<Company Name>

Boolean Logic Calculator
Software Development Plan

Version <0.1>

Revision History

Date	Version	Description	Author
02/21/2024	0.1	Initial Template filled out: No acronyms or synonyms were decided to be used.	Ahmad Awan

Table of Contents [keep this; say N/A when inapplicable]

Software Development Plan

1. Introduction

This Software Development Plan is for a C++ program that can parse and evaluate Boolean Logic expressions. It includes the purpose, scope, definitions, acronyms, abbreviations, references and overview of this Software Development Plan.

1.1 Purpose

The purpose of the *Software Development Plan* is to gather all information necessary to control the project. 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 C++ Boolean Logic Calculator project, including deployment of the product. The details of the individual iterations will be described in the Iteration Plans.
The plans as outlined in this document are based upon the product requirements as defined in the *Vision Document*.

1.3 Definitions, Acronyms, and Abbreviations

None were decided to be used during the meeting.

1.4 References

For the Software Development Plan, the list of referenced artifacts includes:

- Iteration Plans
- Development Case
- Vision
- Glossary
- Any other supporting plans or documentation.

1.5 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.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The aim of this project is to create a C++ program that can parse and evaluate Boolean logic expressions containing operators AND(&), OR(|), NOT(!), NAND(@), and XOR($). The program should be able to handle expressions with parentheses to define precedence and grouping.

The final deliverables will contain a well-expressed project plan, a requirements document, a design document, and a set of test cases derived from the requirements and the design. The accumulation of all this will result in the final product.

2.2 Assumptions and Constraints

- All team members are full-time students (limited working time)
- Deadline of May 6th

2.3 Project Deliverables

- Engineering Artifacts
  - Project Management Plan
  - Requirements document
  - Design Specifications
- C++ Program to parse Boolean Logic Expressions
  - User Manual

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*.

2.4 Evolution of the Software Development Plan

The *Software Development Plan* will be revised prior to the start of each Iteration phase. Unplanned revisions may be done at any time for any reason, but common reasons may include typographical errors, improving clarity of document etc.

3. Project Organization

3.1 Organizational Structure

All members of the team have equal responsibility. There is no managerial hierarchy, except for the Project Leader who generally dictates Project Direction. Organizational roles are flexible and there will be change of roles through project iterations. All available roles are described.

- Project Leader: Overall project leadership; Handles scheduling, taking notes of meetings and logs team member attendance
- Assistant Project Leader: Performs all roles Project leader does, works in capacity as an assistant.
- Quality Assurance Engineer: Automated Q/A Engineer or Manual Q/A Engineer, Secondary approver of code changes.
- Team Administrator: Overviews project progress and teamwork on a daily basis.
- Technical Leader: Overviews code development and is to be referenced at any obstacle during project development.

3.2 External Interfaces

None Used

3.3 Roles and Responsibilities

Person	**Unified Process for EDUcation Role**
Ahmad Awan	Project Leader. (s706a507@ku.edu)
Shaun	Assistant Project Leader. (s286s570@ku.edu)
Jack	Team Administrator. (jcpigott15@gmail.com)
Karsten	Technical Leader. (karwolter920@gmail.com)
Mario	Quality Assurance Engineer. (simentalma@yahoo.com)

Anyone on the project can perform Any Role activities.

4. Management Process

4.1 Project Estimates

N/A

4.2 Project Plan

This section contains the schedule and resources for the project.

4.2.1 Iteration Objectives

Documentation Iteration Objectives

- Software Development Plan (this document)
- Project Requirement Document
- Project Architecture and Design Document

Software Iteration Objectives

- Expression Parsing: Parse user input, considering Boolean Logic rules
- Operator Support
  - AND (&)
  - OR (|)
  - NOT (!)
  - NAND (@)
  - XOR($)
- Unit and Integration Testing (coincidentally or right after software changes)
- User Manual

Each iteration objective may have some overlap, as well as be further subdivided into their constituent properties.

4.2.2 Releases

The latest releases are to be tracked on the online GitHub repository releases page:
https://github.com/sawan201/EECS-348-Project/tree/main/Releases

Otherwise, releases at the time of writing are as follows:

- N/A – no releases at time of writing

4.2.3 *Project Schedule*

Software Development Plan, began 02/22/2024
- Requirements Document, TARGET DATE TO BE DETERMINED
- Architecture and Design Document, TARGET DATE TO BE DETERMINED
- Implement Project (TARGET DATE TO BE DETERMINED)
o Expression Parsing (TARGET DATE TO BE DETERMINED)
o Operator Support (TARGET DATE TO BE DETERMINED), subdivided into each operator type
o Software Documentation (TARGET DATE TO BE DETERMINED)

4.2.4 *Project Resourcing*

N/A

4.3 Project Monitoring and Control

- *Requirements Management: Specify the information and control mechanisms which will be collected and used for measuring, reporting, and controlling changes to the product requirements.*
- *Quality Control: Describe the timing and methods to be used to control the quality of the project deliverables and how to take corrective action when required. Include techniques, metrics, criteria, and procedures used for evaluation— this will include walkthroughs, inspections, and reviews. Note that this is in addition to the Test Plan, which is not enclosed in the Software Development Plan.*
- *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: Describe the approach that will be used to identify, analyze, prioritize, monitor and mitigate risks. If available, refer to the **Risk List** document.*
- *Configuration Management: Describe the process by which problems and changes are submitted, reviewed, and dispositioned. Describe how project or product artifacts are to be named, marked, and numbered, including system software, plans, models, components, test software, results and data, executables, and so on. Describe retention policies, and the back-up, disaster, and recovery plans. **OR** if Available, Refer to the **Configuration Management Plan** document*

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 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.

4.6 Reporting and Measurement

Updated schedule estimates, and metrics summary reports, will be generated at the end of each iteration.

The Minimal Set of Metrics, as described in the RUP Guidelines: Metrics will be gathered on a weekly basis. These include:

Earned value for completed tasks. This is used to re-estimate the schedule and budget for the remainder of the project, and/or to identify need for scope changes.

Total defects open and closed – shown as a trend graph. This is used to help estimate the effort remaining to correct defects.

Acceptance test cases passing – shown as a trend graph. This is used to demonstrate progress to stakeholders.

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

4.7 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.

4.8 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.

5. Annexes

The project will follow the UPEDU process.

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