Planner | Vijay Patel | * Created Chapter 4, test plan * Ran test and documented results. * Researched the description of all HTML5 tags and added them to a hash map * Used hash map to utilized tool tip feature to dynamically display descriptions * Added to the “Problem encountered” sections to all Phase reports. |
| Lead Project Designer | David Chew | * Proof read document * Added outline for Chapter 5. * Added to the “Problem encountered” sections to all Phase reports |

## 9.2 Lessons Learned

**Cesar Lopez**

The experiences I have obtain in this class were amazing. At first, I was really scared that the work asked from me was going to be out my league. It even got worse when I became project lead. I was so scared to mess up. I guess the fear of letting my teammates down drove me to work harder and not wait to the last minute to get stuff done. Our team decided to make an HTML editor. I thought we were going to use java swing. I am proficient with java swing. Then I found out, we are using javaFX, I just went with it and said “challenge accepted”.

I was completely lost when I first started coding with javaFX. I didn’t know anything. I kept doing research and practiced a lot. Some weekend, I slept for a total of 2 hours. At the end, it was worth it because I learned to use javaFX and it is now a part of my coding arsenal. I learned about checkboxes, observable list, and javaFX’s layout managers. It has VBox, HBox, BorderPane, AncherPane, and more. I was happy to find out that the nested layout managers work the same as in java swing.

My favorite part of this project, was that I we were not constraint to certain parameters. In other courses, I had to email the professor and ask if I can add this class or if I can use a hash map instead of something else. In this project, if I wanted to make a new class, I made it.

This journey made me realize that you do not have to be a genius coder, you just must be persistent and not quit until you get it. I also learned the value of team work. Almost everyone’s contribution made this possible.

**Chris Victores**

First and foremost, I learned about how to better handle requirements planning for a project. Given time constraints, a macro and micro perspective should be taken in requirements. From the macro side an estimation of features should be assessed given a time period. From the micro side, each individual detail needs to be planned. For example, parsing an HTML tag has several things to consider by feature, and by user need. If the objective for a feature was to have a tree for all the tags in an HTML document, several sub features need to be considered for this macro perspective. Some of these questions include:

* How will the tags be parsed?
* How will tokens and chars be used to register when a tag is completed?
* How will the program disable the user from breaking the code?

All these questions bring even further detail and micro requirements to the programming feature.

Our teammate, Cesar, has a poster accentuating the problem it reads:

"99 bugs in the code,

99 bugs in the code,

you take one down and patch it around,

127 bugs on the wall.”



These perspectives and requirement analysis come with experience, and are beneficial in realizing in a customer client relationship. As a coder, it is often ambitious to think of some ideas, and say you will implement them, but the requirements for those issues often require some rational meetings on what all needs to happen, and if this is a bigger priority then other issues. Often the requirements for something very small require a lot of throughput, and an easier solution will often save several hours or days. Change management and receptiveness to not being stubborn and changing a feature to something more reasonable is often the best choice. Other things I learned about were the complexity of development operations. The start of a project often sets the tone of how things will be manage, and the initial design of the project needs to have some time and effort to ensure things are done the right way. Teammates need to be flexible and willing to change accordingly, and positions and roles should not be used as an excuse to deter work in any organization. Individuals need to change according to the project need for effective projects, and that is why I think RAD and AGILE are so common in the industry.

**Steve Dickinson**

This was an exciting class that presented many challenges. It was incredible to see how quickly that application grew. Our lead coder, Chris, was constantly addressing new additions that we necessary to run the HTML Editor effectively. He used a variety of additional functionality like Maven and we had to import quite a few .jar files to run the program correctly. So lesson 1 was this. When you think up an idea it will most likely grow bigger and force you to go beyond just the languages you think you will need.

Typically, when I start to try and figure out how to change already existing code I spend a vast amount of time studying the code to figure out what is going on. This application was no different. I struggled early on to understand what was going on and had to move between classes to figure out if I was looking at a method from the API library or something that was created for this application. It’s a very overwhelming feeling trying to walk through and figure out how the code functions and often had to revisit the classes and relearn their functions. So lesson 2 is this. When trying to modify existing code expect to be completely lost when first learning that particular application. I think it is probably normal to feel lost and learn to push through it and know eventually I will reach a point where I understand the code.

I learned quite a bit about the difficulty in keeping files from getting mixed up. The code changes rapidly and trying to get the most current code to run was tough as often the IDE froze up or I forgot to import the additional .jar files needed. From here I am going to learn GitHub.

Much of what I learned was relearning how classes are used to communicate with each other. It was important to be reminded to create code this way so it is clean and modular.

Finally, I was reminded the importance of asking questions. Sitting down with our team lead, Cesar, while he explained many of the functions I was able to understand the code much quicker than by simply sitting and going through it by myself.

**Vijay Patel**

Working with remote team environment was challenge to me and other team members as well when it came to communication and coordination However Project planning and define project scope, break downs of our weekly goals and weekly task allocation to each team members help us to complete this project in time with in the project scope. I used to with Google Hangouts but never used GitHub, in this project we used both Google Hangouts and GitHub efficiently to stay in touch and track our progress. I am glad to say that having a weekly scheduled video/voice chat help us improved the process and manage our project.

I think effective requirement analysis is key for my phases. Since this project is not driven by customer’s requirement we are not able to define all requirements in beginning phase. If we had defined all requirement in initial phase, then we can develop this application with more future. Before this project I don't know about JavaFX this project taught me how to use JavaFX in application development. This project taught me how to implement IT solutions that comply with specifications, requirements, standards, and guidelines. After completing this project, I learned how to plan, conduct research, and complete a collaborative, challenging, computer-related project in compliance with schedule deadlines. This project work allowed me to work in team to design, develop, implement and document project documentation.

**Dave Chew**

I realized having the design plan in the beginning of the project is very important as I understood later in the project phase. In this project, as a novice Java coder, I felt working with JavaFX seemed having to learning another programing language which was learning curve for me. In this project, realizing each member have different skills to provide to complete the project. Design and work together in a team is challenging since the backgrounds and interests of members are different. Our team’s communication was accomplished via “Google Hangout” which was much more efficient and effective way to communicate with all team members than the conventional weekly meeting at a location.

## 9.3 Design Alternative

### 9.3.1 Java Swing

This application was build using JavaFX. An alternative is Java Swing. If Java Swing was used, border layout, grid layout, and JComponents would have been utilized instead of the JavaFX equivalents.

### 9.3.2 CSS Inline

This application used an external cascade style sheet to manipulate the presentation of the GUI. An alternative, would be using inline style. Instead of have a separate CSS file, we could have used CSS with in the object being controlled by it. But then we would have to use it for every java file. It is easier to deal with one file that effects all the files. Thus, external is more efficient.

## 

## 9.4 Design Strengths

The strengths of the HTML Editor are:

* The use of cascade style sheets to manipulate the presentation and layout of the GUI.
* Tool tips to display button descriptions
* URL web search feature
* Tree View to navigate the document
* User friendly
* The theme used minimizes cluttered data and components.

## 

## 9.5 Limitations

### 9.5.1 Multiple HTML Files

often editors have options for opening multiple files, and the ability to navigate from document to document. Currently our editor does not allow this feature. The addition of allowing users to navigate from one text area to the next is a limitation vs. competition.

### 9.5.2 Images for Web View Feature

The web view feature is a feature that lets the user click a button that will display the web view of the code in the editor. In the case where there is code of an existing webpage in the text area, the web view will not display the images in the page because that image file is not in the directory of the project. For example, if the code has “<img href=”blahblah.PNG>”, the image will not appear because the blahblah.PNG file is not in the system where the HTML Editor is running.

## 9.6 Suggestions for Future improvement

### 9.6.1 Cascade Style Sheets

Incorporate cascade style sheet (CSS) in the files would be a great idea. If the user uses an attribute with external cascade style sheets in mind, he or she can press a CSS button. This button will command a pop up window will appear with the CSS file. That way the user can add or change design for their HTML code.

### 9.6.2 Tag Validation

Some of the tags need validations like the table tag. For example, if table data (<td>) is entered before the table head (<th>), the HTML editor will not validate that mistake. It would be a good coding exercise to figure out a solution to this.