Software Requirements Specification (SRS)

X-Browser Editor

Team: 2

Authors: Tarpan Patel, Joshua Michaud, Ben Viscosi, Marcello Barbieri,

Bill Figliolini

Customer: Front-End Developers

Instructor: Dr. James Daly

# Introduction

X-Browser Editor is a web development tool created to streamline web development testing. This initial section will describe this document’s purpose, the scope of the document, define the definitions to be used throughout the rest of the document, and provide a description of the following sections.

## Purpose

This document is the definitive set of requirements for the X-Browser Editor and will help clarify these requirements between the developers, the customer, and the instructor. The intended audience is the development team of the Editor, as well as any users of the Editor who want to understand why the current design was chosen.

## Scope

X-Browser Editor will attempt to streamline the process of website development. Currently with the differing implementations of HTML, CSS, and JavaScript implementations between browsers, testing if one’s current website implementation functions and is consistent across platforms can be time consuming. By directly connecting the browsers to the editor, X-Browser Editor seeks to simplify this process and provide real-time feedback to the developer.

## Definitions, acronyms, and abbreviations

Editor: The X-Browser Editor

User: The website developer using the Editor

JS: JavaScript

Website: the interpreted result of HTML, CSS, and JavaScript code

Browser: The Web browser which the Website will be run on to test its compatibility

Files: Any set of HTML, CS, or JS files.

## Organization

The remainder of the SRS is organized into the following 6 sections:

**Section 2** : This section is an overview of the Editor as well as an overview of the Editor’s functions, user expectations, constraints, and dependencies.

**Section 3**: A formal list of the software’s requirements

**Section 4**: Provides the expected use cases of the Editor, its classes and how they relate, and finally the Editor’s sequence diagrams.

**Section 5**: This will provide an instruction manual for the Editor’s prototype, as well as a link to the prototype.

**Section 6**: A list of all referenced works for the document.

**Section 7**: Contact information for the instructor for questions regarding the Editor.

# Overall Description

This section is intended to cover all the assumptions surrounding the User and the underlying the design of the Editor. It will cover the perspective the developers have taken for the development of the Editor, it’s intended functions, and the expected abilities of the User. It then moves on to the Editor itself, describing the constraints to the Editor, the assumptions and dependencies made during development, and finally the features which have been determined to be outside the Editor’s scope.

# Product Perspective

The Editor is to serve as an independent development environment for HTML, CSS, and JS. By integrating browsers directly into the development environment, it aims to streamline testing and allow the user to see the effects of their changes and if they are consistent across the range of browsers which the User has selected for testing. This will allow for the User to correct any code which results in inconsistencies rapidly, avoiding having to send the website to a third party to verify if it correctly functions on multiple browsers. To maintain easy usability of the code, we have opted to design this editor as a web app, allowing it to be portable across multiple devices.

# Product Functions

The Editor will help streamline website development by allowing the user to see the effects of their changes to their website on multiple browsers in real-time. This will reduce overhead in development, and reduce time being wasted manually verifying if a new HTML, CSS, or JS feature has been adopted across all bowsers.

# User Characteristics

The user is expected to be a web developer with experience in HTML, CSS, or JS. They will be expected to have some knowledge of how to navigate a common IDE, as well as a browser.

# Constraints

The User will be accessing the Editor through a browser, so usage of new and possibly unsupported HTML, CSS, or JS features need to be avoided to maintain general usability.

# Assumptions and Dependencies

The Editor will be based on a Web App, requiring an active internet connection and a working browser.

The Editor will require permissions to save the created Files to the User’s machine.

The Editor will require the user to upload any already existing Files for the Editor to display.

The user is expected to understand at least one of HTML, CSS, or JS for the purposes of using the Editor.

# Apportioning of Requirements

The initial release of the Editor will be limited to displaying the resulting code, leaving the actual validation to the User. In a future update, the Editor may attempt to include automated testing of this, further streamlining development.

# Specific Requirements

1. Webapp Layout
   1. The Editor will have real-time rendering of HTML file and associated CSS and JS files
   2. The Editor will be able to connect and render multiple browsers simultaneously
   3. The Editor will have a means of displaying each browser
2. Code Editor
   1. The Editor will accept user inputted HTML, CSS, and JS Code to make the Website
   2. The Editor will prepare the code for rendering by the Browsers
   3. The Editor will allow the user to save any Files created in the Editor to their computer.
   4. The Editor will allow the User to open any Files that already exist.
3. Virtualized Browsers
   1. The Editor will accept a list of Browsers to test
   2. The Editor will display the user’s Website on all the Browsers which have been selected

# Modeling Requirements

## 4.1 Use Case Diagram

The XBrowser use case diagram is single user application which focuses on meeting user’s expectations as they interact on the website. The use case diagram starts by having the user uploading files and adding browsers to test their work. As the user saves their work, many features are executed to render the layout on the selected browsers corresponding to the contents on the files. Please see below for use case diagram and definitions below:

Diagram

Description automatically generated

|  |  |
| --- | --- |
| Use Case Name: | Upload Code |
| Actors: | User |
| Description: | The user uploads files that they want to edit. |
| Type: | Primary |
| Includes: | None |
| Extends: | None |
| Cross-refs: | Requirement 2.1 |
| Uses cases: | None |

|  |  |
| --- | --- |
| Use Case Name: | Open Virtual Browsers |
| Actors: | User |
| Description: | The user can choose what browsers to view their results |
| Type: | Primary |
| Includes: | Web App Layout |
| Extends: | None |
| Cross-refs: | Requirement 3.1, 3.2 |
| Uses cases: | Load default virtual browsers (Firefox, Chrome, Safari) |

|  |  |
| --- | --- |
| Use Case Name: | Edit Code |
| Actors: | User |
| Description: | The user edits their code. |
| Type: | Primary |
| Includes: | None |
| Extends: | None |
| Cross-refs: | Requirement 2.1, 2.2 |
| Uses cases: | The code files must be html, CSS, or JS. |

|  |  |
| --- | --- |
| Use Case Name: | Save Code |
| Actors: | User |
| Description: | Code is saved. |
| Type: | Primary |
| Includes: | Update Virtual Browsers, Styling, PostCSS, Linting |
| Extends: | None |
| Cross-refs: | Requirement 2.1 |
| Uses cases: | Code must have changed since last save. |

|  |  |
| --- | --- |
| Use Case Name: | Update Virtual Browsers |
| Actors: | User |
| Description: | Code output is displayed in virtual browsers |
| Type: | Secondary |
| Includes: | Web App Layout |
| Extends: | None |
| Cross-refs: | Requirement 1.3, 3.2 |
| Uses cases: | The browsers must be displayed before updating. |

|  |  |
| --- | --- |
| Use Case Name: | PostCSS |
| Actors: | Program |
| Description: | Run code through PostCSS. |
| Type: | Secondary |
| Includes: | None |
| Extends: | None |
| Cross-refs: |  |
| Uses cases: | The user must Save Code. |

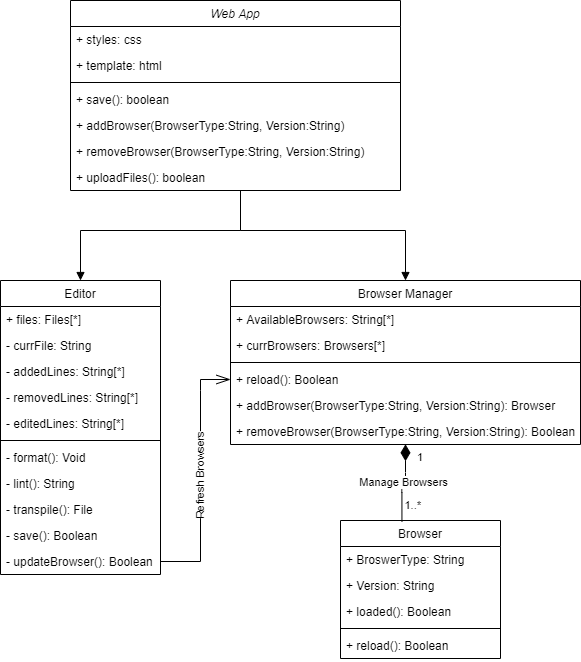
|  |  |
| --- | --- |
| Use Case Name: | Linting |
| Actors: | Program |
| Description: | The user’s code is linted. |
| Type: | Secondary |
| Includes: | None |
| Extends: | None |
| Cross-refs: |  |
| Uses cases: | The user must Save Code. |

|  |  |
| --- | --- |
| Use Case Name: | Linting |
| Actors: | Program |
| Description: | The user’s code is linted. |
| Type: | Secondary |
| Includes: | None |
| Extends: | None |
| Cross-refs: |  |
| Uses cases: | The user must Save Code. |

|  |  |
| --- | --- |
| Use Case Name: | Styling |
| Actors: | Program |
| Description: | The user’s code is styled. |
| Type: | Secondary |
| Includes: | None |
| Extends: | None |
| Cross-refs: |  |
| Uses cases: | The user must Save Code. |

## 4.2 Class Diagram

The class diagram below displays the necessary classes and its operations that XBrowser website is built from. Please see the diagram and its data dictionary below.



|  |  |  |
| --- | --- | --- |
| **Element Name** | | **Description** |
| Web App | | The Web App class provides basic operations to set up an XBrowser working environment. |
| **Attributes** |  |  |
|  | Styles : CSS | CSS file required for the styling the website |
|  | Template : html | Html file required for the layout of the website |
| **Operations** |  |  |
|  | uploadFiles() : boolean | Member of class Web App executes when the user selects the option to upload existing files. |
|  | addBrowser(BrowserType:String, Version: String) | This function interacts with Browser Manager class, allowing the user to add a browser given type (i.e. Chrome) and its version. |
|  | removeBrowser( BrowserType:String, Version: String) | Member of class Web App, interacts with Browser Manager class, allowing to remove browser from the view by specifying its type and version. |
|  | save() | The function allows users to save their edited files. |
| **Relationships** | The Web App class has associated Editor and Browser Manager classes and which creates the instances of the two classes. | |

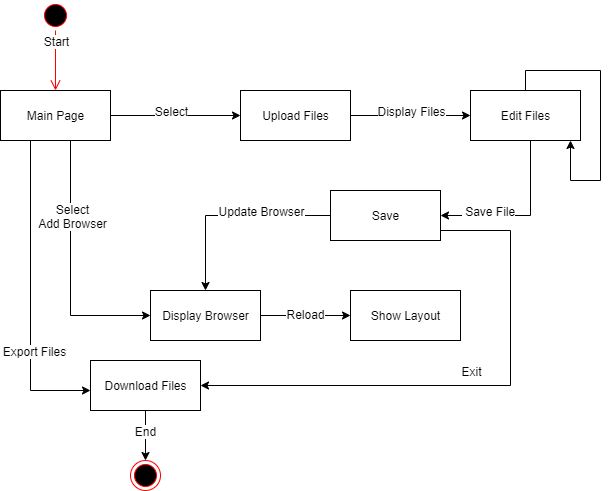
|  |  |  |  |
| --- | --- | --- | --- |
| **Element Name** | | | **Description** |
| Editor | | | The Editor class assists Users on editing files and keeping files error free. |
| **Attributes** |  |  | |
|  | files : Files[\*] | An array provided for uploadFiles method to store information for files. | |
|  | currFile : String | A string that contains the name of the file that is being edited. | |
|  | addedLines : String[\*] | An array of strings containing the new lines of code added to a file(s). | |
|  | removeLines : String[\*] | An array of strings to store lines of code that are removed. | |
|  | editedLines : String[\*] | An array of strings to store lines of code before they are edited. | |
| **Operations** |  |  | |
|  | format(): void | Format function structures the lines of code for readability, understanding and debugging. | |
|  | lint(): String | Lint function helps Users debug files containing errors. | |
|  | transpile(): File | Generates equivalent source to source code in other languages | |
|  | save(): boolean | The function allows users to save their files. | |
|  | updateBrowser(): boolean | Function interacts with Browser Manager class to have the contents updated in the browsers. | |
| **Relationships** | The editor class has an association relationship with Web App class. | | |

|  |  |  |
| --- | --- | --- |
| **Element Name** | | **Description** |
| Browser Manager | | The Browser Manager class fulfils the requests of Users on browser preferences. |
| **Attributes** |  |  |
|  | AvailableBrowsers : String[\*] | An array containing available Browsers for Users to test their code |
|  | currBrowsers : Browsers[\*] | An array that holds browsers the user is currently working with to test their code. |
| **Operations** |  |  |
|  | reload() : boolean | The operation that refreshes the layout on the browser(s) corresponding to edited files. |
|  | addBrowser(BrowserType:String, Version: String) : Browser | Interacting with Browser class, this method adds a browser given type (i.e. Chrome) and its version |
|  | removeBrowser( BrowserType:String, Version: String): boolean | Interacting with Browser class, the method removes selected browser from currBrowsers array given its version and type (i.e. Chrome). |
| **Relationships** | The Browser has an association relationship with Web App class and has composition relationship with Browser class. The Browser Manager updates the browser per Users interaction in Editor class. The Browser manager can have 1 to many browser objects. | |

|  |  |  |
| --- | --- | --- |
| **Element Name** | | **Description** |
| Browser | | The Browser class |
| **Attributes** |  |  |
|  | browser : String | A string that contains the name of the browser. |
|  | version : String | A string that holds the version of the browser. |
|  | Loaded : Boolean | A Boolean value to confirm if the new content is updated on browsers. |
| **Operations** |  |  |
|  | reload() : Boolean | Method updates the layout on the virtual browsers upon save. |
| **Relationships** | The Browser class has a composition relationship of the Browser Manager class. | |

## 4.3 State Diagram

The state diagram demonstrates each stage for Users on how to navigate through XBrowser to set their working environment.

****

## 4.4 Sequence Diagram

The sequence diagram below shows the operations that occur when the user interacts with XBrowser.

Diagram

Description automatically generated

# Prototype

The present Prototype V1 provides a sample of how the web app could be laid out. This will be used to generate feedback as well as help find any glaring flaws in the layout.

# How to Run Prototype

The prototype requires a GitHub account and Yarn installed.

Prototype is accessible online at https://dazzling-tesla-17ff26.netlify.app/

To run your own instance:

1. Install node.js version 10.0 or greater

2. Install Yarn tool

3. Clone XBrowser repository from: https://github.com/marcebdev/XBrowser

4. Open a terminal in the repo directory

5. yarn install

6. yarn generate

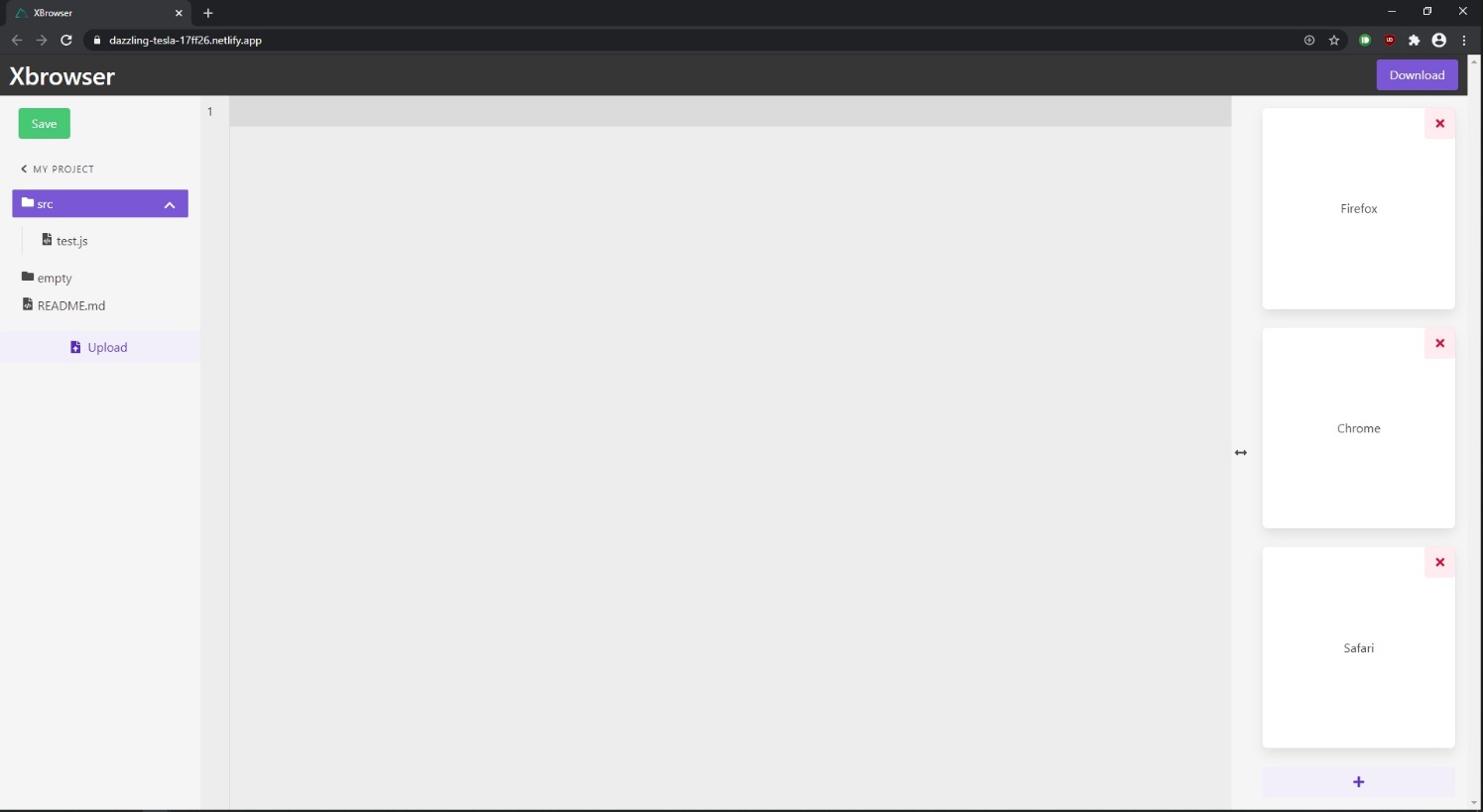
7. yarn start, this will be you to the webpage

8. Go to : <http://localhost:3000/>

Helpful Link installing Yarn: <https://classic.yarnpkg.com/en/docs/install/#debian-stable>

# Sample Scenarios

XBrowser is not a complicated application to use. The user starts off the main page, where they upload files. Once the files are uploaded, the user can select what browsers to test their websites on by providing version and browser type (i.e. Firefox, Chrome etc.). The user can now start editing their code. To view the updated layout on the virtual browsers, the user will have to save the edited files and content in virtual browsers will be refreshed. The user has the option to download files when they are done editing and then exit or start editing other files.



# References

* Provide list of all documents referenced in the SRS
* Identify each document by title, report number, date, and publishing organization.
* Specify the sources from which the references can be obtained.
* Include an entry for your project website.

Start of your text.

# Point of Contact

For further information regarding this document and project, please contact **Prof. Daly** at University of Massachusetts Lowell (james\_daly at uml.edu). All materials in this document have been sanitized for proprietary data. The students and the instructor gratefully acknowledge the participation of our industrial collaborators.