Group 16 LLC

Command Cruncher
Software Development Plan

Version 0.3

Revision History

Date	Version	Description	Author
15/09/2024	0.1	Beginning our project	Hannah, Nifemi, Sneha, Emma, Daniel
22/09/2024	0.2	Continuing work on the project	Hannah, Nifemi, Sneha, Emma, Daniel, Jonathan
22/09/2024	0.3	Filled out personal information	Warren

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Definitions, Acronyms, and Abbreviations 4

1.4 References 4

1.5 Overview 5

2. Project Overview 5

2.1 Project Purpose, Scope, and Objectives 5

2.2 Assumptions and Constraints 5

2.3 Project Deliverables 5

2.4 Evolution of the Software Development Plan 5

3. Project Organization 5

3.1 Organizational Structure 5

3.2 External Interfaces 6

3.3 Roles and Responsibilities 6

4. Management Process 6

4.1 Project Estimates 6

4.2 Project Plan 6

4.3 Project Monitoring and Control 7

4.4 Requirements Management 7

4.5 Quality Control 7

4.6 Reporting and Measurement 7

4.7 Risk Management 8

4.8 Configuration Management 8

5. Annexes 8

Software Development Plan

1. Introduction

This Software Development Plan covers the main purposes of our program, including the scope. It also overviews the other relevant definitions and references necessary for the successful execution and management of the project.

1.1 Purpose

This plan will collect the data needed to develop our project. It will be used to guide the team by providing a structure to support the creation of the project.

These users will utilize the plan:

- **Project manager** uses the plan when creating schedules to help the project stay on track.
- **Project team members** can reference this plan to see what tasks need to be completed and when, and in what order.

1.2 Scope

This *Software Development Plan* describes the plan to be used by the Command Cruncher project, including product deployment. Specific details for each iteration will be described in the Iteration Plans. The plans described in this document are based on the product requirements defined in the *Vision Document*.

1.3 Definitions, Acronyms, and Abbreviations

N/A

1.4 References

- *Vision: Our arithmetic parser program will handle the correct order of operations from input provided the user.*

1.5 Overview

This *Software Development Plan* contains the following information:

Project Overview — A description of the purpose, objectives, and scope of our project. It also contains information about the deliverables that we need to deliver and when.

Project Organization — Describes the organizational structure of the project team. Includes roles and responsibilities.

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 — Provides an overview of the software development process, including methods, tools and techniques to be followed.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

This project will allow users to make mathematical calculations. The objective of this project is to aid in understanding parsing techniques and data structures to aid the user in easily completing computations. Deliverables this project will deliver include:

- *Project management plan*

- *Requirements documentation*
- *Design specifications/Software Architecture*
- *Test cases*
- *Code*
- *User manual*
- *Final Project*

2.2 Assumptions and Constraints

Assumptions:

- Assume that everyone will attend all the meetings and contribute to the project equally.
- Assume that the project timeline is sufficient to complete all planned test cases.

Constraints:

- Time management and task delegation (finding common times in our schedules to meet up and ensuring that everyone finishes their assigned portions in a timely manner).
- Technological errors (computers crashing, code failing to compile, lack of access to the necessary devices, etc.)

2.3 Project Deliverables

- Establish the project management plan (by September 29th)
- Establish requirements documentation (by October 13th)
- Outline project architecture and design specifications (by November 10th)
- Deliver test cases (by December 12th)
- Create a user manual (by December 12th)
- Deliver project implementation (by December 12th)
- Deliver final project (by December 17th)

2.4 Evolution of the Software Development Plan

Circumstance	Changes
Changes to Due Date	Update due dates in relevant sections
Team Member Drops Class	Update team member list and roles

3. Project Organization

3.1 Organizational Structure

Change Control Manager:
- Evaluate change/pull requests
- Analyze how changes affect project scope/timeline

- Oversee code versions. Ensure proper tracking & stability
- Update team on the impact of changes
- Lead discussions to approve/reject changes

Project Manager:
- Develop the project timeline and allocate tasks
- Ensure clear communication and collaboration among team members
- Track project milestones and schedule
- Identify risks, implement mitigation strategies
- Keep stakeholders informed on the project's status
- Ensure teams has necessary resources needed to complete tasks

Lead Developer:
- Responsible for overseeing the technical aspects of the project
- Make architectural decisions
- Ensure code quality

Test Lead:
- In charge of designing and executing test cases for the evaluator
- This should include unit and integration testing for edge cases, invalid input, etc.

Documentation Manager:
- Ensure the project has well-written documentation
- Includes the user manual, README file, and in-code comments
- Oversee the writing of the project requirements and design documents

UI/UX Developer:
- Focus on developing the command-line interface that allows users to input expressions
- Ensure the UI is intuitive, and the results are displayed clearly
- If a frontend is pursued, this individual will oversee development operations

Quality Assurance Manager:
- Focus on the overall quality of the project
- Review both the development and testing processes
- Ensure all requirements are met
- Responsible for organizing QA sections of code reviews and ensuring compliance with the given coding standards

3.2 External Interfaces

- *N/A*

3.3 Roles and Responsibilities

Emma Roy

e664r593@ku.edu

- Availability: Monday: 12PM – 3PM, Tuesday & Thursday: 7PM – 9PM, Wednesday: 5PM-8PM, Friday: 4PM- 7PM
- Computing Platform Experience: Visual Studio Code
- Programming Language Knowledge: Python, C, HTML, CSS

Nifemi Lawal

w271l032@ku.edu

- Availability: Monday: 10 AM - 10:50 AM, Tuesday: 4:30 PM - 8 PM, 12:30 PM - 5:00 PM, Thursday: 4:30 PM - 7:00 PM, Friday: 12:30 - 5:00 PM
- Computing Platform Experience: Visual Studio Code, Visual Studio
- Programming Language Knowledge: Python, C, C++, C#, HTML, CSS, JavaScript, PowerShell, SQL, MATLAB

Hannah Prosch

h704p493@ku.edu
Availability: Sundays 11am-8pm, Mondays after 2pm, Fridays 2-2:30pm, Saturdays 11am-8pm
Computing Platform Experience: Eclipse, Visual Studio
Programming Language Knowledge: Java, JavaScript, SQL, HTML, CSS

Daniel Van Dalsem
d932v894@ku.edu
Availability: 9-12 Tuesday Thursday
9-11: Friday
3 on: Tuesday
4 on: Friday
Computing platform Experience: Visual Studio Code, Neovim, Jetbrains, Zed.
Programming Language Knowledge: Python, Svelte, HTML, CSS, JavaScript, Rust, Java.

Jonathan Kazmaier

j514k528@ku.edu

- Availability: Monday: 11 AM-10 PM, Tuesday: 4 PM-6 PM Wednesday: 11 AM-10 PM, Thursday: 4PM-10PM, Friday: 11 AM – 5 PM
- Computing Platform Experience: Visual Studio Code
- Programming Language Knowledge: Python, C, Java, Javascript, SQL, SOQL, HTML, CSS

Sneha Thomas

s507t485@ku.edu

- Availability: All day Monday; Wednesday 10:30 am – 1:30 pm; Friday 10am – 12pm.
- Computing Platform Experience: Visual Studio Code
- Programming Language Knowledge: Python, C, HTML, CSS, SQL

Warren Tan

w125t659@ku.edu

Availability: Monday: 2pm – 5pm, Wednesday: 2pm – 5pm, Thursday 5pm – 6pm,
Computing Platform Experience: Visual Studio Code, Eclipse
Programming Language Knowledge: Python, Java, Javascript, HTML, Lua

Person	**Unified Process for EDUcation Role**
Nifemi Lawal	UI/UX Developer
Hannah Prosch	Project Manager
Daniel Van Dalsem	Change Control Manager
Sneha Thomas	Test Lead
Emma Roy	Documentation Manager
Warren Tan	Quality Assurance Manager
Jonathan Kazmaier	Lead Developer

Meeting Information: The team will communicate via Microsoft Teams, and have weekly meetings to keep progressing on the project. To best match up with our schedule availability, we are meeting online. Our method for task allocation and tracking is in the projects tab in GitHub.

4. Management Process

4.1 Project Estimates

There are no costs associated with this project.

4.2 Project Plan

We plan on having weekly to biweekly meetings to progress on our project. We'll post meeting updates and attendance on GitHub to help record our progress and track what each group member has been working on. This will ensure that everyone is up to date on what is happening, and the project remain organized.

Schedule:

September 29, 2024	Project Management Plan
October 13, 2024	Requirements Documentation
November 10, 2024	Project Architecture & Design Specifications
December 12, 2024	Test Cases
December 12, 2024	User Manual
December 12, 2024	Project Implementation
December 17, 2024	Final Project

4.2.1 Phase Plan: N/A

4.2.2 Iteration Objectives

We plan to meet on a weekly to biweekly basis depending on the tasks assigned. During these meetings, we will fill out the required documents for that week as well as share whatever tasks we have completed on our own and plan what we will do before the next meeting.

4.2.3 Releases

None yet currently.

4.2.4 Project Schedule

Project Management Plan: September 29th

Project Requirements: October 13th

Project Architecture and Design**:** November 10th

Project Implementation: December 12th

Project Test Cases: December 12th

Project User Manual: December 12th

Project Implementation (updated project management plan, requirements, design, test cases, C++ code, and the user manual): December 17th

4.2.5 Project Resourcing: N/A

4.3 Project Monitoring and Control

- *Requirements Management:* The parser must properly handle the arithmetic operations +, -, *, /, %, and ** as well as correctly interpret parenthesis. Regular reviews will be conducted to ensure that requirements are being met throughout the project lifecycle.
- *Quality Control:* Implement procedures and techniques to ensure that all deliverables meet the required standards and specifications. Conduct code reviews regularly; team members will conduct peer reviews to ensure that each section of code is reviewed by at least two team members. Conduct walkthroughs after major milestones.
- *Reporting and Measurement: Describe reports to be generated. Specify which metrics should be collected and why.* ***OR*** *if available, refer to the* ***Project Measurements and Project Measurements*** *document*

- *Risk Management:* Risks will be identified though team meetings and code testing.
- *Configuration Management:* Changes will be addressed by the Change Control Manager of our team. Change requests will be reviewed by the rest of the team. Artifact naming must be consistent and logical for organization and traceability. Files can be named as such: TaskName_VersionNumber_Date.extension. Each version can be labeled as such: vX.Y.Z.

4.4 Requirements Management

The requirements for this system are captured in the Vision document. Requested changes to requirements are captured in Change Requests and are approved as part of the Configuration Management process.

4.5 Quality Control

Defects with the software shall be recorded as Pull Requests, and metrics about the defects will be gathered (see Reporting and Measurement below).

All deliverables shall go through the appropriate review, as described in the Development Case. The review is required to make sure that each deliverable is of the required quality, using guidelines and checklists.

Any defects discovered during review which aren't fixed before releasing for integration must be added as Pull Requests so that they are not forgotten about.

4.6 Reporting and Measurement

At the end of each iteration, we will provide metric summary reports and update our schedule estimates.

We will gather the following metrics weekly, as outlined in the RUP Guidelines for The Minimal Set of Metrics:

For complete tasks, we will assign an earned value that will be used to provide new schedule and budget estimates for the project. This information also is used to identify if our project needs adjustments made to the scope.

The trend graph of total open and closed defects will be used to calculate the remaining effort needed for the project and to make corrections.

The trend graph of acceptance test cases that passed will demonstrate to stakeholders the progress begin made.

Refer to the Project Measurements Document (AAA-BBB-X.Y.doc) for detailed information.

4.7 Risk Management

Risks for our project are identified during the inception phase as outlined in the RUP for small Projects activity “Identify and Assess Risks”. At least one time per iteration, we will evaluate the project risk and document it in a table.

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

4.8 Configuration Management

A database of Change Requests and a controlled version repository will be used to store project artifacts.

The baselines will include all of the data files, test scripts, and source code. Other documents related to the source code will also be included here. All deliverable artifacts are included in the final baseline of the iteration, including any executable files. All change requests will be approved and reviewed by a single member of this project, who has the Change Control Manager role.

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

5. Annexes

The project will follow the UPEDU process.

Other any applicable process plans are listed in the references section.