I highly recommend referencing the “00-2024-EECS348-Term-Project.pdf” file under /Project_artifacts, it basically has all the answers

Don: 1, 1.1, 1.2
Evan: 1.3, 1.4, 1.5
John: 2.1, 2.2 ,2.3
Bradley: 2.4, 3.1, 3.3
Kyle: 4.2, 4.2.2, 4.2.3 4.2.4
Cooper: 4.3, 5, title & header/revision history

1: **Introduction**
The Software Development Plan outlines our approach for developing the Arithmetic Expression Evaluator in C++. The document provides a detailed guide for the project moving forward. This introduction highlights the plan's purpose, scope, key definitions, references, and overall structure.

1.1: **Purpose**
The purpose of the Software Development Plan is to gather all information necessary to control the development of the Arithmetic Expression Evaluator. It describes the approach to the development of the software and is the top-level plan generated and used by managers to direct the development effort.
The following people use the Software Development Plan:

- 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

1.2: **Scope**
This Software Development Plan describes the overall plan to be used by the Arithmetic Expression Evaluator in C++ project. It covers the entire project, including design, implementation, testing, and deployment of the product. The plans outlined in this document follow product requirements as defined in the Vision Document.

1.3: **Definitions, Acronyms, and Abbreviations**
This section provides definitions to a variety of key terms, acronyms, and abbreviations utilized in this software development plan for the Arithmetic Expression Evaluator.
Terms:

- Arithmetic Expression Evaluator: A C++ program that takes arithmetic expressions as input, parses them, and evaluates the result following the order of operations (PEMDAS).
- Tokenization: The process of breaking down a string of characters (such as an arithmetic expression) into smaller parts called tokens, which can be used by a parser.
- Error Handling: In programming, error handling is the process of responding to and recovering from error conditions in a program.

- Parser: A component of the program that interprets the structure of the tokenized input and processes it to evaluate the final result, following operator precedence and parentheses rules.
- Expression Parsing: The process of analyzing a string of arithmetic expressions to identify numbers, operators, and parentheses in order to evaluate them correctly.
- Operator Orders: A set of rules determining the order in which different operators (e.g., +, -, *, /) are applied in an arithmetic expression, commonly referred to by PEMDAS
-

Acronyms and Abbreviations:

- PEMDAS: Parentheses, Exponents, Multiplication, Division, Addition, Subtraction – Used for remembering the order of operations in arithmetic expressions.
- GUI: Graphical User Interface – A form of user interface that allows a user to interact with a device through graphical icons and visual indicators.
- C++: A general-purpose programming language used for this project.
- UX/UI: User Experience/User Interface – focusing on the design and feel of the program.

1.4: **References**

Iteration Plans

Title: Iteration Objectives for Arithmetic Expression Evaluator
Date: September 2024
Source: Internal Project Documentation

Vision Document

Title: *Vision Statement for Arithmetic Expression Evaluator*
Date: September 2024
Source: Internal Project Document

Glossary of Terms

Title: Glossary for Arithmetic Expression Evaluator
September 2024
Source: Internal Project Document

1.5: **Overview**

Project Overview: This section describes the project's purpose, scope, and objectives. It details the key features of the software, including expression parsing, operator precedence handling, and error management. Additionally, it outlines the expected deliverables, such as a functional C++ program, documentation, and a user manual.

Project Organization: This section defines the organizational structure of the project team, including roles and responsibilities. It describes how team members, including the Project Manager, Technical Lead, UX/UI Designer, Configuration Manager, Scrum Master, and Quality Assurer contribute to the success of the project.

Management Process: This section outlines how each team member's role contributes to the project. Along with this it also includes a table that assigns each team member by name and confines each of them to their role along with the responsibilities that come with being that role.

Applicable Plans and Guidelines: This section's purpose is to create and label iteration objectives along with brief descriptions of what the iteration is attempting to accomplish. Along with this it also includes a release log, highlighting the demo, beta and final release and providing a brief description as to how they will look. Following that, there is a schedule that includes a variety of milestones and dates at which they should be completed by.

2.1:
The program will take in an arithmetic expression for input, parse it, and then calculate while taking the order of operations (PEMDAS) into consideration. It will be able to support parenthesis and numeric constants as well.

2.2:
Initially, it is assumed that the input consists of integers only. Later, it might shift to accommodating floating point numbers if there is a request to do so.

2.3:
Deliverables for this project should include:

1. A project management plan
2. A requirements document
3. A design document
4. A test plan, with test cases, expected results, and actual results

The final deliverable should be a thoroughly-documented C++ program with the requirements listed in this document met, alongside a well-formatted README file explaining how to run and use the program.

2.4 Evolution of the Software Development Plan

Version	Date	Description of changes	Reason for revision	Authorized by
Version 1.0	9/17/24	Initial meeting created roles looked over the Software Development	Create first version of plan for the project	Project Manager

		plan Sheet		
Version 2.0	9/26	Checked over software development Plan and changed a few roles to better suit members	Changed roles to better suit team members	Project Manager

Criteria for Unscheduled Revision and Reissue:

Major requirement changes: if there is any changes in the project outline
Technical risks: if there are any errors that we identified and needed to correct in code
Design Adjustments: any change in the design of the GUI or any other unplanned changes in the design that would change the implementation schedule
Team Changes: Any team reconstructions that would change roles and maybe speed up or slow down the Implementation schedule

3.1 Organizational Structure

Project Manager: oversees the entire development process, ensures that the project sticks to their deadlines, and quality standards.

Technical lead: takes care of the technical side of the project, has the final say in design decisions, code implementation, and ensures that all the code aligns with the project requirements and meets all the expected standards.

Development team: a team of software engineers that are responsible for implementation of the code. Each member of the team will have a different role in creating the code for the project but all must work together for the final project to work properly.

Quality assurance team: this team verifies that the final versions of the project meet all the quality standards, checks the design of the program, code, and performs test cases to assure no bugs are found in the code.

Review board: similar to quality assurance this group checks the code once more to assure no bugs or errors have been overlooked and to confirm the project adheres to the project requirements.

3.3 Roles and Responsibilities

Person	Unified Process for EDUcation Role
Bradley Brown	UX/UI Designer (Designs user Interface make the project look nice and be user friendly)
Don Davis	Configuration Manager (manages the version control, Tracks major changes done to project and ensures teammates are working on the correct versions)
Kyle Flemming	Quality Assurance (Makes sure project meets the quality standards of design and functionality)
Evan Noeth	Technical lead (ensures the team follows good code practice and makes sure everyone knows how to do their tasks)
John Rader	Scrum Master (Keeps the team organized and on task and checks in with teammates to ensure the project is going smoothly)
Cooper Wright	Project Manager (Keeps track of schedules makes sure goals are being

	met,assigns tasks, and handle any unexpected issues as they come up)

4.2 Project Plan

4.2.2 Iteration objectives

- Iteration 1: Requirement Engineering & Design
    - Objective: Finalize project requirements, define scope, and develop high-level design for the expression parser - includes detailing expected features and user interface
- Iteration 2: Tokenization & Basic Parsing
    - Objective: Implement tokenization or expressions and a basic structure that recognizes numbers and basic operations such as +, -, *, /
- Iteration 3: Operation Order & Parentheses
    - Objective: Implement the logic to implement (PEMDAS) operator orders and ensure correct handling of parentheses
- Iteration 4: Advanced Operators & Error Handling
    - Implement the (%, **) operators and add error handling for invalid inputs such as division by zero or missing operators
- Iteration 5: User Interface and Testing
    - Objective: Finalize the interface for the program and develop a set of unit test to verify the correctness of the parser and make sure all error’s are covered

4.2.3 Releases

- Release 1: Demo Release (Post Iteration 2)
    - Basic working version that support basic numbers and arithmetic operations. No PEMDAS or parentheses handling yet
- Release 2: Beta Release (Post Iteration4)
    - More complete version of parser with full operator support, parentheses handling, PEMDAS order, and error detection
- Release 3: Final Release
    - Completed project including all specified operators, error handling, and finalized user interface. Documentation and a user guide will accompany the release

4.2.4 Project Schedule

Tentative schedule below

Milestone	Target Date
Requirement Finalization	09/27
Design Completion	10/04

Iteration 1	10/11
Iteration 2	10/18
Release 1	10/25
Iteration 3	11/01
Iteration 4	11/08
Release 2	11/15
Final Release	11/22