Requirements Document

Anthony Guirguis (guirguia)
Hoda Mohammadi (mohamh8)
Mikolaj Hrycko (hryckom)

October 6, 2015

Contents

List of Tables

List of Figures

Revision History

Table 1: Revision History: Requirements Document

DATE	**DEVELOPER**	**CHANGE**	**REVISION**
October 6, 2015	Anthony Guirguis	Initial Draft	0
October 6, 2015	Hoda Mohammadi	Initial Draft	0
October 6, 2015	Mikolaj Hrycko	Initial Draft	0

1 Project Drivers

1.1 The Purpose of the Project

1.1.1 The User Business or Background of the Project Effort

This project is a reimplementation of a JavaScript math library called MathJS and we intend on going a step further than the original implementation and creating a website that uses our implementation. We plan on allowing users to input derivative, integral and numerical expression problems that can be solved and outputted straight to the user. The original library featured a flexible expression parser with support for symbolic computation and a large set of built-in functions. Our motivation to make this project is to make a place where complex mathematical expressions can be solved with little effort from the user and to offer a user-friendly service instead of just developing an API like MathJS.

1.1.2 Goals of the Project

Our product will allow users to input a variety of mathematical expressions and obtain the answers to these expressions on an elegant, easy to use website. The goals for this project are creating a JavaScript program that implements a library similar to MathJS, documenting the processes involved in making the project and getting a website to host the program. The website is there so users can easily access the program and use it.

1.2 The Stakeholders

1.2.1 The Client

An external entity that will be reviewing the project and is interested in implementing a math expression solver for general as well as specific use.

1.2.2 The Customer

The customer of our project is the general public, and more specifically students, teachers and people working with math expressions daily.

1.2.3 Other Stakeholders

- Members of the public

  Represents the rest of the user base. No extra knowledge or involvement required, due to the easy to use nature of the website.

1.2.4 The Hands-On-Users of the Product

- University Students

  Interacts with website

  Master at navigating the web

  Ability to access completed website

1.2.5 Personas

- Name: Joe Schmo

  Age: 20

  Job: Unemployed

  Family: Jake Cane, Father

  Hobbies: Researching the psychological effects of sleep deprivation

  City: Llanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogoch, United Kingdom

  Favourite Food: Fermented cucumbers

  Favourite Music: Classic Pop

  Likes: Math, Integrals, Computers

  Dislikes: Short city names, sleeping

  Attitude to Technology: Positive

  Attitude to Money: Indifferent

1.2.6 Priorities Assigned to Users

Key Users:
The Customer

Secondary Users:
The General Public

Unimportant Users:
The General Public

1.2.7 User Participation

The Customer
No input necessary.

The General Public:
No input necessary.

1.2.8 Maintenance Users and Service Technicians

Project Maintenance Team
These users will maintain and update the code in the project and the performance of the website.

2 Project Constraints

2.1 Mandated Constraints

2.1.1 Solution Constraints

Description: The product shall operate on computer browsers that allow JavaScript to run through a website hosting server.

Rationale: The user will be using a desktop/laptop and browser that is not outdated to access our site.

Fit criterion: Our website will be usable to users with browsers that have an updated version of JavaScript.

Description: The program shall be used only when connected to the internet so that user inputs can be processed.

Rationale: The internet connection of the user should be appropriate for the program to run.

Fit criterion: The user shall have an internet connection with at least 100kbs upload and download speed

Description: The product shall resize to the different resolutions of internet browsers and the different sizes that browsers use.

Rationale: The user should be able to resize the browser with the program resizing accordingly.

Fit criterion: The program shall be able to resize on browsers on desktops and laptops. The minimum size allowed is 720 by 1080 pixels and the maximum is 7680 by 4320 pixels.

2.1.2 Implementation Environment of the Current System

Implementation Environment of the Current System

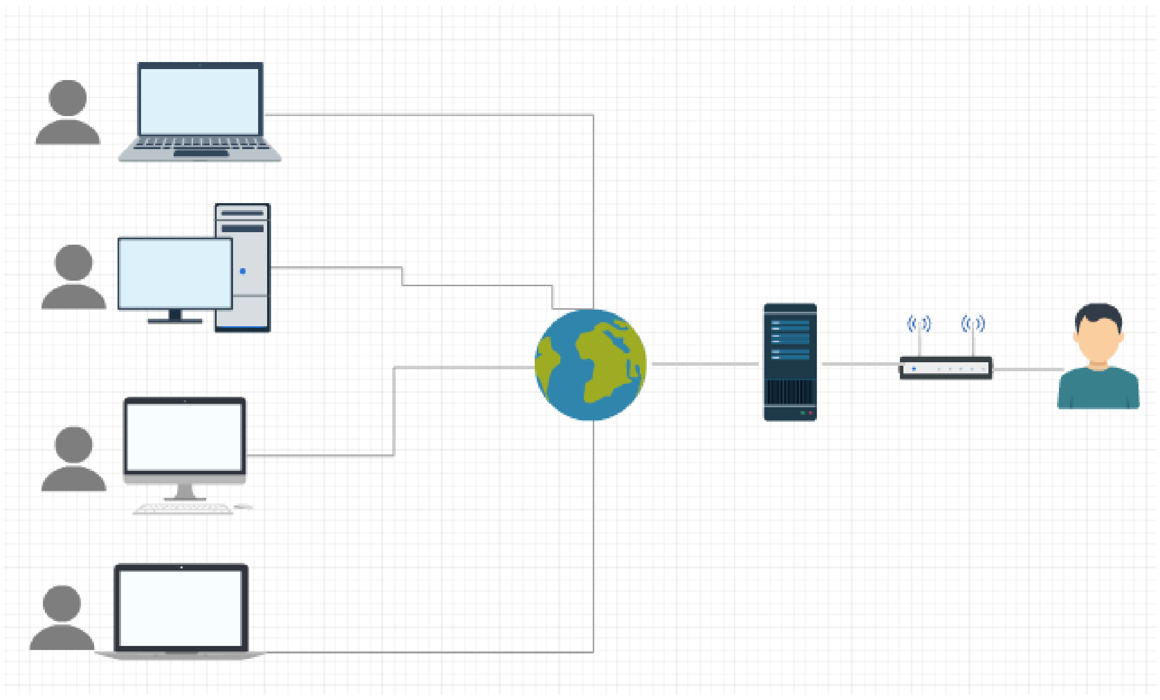

Figure 1: Implementation Environment

2.1.3 Partner or Collaborative Applications

The program relies on React, a JavaScript library to properly layout the user interface to the website. It also relies on a browser supporting JavaScript and a web-hosting server.

2.1.4 Off-the-Shelf Software

For the product to be executed the following off-the-shelf software is required:

- A web browser
- JavaScript (Available from https://java.com/en/download/)
- A client where the web browser can run (i.e. desktop, laptop)

All three of the OTS software can be received for free off the Internet.

2.1.5 Anticipated Workspace Environment

The anticipated workplace environment for the product is anywhere. As long as some sort of internet connection is present, the website can be accessed.

2.1.6 Schedule Constraints

There is no schedule constraint that affects the product requirements. However, a deadline of the beginning of December has been given to finish the project.

2.1.7 Budget Constraints

There are no budget constraints as everything that is required is either open source or free to use.

2.1.8 Enterprise Constraints

The program will be free to use and available on desktop and laptop platforms, that have browsers capable of running JavaScript.

2.2 Naming Conventions and Terminology

2.2.1 Definitions of All Terms, Including Acronyms, Used by Stakeholders Involved in the Project

Table 2: Definitions

ACRONYM/ABBREVIATION	INTENDED MEANING
JS	JavaScript
HTML	Hypertext Markup Language, used mainly in website creation
OS	Operating System
React	a JavaScript library for building user interfaces

2.3 Relevant Facts and Assumptions

2.3.1 Relevant Facts

Some relevant facts for this project are that there are 60,000 lines of code in the library. The library has a Apache license and JavaScript is needed to start using the library.

2.3.2 Business Rules

The documentation will be equally written by all members of the group. Members will specifically code parts of the project that they feel are stronger at coding.

2.3.3 Assumptions

Some assumptions about the project are the web-hosting server will be free. Also, React will be available for us to use to build the user interface for the website. In terms of coding, the parser will be coded first, as the rest of the code is reliant on this.

3 Functional Requirements

3.1 The Scope of the Work

3.1.1 The Current Situation

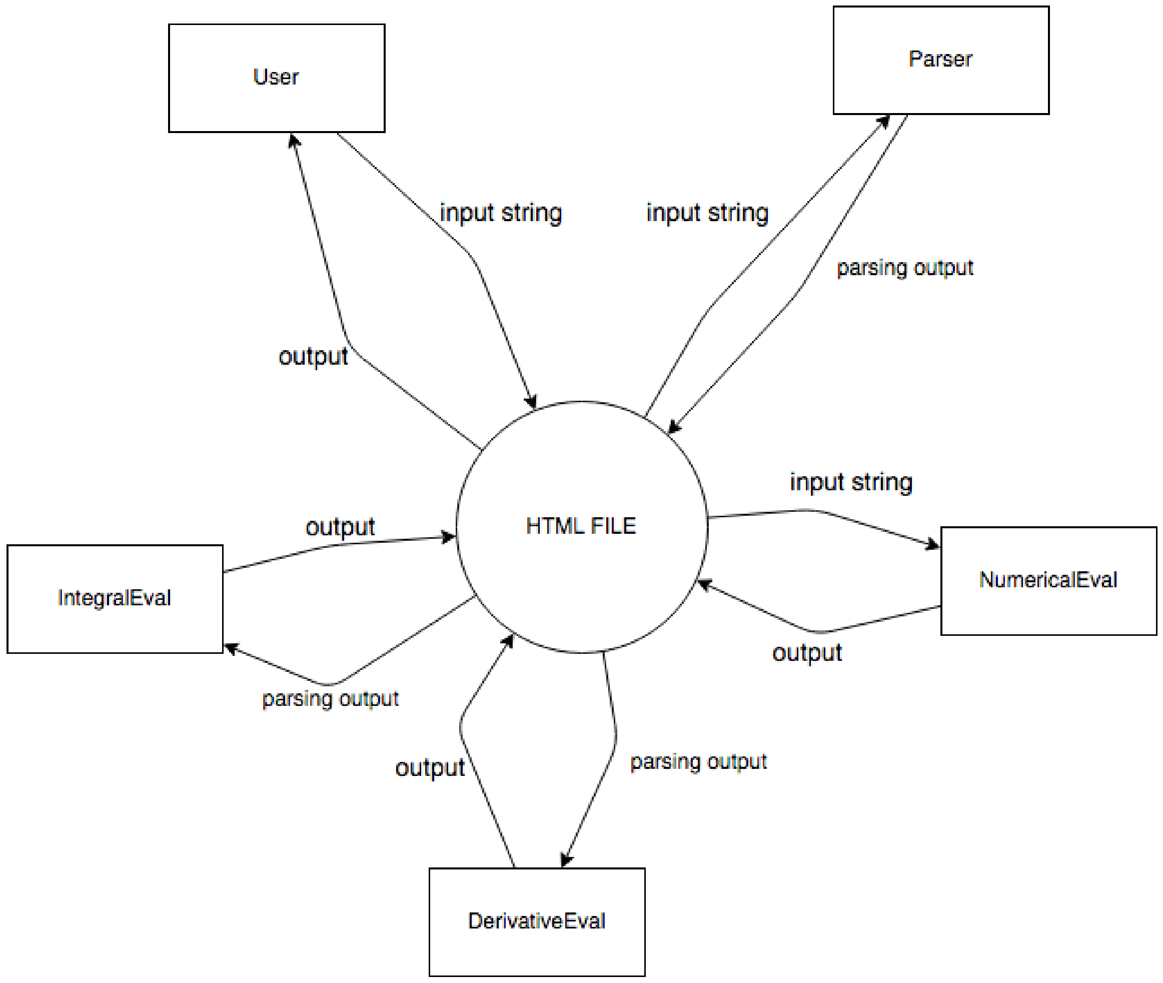

Figure 2: Data Flow Diagram

3.1.2 The Context of the Work

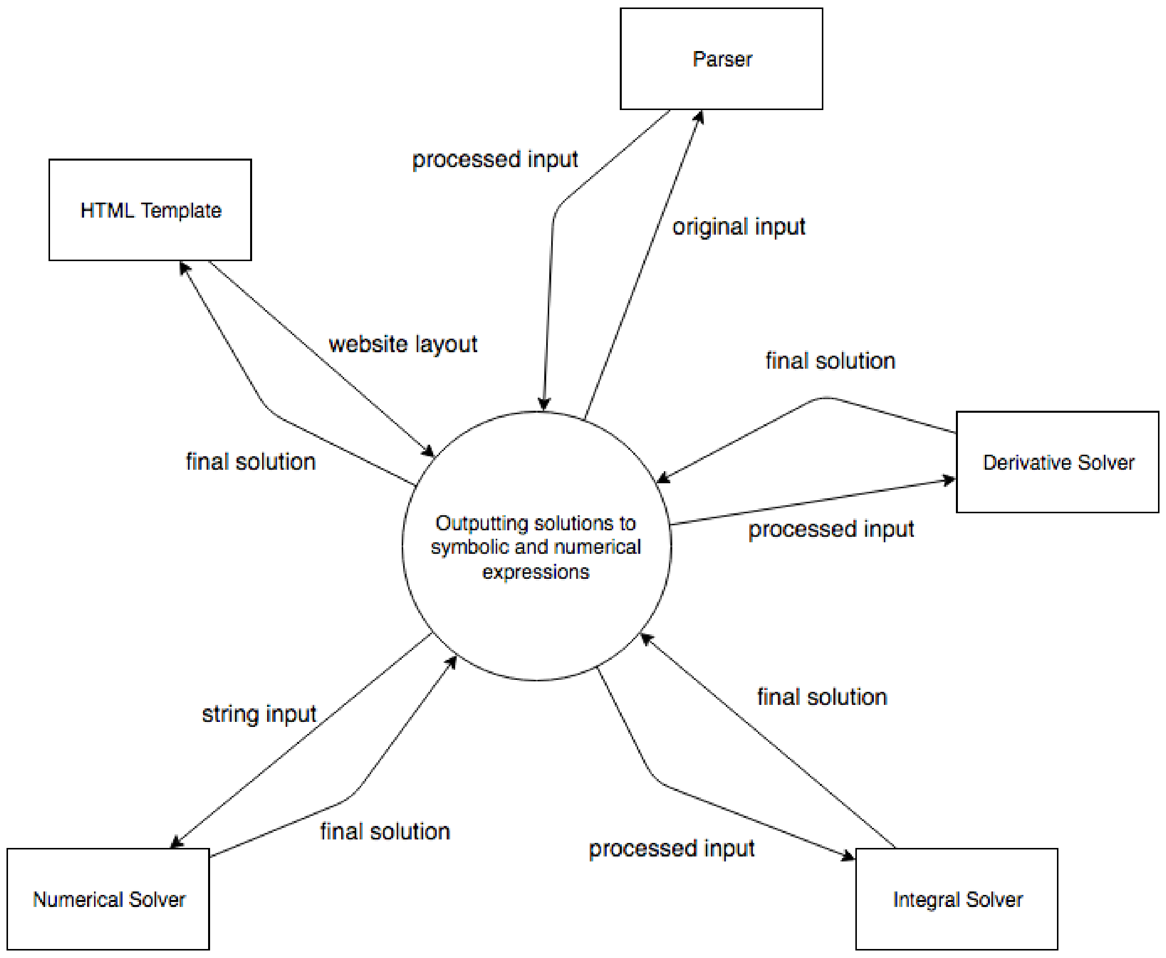

Figure 3: Work Context Diagram

3.1.3 Work Partitioning

Table 3: Work Partitioning Part I

Event Number	Event Name	Input	Output
1	Parser manipulates input	Website input label	Developer code
2	Parser organizes data	Developer code	Developer code
3	Functions process parser data	Developer code	Developer code
4	Data to Site	Developer code	Website
5	Website creation	Developer code	Website
6	Final Edits	Developer code	Website

Table 4: Work Partitioning Part II

Event Number	Summary of BUC
1	Parser receives input from user via website which will be later created
2	Parsed data will be organized in a way to make it easy to manipulate the data
3	Create functions that process the data
4	Sends processed data from code to the front end of a website that is later created
5	Create an easy to use website that will allow the user to input a mathematical problem and receive a solution for their given problem on the website
6	Finishing edits to the project

3.1.4 Specifying a Business Use Case (BUC)

Not applicable.

3.2 Business Data Model & Data Dictionary

3.2.1 Business Data Model

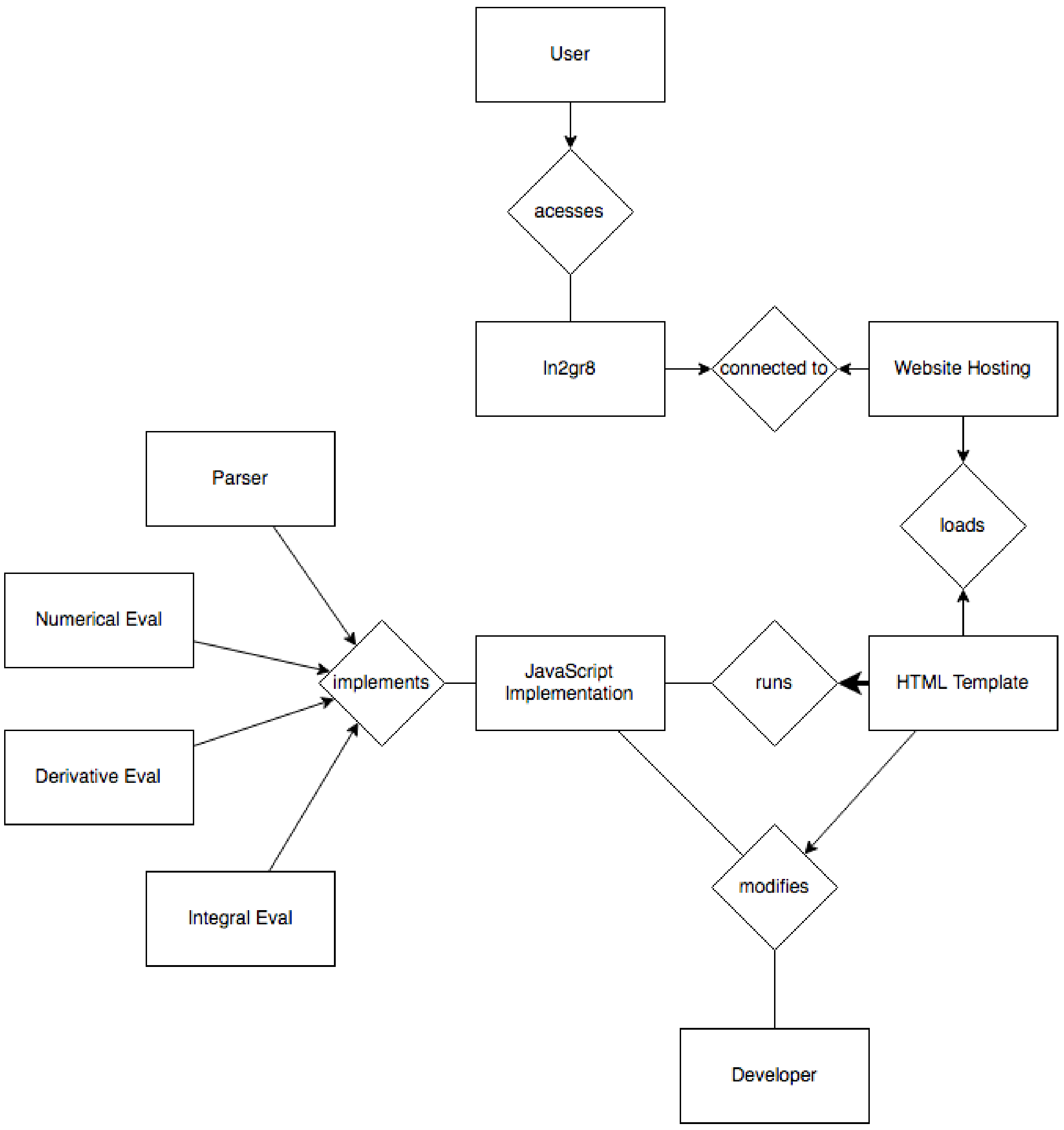

Figure 4: Business Data Model

3.2.2 Data Dictionary

Table 5: Data Dictionary

Name	Content	Type
Developer	Developer Name	Class
User	Username	Class
In2gr8	In2gr8 Instance	Class
Website Hosting	IP + Domain	Class
HTML Template	Template Name	class
JavaScript Implementation	JS Function	Class
Parser	Parser Input	Class
Numerical Eval	Parser output + Final output	Class
Derivative Eval	Parser output + Final output	Class
Integral Eval	Parser output + Final output	Class
Developer Name	String	Attribute/ Element
Username	String	Attribute/ Element
In2gr8 Instance	Created Instance ID	Attribute/ Element
IP	IP address	Attribute/ Element
Domain	Domain address	Attribute/ Element
Template Name	String	Attribute/ Element
JS Function	Document Description String	Attribute/ Element
Parser Input	String	Attribute/ Element
Parser output	Array	Attribute/ Element
Final output	String	Attribute/ Element

3.3 The Scope of the Product

3.3.1 Project Boundary

3.3.2 Product Use Case Table

3.3.3 Individual Product Use Cases

3.4 Functional Requirements

- Upon accessing the final website with the given domain, a new browser window will appear.

  Fit Criterion or Test Case:
  Does a new instance of the website appear upon accessing the website?

- The website can be accessed from any of the popular internet browsers such as Chrome, Firefox, Opera, and Microsoft Edge.

  Fit Criterion or Test Case:
  Attempt to access the website from different internet browsers.

- The website will remain idle and nothing will happen until a user passes through an input

  Fit Criterion or Test Case:
  Access the website and dont submit any input to see if the website performs any actions without any user input.

- Upon receiving input, the website will begin to solve for a solution to the given problem.

  Fit Criterion or Test Case:
  Provide user input to see if the website begins to solve the given problem.

- Upon accessing the website, the input and output fields will remain empty which is their default state.

  Fit Criterion or Test Case:
  Access the website before submitting user input to ensure that a new instance is created every time the website is accessed.

- Upon a solution being found, the solution will be displayed to the user and allow the user to input a new mathematical problem to be solved.

  Fit Criterion or Test Case:
  Return the solution to the given problem and have the ability to enter a new problem to be solved..

- If there is any errors with the user input such as unrecognized characters or improper format, the website will return an error indicating that the users input was incorrect. There will be no attempt to solve the mathematical problem is the input is incorrect.

  Fit Criterion or Test Case:
  Test improper formats or unrecognized characters to ensure that the program returns an error and does not continue until the input is fixed.

4 Non-functional Requirements

4.1 Look and Feel Requirements

4.1.1 Appearance Requirements

The product shall have one neutral theme color to not distract users from the function of the website. In2gr8 shall have labels for the input boxes to indicate the type of input that is required. The In2gr8 shall also have the name of the product visible on all pages of the website. It shall have a clear division of sections for each function that it does.

4.1.2 Style Requirements

The product shall provide a professional atmosphere that is appropriate for educational and workplace use. It should seem trustworthy so the users feel the results they receive are accurate.

4.2 Usability and Humanity Requirements

4.2.1 Ease of Use Requirements

The product shall provide all the steps taken to solve the problem to make the user feel confident the results are accurate. It shall make it easy for the user to make changes to their initial input and make it easy to clear all inputs and start over. The product shall be used by people with very little knowledge of the math concepts, the english language or training in the use of the product. The product shall be usable by users of ages 8 and up.

4.2.2 Personalization and Internationalization Requirements

The product shall allow the users to pick their preferred units for certain inputs/outputs and pick the function that they would like to use.

4.2.3 Learning Requirements

A person with access to internet and basic knowledge of browsing internet shall learn to use the product in very little to no time.

4.2.4 Understandability and Politeness Requirements

The product shall use universal math symbols.

4.2.5 Accessibility Requirements

The product shall conform to the ontarians with disabilities act policies that are related to the product.

4.3 Performance Requirements

4.3.1 Speed and Latency Requirements

The product shall compute simple solutions within 2 seconds and more complicated ones within 5 seconds. The page loading shall have a maximum response time of 2 seconds.

4.3.2 Safety-Critical Requirements

The website shall not access/compromise the users machine or data.

4.3.3 Precision or Accuracy Requirements

The product shall return solutions with 4 decimal accuracy and in exact form when possible. The product shall avoid rounding outputs as much as possible to avoid inaccuracy. The percentage error for the results shall be +/- 0.5

4.3.4 Reliability and Availability Requirements

The product shall be available for use 24 hours per day, 365 days per year.

4.3.5 Robustness or Fault-Tolerance Requirements

The other functions of the program shall continue to operate should one function fail or if it is down for maintenance.

4.3.6 Capacity Requirements

The website shall not exceed the server load.

4.3.7 Scalability or Extensibility Requirements

The server shall handle a load of approximately 10 users per day within the next year.

4.3.8 Longevity Requirements

The product shall be expected to operate until the end of course term and continue to operate for at least one year, as long as maintenance and update is performed when needed.

4.4 Operational and Environmental Requirements

4.4.1 Expected Physical Environment

The product shall be used in quiet study areas or classrooms or anywhere that there is a device with access to the internet.

4.4.2 Requirements for Interfacing with Adjacent Systems

The products shall work on the latest releases of Chrome, Safari and FireFox and it shall be accessible from computer, tablet and mobile phones.

4.4.3 Productization Requirements

The product shall be hosted on a website to become distributable.

4.4.4 Release Requirements

The product shall be updated to fix any bugs after one month of release and from then on a maintenance release will be offered to users once a year.

4.5 Maintainability and Support Requirements

4.5.1 Maintenance Requirements

The product shall be updated overnight when maintenance is being performed.

4.5.2 Supportability Requirements

The product shall have a user manual page to demonstrate how the product can be used and a help section for users to ask any questions they might have.

4.5.3 Adaptability Requirements

The product shall run on any device that has a web browser.

4.6 Security Requirements

4.6.1 Access Requirements

Anyone shall access the website to use all features but only team members shall have access to the help requests made by users.

4.6.2 Integrity Requirements

The product shall not change or delete any of the software code or alter the input user data.

4.6.3 Privacy Requirements

The product shall not collect any user data other than the information they input into the system which shall be deleted from the system once the user exists the application.

4.6.4 Audit Requirements

N/A

4.6.5 Immunity Requirements

N/A

4.7 Cultural Requirements

This product shall be able to compute solutions for various units such as metric and imperial systems. It shall also minimize the use of words and use symbols instead to accommodate users with different languages.

4.8 Legal Requirements

4.8.1 Compliance Requirements

This website shall not compromise any laws.

4.8.2 Standards Requirements

This website shall adhere to the MIT Open License.

5 Project Issues

5.1 Open Issues

N/A

5.2 Off-the-shelf Solutions

5.2.1 Ready-Made Products

- Servers

5.2.2 Reusable Components

N/A

5.2.3 Products That Can Be Copied

Since this is based off an open-sourced project, the original implementation can be useful for guidance with the base code for this new open source project.

5.3 New Problems

5.3.1 Effects on the Current Environment

This new service were offering will be a new addition to the already massive pool of websites that already exist. Therefore this new website will not have much if any affect on the environment. The only issue that may arise is if there is too much traffic to the website, where we would end up having to upgrade our servers.

5.3.2 Effects on the Installed Systems

There will be no effect on already existing system as the new system will be a new instance of the service.

5.3.3 Potential User Problems

N/A

5.3.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

The only limitation that may be experienced is if the traffic is extremely high on our website, which will result in us having to upgrade our server.

5.3.5 Follow-Up Problems

Some problems that may be experienced include not being able to develop a well made and easy to use website since our main priority is getting all our functions to function properly. We are limited by time but we hope to be able to have a fully functional website thats user friendly and has some nice aesthetics by the end of the semester.

5.4 Tasks

5.4.1 Project Planning

Table 6: default

Task	Completer?s Role	Timeline
Parsing Implementation	Software Engineers	Oct 12th
JavaScript design/functions	Software Engineers	Oct 30th
HTML implementation	Software Engineers	Nov 5rd
Hosting	Software Engineers	Nov 25rd
Maintenance	Software Engineers	Yearly

5.4.2 Planning of the Development Phases

Phase 1 Parsing Implementation: This phase is what the rest of the project depends on as we must be able to parse the user's input and organize the data in a way to make it easier to manipulate all the data. Organizing all the data to be as specific as possible will make it easier for us to develop the functions since the users input will be split up and easily accessible.

Phase 2 Different Mathematical Functions Implementations: In this phase, we will create all the different functions for each type of mathematical problems. A few examples include finding the derivative, finding the zeros, etc. As the semester goes by, the more excess time we have, the more mathematical problems we will support.

Phase 3 Website development and Hosting Implementation: After the addition of support for different mathematical problems, a website will be created in order to offer a user-friendly experience where users can input their mathematical problem and receive a nicely formatted solution.

5.5 Migration to the New Product

5.5.1 Requirements for Migration to the New Product

N/A.

5.5.2 Data That Has to Be Modified or Translated for the New System

N/A.

5.6 Risks

N/A.

5.7 Costs

None since one of the group members already has a server in place and the domain used will be a free domain offered by the hosting company.

5.8 User Documentation and Training

5.8.1 User Documentation Requirements

N/A.

5.8.2 Training Requirements

N/A.

5.9 Waiting Room

N/A.

5.10 Ideas for Solutions

An efficient way to parse a users input could be to use the split function in order to split the users input around every space, that way you can have each term of a function in a separate index in an array. Then you can search through the array and search for brackets in order to group up everything in between brackets, if applicable. To do all of this however, would require the assumption that a user has a space between each term.