<Company Name>

Boolean Logic Simulator
Software Development Plan
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.] Marked (shaded) areas: items that are OK to leave out.

Revision History

Date	Version	Description	Author
24/2/24	1.0	Initial creation of document	Cyn Tran, Kyler Luong, Jackson Holle, Jack Gerety, Kaleb Howard, Malek Kchaou

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

Software Development Plan

1. Introduction

1.1 Purpose

The purpose of the Software Development Plan is to create a holistic and comprehensive documentation to collect, organize, track the requirements, designs, implementations, tests for the project that is being worked on by the team. This will allow project lead, Professors, GTA and team members alike to know what and how the team is working toward building a complete and working project.

- The group member: uses this plan to understand and know each of their responsibilities in the project. This will allow each member to work responsibly to satisfy the work that is assigned to them and to ensure the progress of the project goes as scheduled.
- The group lead: uses this plan to communicate effectively with all of the team members and make sure that the project progresses as planned. And to make sure the collaboration between each of the team members run smoothly

1.2 Scope

This Software Development Plan provides details, and description of the project. Which includes the responsibility of each member of the group. It illustrates what the team has agreed to build the project: the components, the requirements, the design, the implementation. And it shows the progress of the project including changes in agreement on the project. This allows team members to clearly communicate each other's responsibility, their needs, and each member can understand what others are working on so everyone can support each other in the team.

1.3 Definitions, Acronyms, and Abbreviations

Digital Logic - Manipulation of binary values through circuit board technology to design and implement computer operations

Boolean Logic - A type of algebra that works with logical operators and binary values.

Logic Gates - A simple switch circuit that performs a boolean logic operation to determine whether an input current is allowed to travel through. These gates are usually implemented in transistors and integrated circuits in the context of a computer.

Truth Table - Lists all possible combinations of inputs and outputs for a logic circuit in a table.

1.4 References

Klein, M. (2022, March 21). *What Boolean Logic Is & How It's Used In Programming*. Codecademy In_development. Retrieved February 24, 2024, from https://www.codecademy.com/resources/blog/what-is-boolean-logic/#:~:text=Boolean%20logic%20is%20a%20type%20of%20algebra%20in,three%20basic%20logical%20operators%3A%20AND%2C%20OR%2C%20and%20NOT.

Rouse, M. (2018, June 13). *Digital Logic*. Techopedia. Retrieved February 24, 2024, from https://www.techopedia.com/definition/27425/digital-logic

geeksforgeeks (2023, July 23). *Logic Gates - Definition, Types, Uses*. Retrieved February 24, 2024, from https://www.geeksforgeeks.org/logic-gates/

geeksforgeeks (2024, January 25). *Truth Table*. Retrieved February 24, 2024, from https://www.geeksforgeeks.org/truth-table/

1.5 Overview

This *Software Development Plan* contains the following information:

Project Overview — Describe the project's objective, purpose, scope in concise detail for the team members. Identify the constraints and assumptions that can affect the project. List the functional as well as non-functional attribute of the program

Project Organization — Describe the role of each member of the team, their responsibility, expectation and how each member works together to build the project.

Management Process — Define and identify the major phases of the project and it milestone to organizes an effective collaboration among team members in order to have the project complete and work as intended

Applicable Plans and Guidelines — Provide a description of how the project will be completed as the members agree, and an outline for how the member will work on the project.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The overall objective of this project is to create a boolean logic simulator/evaluator utilizing the object-oriented nature of C++. Some smaller objectives include developing sufficient documentation to explain the program, developing test cases to evaluate the correctness of the outputs, ensuring the program catches and communicates errors in a clear way, and developing a user-friendly interface. The deliverables that are expected of this project include common software engineering artifacts, a C++ program that meets all the requirements, and a comprehensive user manual. The artifacts expected are a project management plan, requirements document, and test cases.

2.2 Assumptions and Constraints

Constraints:

- Limited development time due to project deadlines.
- Resource constraints such as budget and availability of team members.
- Compatibility requirements with different operating systems and browsers.

Assumptions:

- Team members have a basic understanding of Boolean logic and programming concepts.
- Access to necessary development tools and software libraries.
- Stakeholders' requirements are well-defined and won't significantly change during the development process.

Functional Attributes:

- Input interface for Boolean expressions.
- Logic simulator to evaluate expressions.
- Visualization tools for illustrating logic gates and truth tables.
- Error handling and feedback mechanisms.

Non-functional Attributes:

- User-friendly interface with intuitive design.
- Efficient evaluation of complex expressions for fast simulation.
- Compatibility with multiple platforms and browsers.
- Robustness to handle unexpected inputs and edge cases.

2.3 Project Deliverables

The project will include the following major phases:

- Planning and Requirements Analysis
- Design and Prototyping
- Implementation and Coding
- Testing and Quality Assurance
- Documentation and Deployment

Milestones/Artifacts:

- Completion of requirements gathering and project planning.
- Prototype demonstration and feedback review.
- Beta release for internal testing.
- Final testing and bug fixing.
- Documentation completion and software deployment.

The project progress will be monitored through regular status updates, task tracking, and milestone reviews. Any deviations from the schedule or issues encountered will be addressed promptly to ensure timely completion.

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.

The software development process will follow the Agile methodology, with iterative development cycles and continuous feedback integration. Tools such as version control (e.g., Git), issue tracking , and collaborative platforms (e.g., Slack) will be utilized for effective communication and task management.

Members will work collaboratively using a combination of pair programming, code reviews, and regular meetings to ensure alignment and progress towards project goals. Additionally, documentation standards and coding guidelines will be established to maintain consistency and facilitate knowledge sharing among team members.

3. Project Organization

3.1 Organizational Structure

The team has 6 members: Cyn Tran, Kyler Luong, Jackson Holle, Malek, Kaleb, Jack.

Cyn Tran is the team lead, however, this role can change to fit with what the team wants. Cyn has the responsibility to document all of the group meeting, organize the date in which the group will meet. He has to make sure that the progress of the project goes as scheduled and ensure everyone in the group has a fair amount of work and that their work is assisted as needed. He has the responsibility to communicate with the GTA and ensure that the team's work is submitted successfully. He will also work with other team members to identify the requirement, design and implement the project.

Kyler has the responsibility of ensuring that the implementation of the project design meets the requirement and the functionalities of the program work as intended. He will be the main group member that identify all of the non-functional qualities of the program. He will document what and how our design works, and he will provide a test case for each of the functionality the team works on. He will also document the team failure and reason why.

Malek, Jackson, Kaleb and Jack has the responsibility as the main designer and implementer of the functionality of the program. They will work with other team members to identify all of the requirements for the project, and to come up with the design of the program. They will work with Cyn and Kyle to implement all the functionality of the program and make sure the program work as intended.

alternative way of stating the above

Team members will be assigned different roles crucial to the efficient completion and delivery of the different project artifacts.

Team Roles will include (non extensive list):

- Project Manager, Cyn: Responsible for overall project planning, scheduling, and coordination among team members.
- Software Engineers: Responsible for designing, coding, testing, and documenting the software components.
- UI/UX Designer: Designs the user interface for the simulator, ensuring usability and aesthetics.
- Quality Assurance Tester: Tests the software for bugs, errors, and performance issues.
- Technical Writer: Creates documentation including user manuals and developer guides.

Each member works collaboratively, with regular communication and coordination facilitated by the Project Manager. Weekly meetings are held to discuss progress, address any issues, and adjust plans as needed.

3.2 External Interfaces// leave out

3.3 Roles and Responsibilities *[the more details here, the easier your job; include contact info, availability info, expertise, …]*

Person	Unified Process for EDUcation Role
Kaleb Howard (316) 300-7228 kalebhoward@ku.edu	Available: All Wednesdays and Fridays, Weekends, and all evenings. Expertise: Python coding, Object oriented programming, following instructions and getting work done on time. Responsibilities: Change Control Manager, Project blueprint planning, Risk Evaluation, Co-lead
Kyler Luong (913) 375-0168 k4811988@ku.edu	Available:Weekends, evening Expertise:Python coding, communications, Object oriented Responsibilities: Code tester
Jack Gerety (417) 987-9137 jackgerety@ku.edu	Available: All weekday evenings. All weekends Expertise: Python, Object Oriented Programming Responsibilities: Programming Associate Co-lead
Jackson Holle (913) 608-1705 jacksonholle@ku.edu	Available: Weekends, evenings Expertise: Python, Object Oriented Programming, Linux Environments, Communication Responsibilities: Software Engineer Co-lead
Malek	Expertise: C, C++, Python, OPP, Github, Git Responsibility: Project Engineer Lead
Cyn (929) 273 3191 cyntran14@ku.edu	Available: Weekends and Weekday morning, and evening Expertise: C, C++, OPP Responsibility: Team lead,

Anyone on the project can perform Any Role activities.

4. Management Process

4.1 Project Estimates //Leave out

4.2 Project Plan

4.2.1 Phase Plan // leave out

4.2.2 Iteration Objectives

Operator Support

- Supports the following logical operations:AND(&), OR(|), NOT(!), NAND(@),XOR($)

Complete Expression Parsing

- Breaking down the expression and determining the order of operation

Truth Value Input

- Defining Truth Values for each variable True and False

Evaluation and Output

- Calculating the entire boolean expression

Error Handling

- Handles the invalid boolean expression

Parenthesis Handling

- Handles expression in the parenthesis properly

4.2.3 Releases

4.2.4 Project Schedule

2/24/24-3/2/24	Complete Operator support
3/2/24-3/9/24	Complete Expression Parsing
3/9/24-3/16/24	Truth Value Input
3/16/24-3/23/24	Evaluation and Output
3/23/24-3/30/24	Error Handling
3/30/24-4/6/24	Parenthesis Handling

4.2.5 Project Resourcing // leave out

4.3 Project Monitoring and Control

- *Requirements Management: When changes are provided to the product, there will be logs recording the change with time and date. What will be used to conduct these test to the product are the examples of valid and invalid boolean expressions to see if the product can identify from right and wrong.*
- *Quality Control: Every week the team will have a set to complete. When that deadline comes for that week, the team will hold a meeting to further understand what was completed and how it will blend into the main program. If it blends well, it will be further added into the program, if not it will be revised with the entire group to further enhance and have it blended in more*
- *Reporting and Measurement://leave out*
- *Risk Management: The manager will oversee all that's done to make sure everything runs as smoothly as possible and if there is a risk that occurs, the manager will hold a meeting to further discuss how it is possible to eliminate this risk.*

- *Configuration Management: The problems will be identified while building and testing the code. When problems arise, they will be tested immediately to identify the issue at hand, then it will be looked over by the team to find the best solution and resolved so it will blend in with the rest of the program. As the project is going on, every artifact will be labeled with the date, time it was created, and will be labeled for its use on the project. Every time it is changed, it will be documented what was changed and when it was changed. The team will create a backup log prior to finishing up for the day, and if something bad were to happen, the team will backtrack, or pull up that copy of the backup to so all's not lost*

4.4 Requirements Management// leave out

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// leave out

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 the 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 (4.3 Project Monitoring and Control) 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 (4.3 Project Monitoring and Control) 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.