EECS 348 – Calculator Application

Software Development Plan

Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 09/19/2024 | 1.0 | Planning our project | Sione, Jett, Meg, K, Zeidan, Mohamed, Dakota |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1. Introduction 4](#_Toc11132094)

[1.1 Purpose 4](#_Toc11132095)

[1.2 Scope 4](#_Toc11132096)

[1.3 Definitions, Acronyms, and Abbreviations 4](#_Toc11132097)

[1.4 References 4](#_Toc11132098)

[1.5 Overview 5](#_Toc11132099)

[2. Project Overview 5](#_Toc11132100)

[2.1 Project Purpose, Scope, and Objectives 5](#_Toc11132101)

[2.2 Assumptions and Constraints 5](#_Toc11132102)

[2.3 Project Deliverables 5](#_Toc11132103)

[2.4 Evolution of the Software Development Plan 5](#_Toc11132104)

[3. Project Organization 5](#_Toc11132105)

[3.1 Organizational Structure 5](#_Toc11132106)

[3.2 External Interfaces 6](#_Toc11132107)

[3.3 Roles and Responsibilities 6](#_Toc11132108)

[4. Management Process 6](#_Toc11132109)

[4.1 Project Estimates 6](#_Toc11132110)

[4.2 Project Plan 6](#_Toc11132111)

[4.3 Project Monitoring and Control 7](#_Toc11132112)

[4.4 Requirements Management 7](#_Toc11132113)

[4.5 Quality Control 7](#_Toc11132114)

[4.6 Reporting and Measurement 7](#_Toc11132115)

[4.7 Risk Management 8](#_Toc11132116)

[4.8 Configuration Management 8](#_Toc11132117)

[5. Annexes 8](#_Toc11132118)

Software Development Plan

# 

# Introduction

This Development plan will cover the basic plans of the project. This will cover things such as the purpose, scope, and different roles of the project. It will give a general overview of how the project will be run and cover the stages of the project.

## Purpose

The Software Development plan's purpose is to gain all the information required to build and run the project. It will give a schedule of the different deadlines already faced for the project. Contains all the required contact information and schedule of the group which will be used by the group managers to track progress and goals.

The following roles will use the Software Development Plan:

* The Group Leader will use this to help decide on meeting dates and times.
* The Team Administrator will use this to help manage the team.
* The Quality Tester will use this to make sure each different part of the project will be done and test the accordingly
* The Technical Leader will use this to make sure each requirement of the project is met.

## Scope

The Software Development Plan describes the plan used in making the Calculator Application. It will hold the details about the group such as contact information, meeting times, and availability. The plans outlined in this document are based on the product requirements defined in the Vision Document.

## Definitions, Acronyms, and Abbreviations

See the Project Glossary.

PEMDAS: This is the order of operations, which stands for parenthesis, exponents, multiplication division, addition, and subtraction.

Compiler: It translates the C++ code that can be executed with the computers.

Debugger: it tests and debugs any program errors.

## References

* Vision
  + The vision of this project is to make a Calculator Application that handles basic arithmetic such as addition, subtraction, multiplication, division, modulo, and exponentiation.
* Glossary
  + To be updated for each iteration when necessary

## Overview

This *Software Development Plan* contains the following information:

Project Overview  — provides a description of the project's purpose, scope, and objectives.  It also defines the deliverables that the project is expected to deliver.

Project Organization  — describes the organizational structure of the project team.

Management Process  — explains the estimated cost and schedule, defines the major phases and milestones for the project, and describes how the project will be monitored.

Applicable Plans and Guidelines — provide an overview of the software development process, including methods, tools, and techniques to be followed.

# Project Overview

## Project Purpose, Scope, and Objectives

The purpose of this EECS 348 project is to create a C ++ program that acts as a calculator which can calculate numbers like addition, subtraction, division, multiplication, modulo, and exponentiation. It should also be able to follow the rules of PEMDAS/BODMAS.

Project management plan: A plan of how our project will be completed like role assignment, timelines, etc.

Requirements document: A list of what the program does, like the features and functions.

Design document: A document showing how our program structure will meet the requirements.

Test plan: Test cases to verify the C ++ code is working perfectly with no errors.C++ program: Programming the code itself that functions properly and follows all the guidelines.

User instructions: The instructions on how to use the program.

## Assumptions and Constraints

**Assumptions -**

Knowledge of C++

Compilers, debuggers, etc.,

**Constraints -**

The team is limited to 7 members.

Schedule: This project is due at the end of the Fall 2024 semester; some group members might have a conflict of time making it harder to meet deadlines as a team.

Quality standards: Error handling, easy-to-understand, documentation, etc.

## Project Deliverables

Deliverables for each project phase are identified in the Development Case. Deliverables are delivered towards the end of the iteration, as specified in section *4.2.4 Project Schedule*.

|  |  |  |
| --- | --- | --- |
| Deliverable | Description | Target Date |
| Project Plan | The plan for the outline of our project including the objectives | Due 9/29 |
| Project Requirements Plan | Plan of all the requirements of the project | 10/13 |
| Project architecture and design | The draft of the architecture and design of the project | 11/10 |
| Project implementation | Implementation of the requirements into useable code | 12/12 |
| Project test case | Test cases made based on requirements of the project | 12/12 |
| Project user manual | Manual used to teach how to use the project | 12/12 |

## Evolution of the Software Development Plan

The *Software Development Plan* will be revised prior to the start of each Iteration phase.

Current Revisions:

Sept. 19 – Went through plan and decided on how to structure the program. (Subject to change depending on the requirements of each submission deadline).

Sept. 26 – Continue to follow the outline listed for deadline one, to prepare for next iteration

Oct. 3 – Next scheduled iteration...

# Project Organization

## Organizational Structure

See Roles and Responsibilities (3.2) for team roles and organizational structure.

All submitted work from The Company Company, is subject to review by Prof. Saiedian and the rest of the EECS 348 faculty.

## Roles and Responsibilities

|  |  |
| --- | --- |
| **Person** | **Unified Process for Education Role** |
| Sione Daniels | Group Leader, 918-791-4021, GitHub, Python |
| Meg Taggart | Team Administrator, 785-259-8909, GitHub |
| Jett Viduya | Assistant Group Leader, 913-353-0686, Python, GitHub |
| Mohamed Ashraq | Assistant team administrator, 913-490-6826, Python, GitHub |
| K Li | Assistant Technical Leader,913-333-8329, python,github |
| Abdelrahman Zeidan | Technical Leader, 816-462-2766, Python, C, |
| Dakota Kling | Quality Assurance, 918-350-6029, Github, Python |

Anyone on the project can perform [Any Role](file:///C:\process\workers\wk_any.htm) activities.

# Management Process

## Project Plan

### Iteration Objectives

Main Project Documentation Iterations

* Project Plan (Part 1 & 2)
* Software Requirements
* Software Architectural Design
* Test Planning
* Full Release

### Releases

|  |  |  |
| --- | --- | --- |
| Iteration | Type | Description |
| Project Plan | Demo | An official document that outlines the overall project plan for all stakeholders. |
| Software Requirements | Pre-Alpha | Desired behaviors of program will be clearly stated, acknowledged, and accounted for during implementation. |
| Software Architectural Design | Alpha | Main development and implementation will be applied to the project to reflect established requirements. |
| Test Planning | Beta | Tests and adjustments will be conducted by the development team to ensure quality and functionality. |
| Full release | Final Version | Software product will be finalized and fully released for open use. |

### Project Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Milestones | Start Date | Target Date | Resources | Team Meetings | Additional Notes |
| Project Part 1: Project Plan  (Parts 1 & 2) | September 19th | September 29th | All team members, Laptops, Study Room | Meeting 1: September 19th  Meeting 2:  September 26th | Establish project objectives, scope, and timeline. |
| Project Part 2: Software Requirements | October 3rd | October 20th | All team members, Laptops, Study Room | Meeting 1: October 3rd  Meeting 2:  October 10th | Define and document all software requirements and specifications. |
| Project Part 3: Software Architectural Design | October 24th | November 9th | All team members, Laptops, Study Room | Meeting 1: October 24th  Meeting 2:  October 30th | Create Architectural design, user interface, and functionality. |
| Project Part 4: Test Planning | November 14th | November 28th | All team members, Laptops, Study Room | Meeting 1:  November 14th  Meeting 2:  November 21st | Review all testing activities to identify and fix bugs and make any necessary adjustments. |
| Project Part 5: User Manual (Full Release) | December 5th | December 10th | All team members, Laptops, Study Room | Meeting 1:  December 5th | Compile user documentation and finalize the user manual for release. |

## Project Monitoring and Control

* Requirements Management:
  + To ensure that the final product meets all of the requirements. We will maintain a document that will contain all the requirements. The document will be frequently reviewed, and any modifications will undergo a formal review and approval process before implementation to ensure it aligns with the project’s goals.
* Risk Managment:
  + We will take a structured approach to identify, analyze, and prioritize risks. A risk log will be used to track and identify risks, and to outline strategies. Regular reviews will ensure we continuously monitor and update the risk list, allowing us to respond effectively and adjust our mitigation plans as needed.
* Configuration Managment:
  + We will follow a formal process in which all requests are going to be submitted via GitHub. Then the group will review and resolve any issues as a collective while maintaining the quality. All project artefacts, including system software, plans, models, components, test software, results, data, and executables, will be named, marked, and versioned according to a defined naming convention
* Quality Control:
  + Using the due dates given by canvas to ensure each part of the project is done. The technical leader will oversee quality assurance of code and the overall architecture of the project. When issues are presented, they’ll be handled either by one or two members of the group or the entire group if a problem proves to be enough of an issue. The criteria of the project will be based on the information given from the canvas requirements and also the group's decisions on what is considered important or necessary.

## **Quality Control**

Defects will be recorded and tracked as Change Requests, and defect metrics will be gathered (see Reporting and Measurement below).

All deliverables are required to go through the appropriate review process, as described in the Development Case. The review is required to ensure that each deliverable is of acceptable quality, using guidelines and checklists.

Any defects found during review which are not corrected prior to releasing for integration must be captured as Change Requests so that they are not forgotten.

## **Risk Management**

Risks will be identified in Inception Phase using the steps identified in the RUP for Small Projects activity “Identify and Assess Risks”. Project risk is evaluated at least once per iteration and documented in this table.

*Refer to the Risk List Document (CCC-DDD-X.Y.doc) for detailed information.*

## **Configuration Management**

Appropriate tools will be selected which provide a database of Change Requests and a controlled versioned repository of project artifacts.

All source code, test scripts, and data files are included in baselines. Documentation related to the source code is also included in the baseline, such as design documentation. All customer deliverable artifacts are included in the final baseline of the iteration, including executables.

The Change Requests are reviewed and approved by one member of the project, the Change Control Manager role.

*Refer to the Configuration Management Plan (EEE-FFF-X.Y.doc) for detailed information.*

# 

# Annexes

The project will follow the UPEDU process.

Other applicable process plans are listed in the references section, including Programming Guidelines.

**Programming Guidelines:**

The code must satisfy OOP Principles and C++ coding standards.

Code is cryptic, comment those part which can be commented or don't manage your own memory (i.e. smart pointers)

**Design Guidelines:**

Basic design patterns

Architecture — Explain by diagrams (Class, Sequence)

**Testing Guidelines:**

Unit test (e.g. Google Test for C++).

Try for error handling, boundary cases and normal use.

**Version Control:**

Utilization of Git with meaningful commit messages and correct branching.

Documentation Guidelines:

Making sure all the documents (the Requirements Document, design document, test plans) are updated and consistent.

**Error Handling**:

print out error messages for common mistakes, e.g., division by zero

Do careful exception for runtime error