<Project Name>

Version <1.0>

[Note: The following template is provided for use with the Unified Process for EDUcation. Text enclosed in square brackets and displayed in blue italics (style=InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).]

[To customize automatic fields in Microsoft Word (which display a gray background when selected), select File>Properties and replace the Title, Subject and Company fields with the appropriate information for this document. After closing the dialog, automatic fields may be updated throughout the document by selecting Edit>Select All (or Ctrl-A) and pressing F9, or simply click on the field and press F9. This must be done separately for Headers and Footers. Alt-F9 will toggle between displaying the field names and the field contents. See Word help for more information on working with fields.]

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| Date: <09/27/2003> | <1.0> | <> | <Zutshi> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction

1.1 Purpose

1.2 Scope

1.3 Definitions, Acronyms, and Abbreviations

1.4 References

1.5 Product perspective

1.5.1 System Interfaces

1.5.2 User Interfaces

1.5.3 Hardware Interfaces

1.5.4 Software Interfaces

1.5.5 Communication Interfaces

1.5.6 Memory Constraints

1.5.7 Operations

1.6 Product functions

1.7 User characteristics

1.8 Constraints

1.9 Assumptions and dependencies

2. Classification of Functional Requirements

# Introduction

The following will describe the purpose, scope, definitions, acronyms, abbreviations, references, and the overview of the Arithmetic Expressions HTML page being developed.

## Purpose

The aim off this SDP is to provide a comprehensive overview of the methodologies, development practices, and management strategies being used for the AE HTML page. All parties involved shall use this as a blueprint to engage in the project in the proper manner so that a structured, and unambiguous rollout of the product can occur. This can also be reflected on for when the document might need any kind of maintenance:

* The **project manager** uses it to plan the project schedule and resource needs, and to track progress against the schedule.
* **Project team members** use it to understand what they need to do, when they need to do it, and what other activities they are dependent upon.

## Scope

This SDP encompasses the creation of the AE (Arithmetic Expressions) HTML page. This will be a webpage capable of handling all PEMDAS expressions inputted by the user and returning the proper output after following the proper order of operations. It will also be capable of handling roots, otherwise known as fractional exponents.

## Definitions, Acronyms, and Abbreviations

* SDP: Software Development Plan
* AE: Arithmetic Expression
* PEMDAS: referencing common acronym used in arithmetic expressions to solve order of operation problems
* HTML: stands for “Hyper Text Markup Language,” it is the standard markup language used to create webpages
* CSS: stands for “Cascading Style Sheets,” stylesheet language used to format HTML doc, add color, explains how the render should look on the screen for the user interface
* PL: project lead
* FE: Front end
  + Front end development has to do with what you see and interact with on a website
* BE: Backend
  + Back end development has to do with the code that is running behind the scenes to make the front end functionable
* FEL: Front end lead
* BEL: Back end lead
* Push: committing a block of code to the main code repository
* Digital Ocean: an online cloud server host we will use to put our website available for all to see from their own hosts
* Oat++: open source C++ web framework available on github
* C++: coding language to be used on backend development
* JavaScript: code that can be integrated into an html page for the frontend
* Stakeholder: someone involved in the development process, whether it be directly working on development, funding development, or overseeing development
* POSIX(Portable Operating System Interface): family of standards specified by the IEEE for maintaining compatibility between operating systems
* cpp-httplib: a C++ library designed to handle HTTP client/server communication
* outbound connection: web application initiates a connection to another server or service on the internet
* inbound connection: When external entities, like users or other servers, initiate a connection to the web application
* OS Socket: Sockets are virtual "doorways" in a computer system, managed by the operating system, that let data in or out over a network. They operate using specific rules, known as an API
* API: Application Programming Interface, set of rules that regulates how different software’s exchange data with one another, can include for easier functionality

## References

* This subsection provides a detailed list of documents and resources cited within the SDP. Each document will be organized and recognizable by its title, report number, publication date, and/or the person who is responsible for its development. This way, if there is ever a problem regarding code integration or simply citations, this can be looked back at to resolve it.
  + Slides from EECS 388
    - Majority of plan used this as a reference
  + Iteration Plans
    - This is can be found in section 4.2
  + Digital Ocean
    - <https://www.digitalocean.com/go/developer-brand?utm_campaign=amer_brand_kw_en_cpc&utm_adgroup=digitalocean_exact_exact&_keyword=digitalocean&_device=c&_adposition=&utm_content=conversion&utm_medium=cpc&utm_source=google&gad=1&gclid=Cj0KCQjwvL-oBhCxARIsAHkOiu2-EyZwKVlsf-riHa1mgS0LzwrCJM2k6hu2t7dyAZwqQ_eQhOAK80MaAimsEALw_wcB>
  + Github
    - All our code will be stored on github, allows for everyone to pull and push code, (leads will approve code pushs)
    - <https://github.com/zmanna/SoftwareEngineering>

# Overall Description

# The document initiates with a comprehensive analysis of all external variables, spanning software and hardware, that have an impact on the project. This includes understanding the general expectations of users and the envisaged user demographic. Additionally, factors that pose limitations or constraints on the project's design are broken down

* **Corrective Maintenance Requirements**: These are quintessential and indispensable for the foundational operation of the project as a calculator.
* **Perfective Maintenance Requirements**: While they enhance the project's efficiency and user experience, they stand secondary to the main objective of the calculator's primary functions
* **Adaptive and Preventive Maintenance Requirements**: These enrich the basic capabilities of the project. However, they can be optionally included or omitted without altering the core functionality

## Product perspective

### System Interfaces

Our project is built using a Linux environment so it will have to be ran using a Linux environment. We will use the POSIX system like described above to access our OS operations.

### User Interfaces

The user interface will be through the web application. The web page will have a GUI of a calculator which will allow the user to type in their AE. The user will also be able to view their input/output history, and login to save their history.

### Hardware Interfaces

We are not responsible for any hardware management; we are outsourcing our server use to Digital Ocean.

### Software Interfaces

[blank fill this in]

### Communication Interfaces

The user will be interacting through the webpage, utilize cpp-httplib to handle this. OS sockets will be used to host these cpp-httplib interactions on the web

### Memory Constraints

Less than 512 MiB(mebibyte) of memory will need to be used for the free Digital Ocean droplet in the web application.

### Operations

The application will handle user interactions via an HTTP interface, providing responses accordingly. Behind the scenes, the system will either evaluate mathematical expressions using an integrated calculator or manage database tasks like script storage, history retrieval, or user account administration

## Product functions

Application serves as platform for user interactions. Functions include arithmetic expression evaluation, history storage, and utilizes scripting functions for added flexibility.

## User characteristics

The product is intended to be used by other software students so they can see various implementations of their own projects. The user should be able to login if they desire and have a history saved for them to return to.

## Constraints

The project is mandated to be crafted in C++, ensuring compatibility with a Linux virtual machine environment on DigitalOcean. All interactions with the operating system must employ POSIX system calls. To facilitate source code access for graders and enable potential future expansions, the project will be released under an open-source license. Project must be completed by December 5th and the breakdown of the development phases can be seen in the project plan.

## Assumptions and dependencies

Project will use multiple open source libraries including but not limited to node.json, cpphttplib, and others described above. Project is dependent on Digital Ocean server hostage for access across the web.

# Classification of Functional Requirements

Users can enter and submit arithmetic expressions via a web interface. Upon submission, the application will compute the result and present it, or flag any errors, whether from improper arithmetic formatting or server-side issues. Valid inputs include positive and negative numbers, basic arithmetic operations, and parentheses. All calculations adhere to the PEMDAS order of operations.

|  |  |
| --- | --- |
| **Functionality** | **Type** |
| all C++ functions | essential |
| add function | Essential |
| Subtract function | Essential |
| Multiply function | essential |
| Division function | essential |
| Parentheses balancer | essential |
| JavaScript calculator GUI | desirable |
| Import from backend to frontend(cpp-httplib) | essential |
| Error Messages/Handling | essential |
| User Authentication Interface | desirable |
| History Access | desirable |
| Hyperlinks to other html pages | optional |