LMSlite

Software Project Management Plan

Contributors:

Paul Davis, Benjamin Benson, Kaleab Teka

I. Introduction

In many thriving learning institutions, an online platform is crucial for success. Success of an institution can be calculated in how efficiently it keeps track of paperwork, students, courses, and faculty to name a few. This is why an intricate online platform like Learning Made Easy is needed. LMS is a software that computerizes the needed components of education to make it easy for faculty to teach and students to learn using a management system with a GUI. LMS incorporates the following functions to further support efficiency.

- ➢ Ability to organize students in sections of courses
- ➢ Ability to facilitate a list of student signed up for a specific course section
- ➢ Ability to organize undergrads and grad students based on degree plans
- ➢ Ability to differentiate between student and admin control to specify abilities for each acto**rs**

II. Project Overview

This document serves to outline the scheduling, outline, implementations and collaborations of the LMS project. The schedule for the project is documented in a gantt chart below. This document also serves to describe the managerial and technical aspects of the LMS project. The following pages contain the blueprints used to construct the GUI software using the agile software development processes.

III. Project Estimates:

Learning Made Simple seeks to develop a software capable of providing certain valued features for institutions. These features will allow institutions to keep track of student's progress using sophisticated database models. This will be delivered by following the analysis of the project requirements. This will then be followed by the design phase of the project where the requirements will be focused to be satisfied. After developing a tangible software, several tests will have been completed using simulated cases of institutions.This will be done using generated sets of random data from MAcroos.. The modeling of the database is to concise and efficient to lessen error-prone possibilities. The local database used for this project is realized using SQLite, which is an inferior software to server based database systems, however it has proven to be successful for the project.

The team structure used to develop this project is the chief programmer model. The lead programmer is held by Paul Davis and supported by programmer Benjamin Lee Benson followed by programmer Kaleab Teka. The team utilized git version control, and regular communication via GroupMe and Zoom for progress evaluation.

Gantt Chart Representation

Task I	Work Breakdown Structur	Planned Start	Planned Finish	Progress
id	lv1	plannedStart	plannedFinish	progress
1	Requirment and specifications Phase	2020/03/15	2020/03/25	100.0%
2	Design Phase	2020/03/24	2020/03/30	100.0%
3	Implimentation Phase	2020/04/01	2020/04/10	100.0%
4	Testing debugging Phase	2020/04/10	2020/04/17	100.0%
5	Review and Maintance	2020/04/15	2020/05/02	100.0%

IV. Process Model

The project was initiated on February 20 and has been used to learn and practice efficient software development lifecycles in the software engineering course, CS 3321. For efficiency's sake, the engineers of this project followed the iterative agile programming protocols to minimize errors. With the help of git-hub, utilizing version control features allowed for a stable environment to correct errors and ensure constantancy. The milestones of the agile life cycle for this project are as follows.

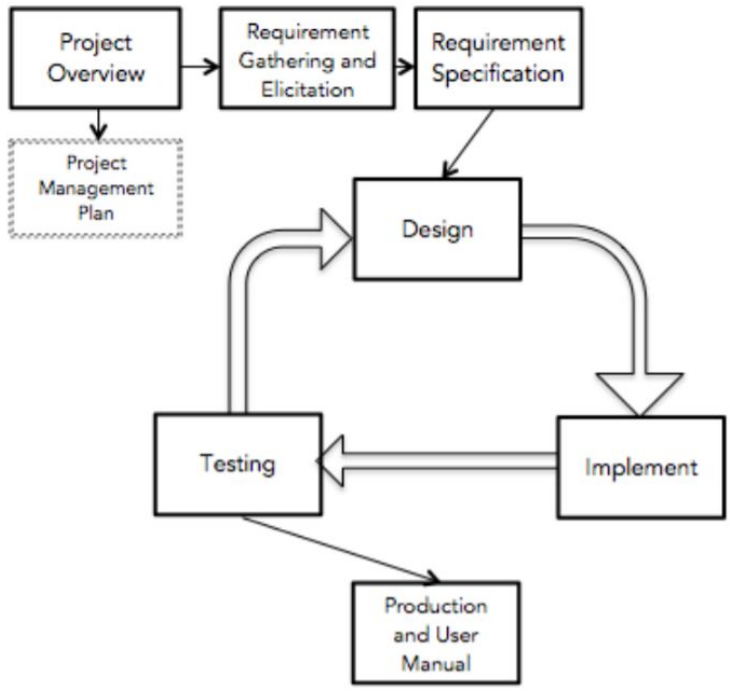

Fig. Agile Life cycle

V. Definitions, Acronyms and abbreviations

- **Database Design**: The design of the data store to be used in the system.mapping the various entities, their attributes, and how they are associated with other entities.
- **Qt Creator:** A software capable of cross platforming C++, QML and Javascript projects to construct a GUI application.
- **Graphical User Interface (GUI):** A medium that allows users to visually interact with software
- **Iteration**: In this document, iteration refers to the repetition of phases in software development cycles.
- **Agile:** Agile is a software development life cycle that involves requirement collection, requirement analysis, designing and implementation phases.
- **Use case:** An algorithmic interaction of the software by designated actors.
- **Software Design Specification (SDS)**: Specifics regarding the implementation of the project. Outlines various design decisions, including: Architectural, Interface, Database, and Component Design.
- **Testing:** A process to find and or eliminate potential bugs from code by running simulations.
- **Structured Query Language (SQL):** A programming language used to run queries from built databases.
- **Software Project Management Plan(SPMP)** : The document highlighting the major milestones of the project development
- **Application programming interface(API)**: A set of protocols that instruct the interaction of different components in an application.

Vi. Project Planning

The project planning portion involved the creation of the system requirements that are to be pursued in the implementation phase. These requirements were forged by analysing existing platforms and intuition about the needs of institutions. After looking at a list of requirements, analysis was used to calculate tasks needed to be achieved. These tasks then took on a journey of research that resulted in the making of fully developed modules that integrated together to make Learning made Simple Lite. The following comprises the general list of items completed by the team.

- Implementing Qt Creator to compile live software for user interface.
- Drawing out an E/R diagram to illustrate the connections between the entities at play.
- Research about the workings of online database platforms used by successful institutions.
- Research on how to use and implement a user interface using Qt Creator.

- Seek new methods to satisfy requirements

VII. Requirement Gathering and Analysis

To come with ample requirements for the project, we looked at a few of the successful platforms that are used by institutions like BlackBoard. From there, we came up with a list of features that the project would need to pack. The requirements we gathered right of the bat were the following:

- ❏ Must be two accessing modes: student and administrator.Students and administrators will login through a shared window.
- ❏ Software features for students and administrators should be in separate windows.
- ❏ The administrator must be able to insert, update, and monitor all processes in the software.
- ❏ Administrators should be able to view student details.
- ❏ Information available in the system will include student's name, student's ID, registered course, assignment grades, and GPA calculation

These requirements served as the basis for the project. However, the requirement phase was not yet complete. Each of the requirements needed refinement and close analysis to determine their feasibility, consistency, and completeness. After a thorough investigation, a few more requirements were added to have a complete system with integrated security. The following are the non functional requirements that were appended.

- ❏ Passwords must be kept private
- ❏ No student should be able to access another student's data
- ❏ The software must not be impaired by larger amounts of data. Speed must be maintained throughout the growth of the school.

VIII. System Design

The specific purpose of the design phase is to implement a solution that satisfies the requirements of the project. For the LMS project, the team decided to utilize a database system that maintains all of the data needed. The blueprint for the database was constructed using SQlite and written in C++. After populating the empty database using a random data generator, the application needed to be accessed in a user friendly environment. To integrate the UI, a platform that allows integration between C++ and databases was needed. Qt Creator is a software that does just that. Using Qt Creator, the realization of a live application using C++ code and QML was possible.

The actors for this project, which are distinctly students and administrators, were given different abilities in the application. The student could access grades, course information, and a few other student oriented data. As well as on the administrative side, data oriented around the role of the administrator

will be accessible. This is done by assigning login credentials to each of the actors with specific access to needed data.

IX. Analysis Review

After the application came to flourish, it was ready for an in depth analysis to check if the requirements were completely checked off. The team met via Zoom to discuss any possible corrections and enhancements that could be made. The minutes from the meeting included topics like enhancing a few features that could give the application more value like security, GUI enhancements...etc. These were then implemented and analysed again.

X. Testing and Debugging

Random student and faculty data was needed to simulate the real processes used by institutions. Therefore the team utilized an online API, Mockaroo.com, that generated random data that would allow for a genuine testing of the application. After a few runs, a few minor bugs popped up, allowing for a solution to be promptly designed.

XI. Sources

Connolly, T. M., & Begg, C. E. (2010). Database systems: A practical approach to design, implementation, and management. Boston: Addison-Wesley.

SQL Tutorial. (n.d.). Retrieved May 02, 2020, from https://www.w3schools.com/sql/sql_ref_keywords.asp

Schach, S. R. (2011). Object-oriented and classical software engineering. New York: McGraw-Hill.

Software Engineering. (n.d.). Retrieved April 04, 2020, from https://www.geeksforgeeks.org/software-engineering/

Tutorials Inhand. (2020, April 20). Software Project Management Plan - SPMP Document. Retrieved April 24, 2020, from https://tutorialsinhand.com/tutorials/software-engineering-tutorial/software-project-management/spmp-document.aspx

UML Tutorial. (n.d.). Retrieved March 10, 2020, from https://www.tutorialspoint.com/uml/index.htm

Agile Process Diagram. SPMP Team Wakati. April 29, 2014