**Software Project Management Plan**

**Learning Management System**

02/19/19

**Team Members**

Dzung

Kevin

Thanh

Alejandro

Yerusalem

# Overview

## Purpose and Scope

A graphical user interface learning management system that will allow both students, professors, and administration for a university. It must be easy to use and meet the requirements that will satisfy all three groups. The system will contain basic information such as student’s name, student’s ID, registered courses in the current semester, the exam’s score in one course, GPA calculation. The system will allow professors to manage assignments and grades including being able to modify them. Admin of the system will be able to manage login information for both staff and students. The system will not fully comprise of all the necessity of a real system, but only contain what is essential for the project.

## Goals and Objectives

*The goal of the project is to create a learning management system must store and retrieve basic information such as student’s name, student’s ID, registered courses in the semester, each exam’s score in one course, GPA in the current semester.*

Project goals:

1. Establish a learning management system for university’s students and staff.
2. Create a graphic user interface system that meet the requirements.

Project objectives:

1. Create a database or a storage for keeping the information of students and staff.
2. Create an interface that will allow students to check their courses’ information, grades, and assignments. Professors should be allowed to update course information, change grades, and

## Assumptions and Constraints

Assumptions are conditions, usually outside the control of the project team, that are taken for granted. Project plans (i.e. estimates) typically depend on certain assumptions being true. Assumptions that turn out to be false, may jeopardize project success. In order to reduce project risk, the project manager may elect to validate certain assumptions as part of the risk management process.

This is also a good place to document verbal promises or assurances given to you.

Constraints are limits or restrictions on freedom. Projects may have technical as well as non-technical constraints. Priorities for schedule and budget can impose non-technical constraints on a project. Restrictions on programming language or delivery platform are examples of technical constraints that limit design and implementation options.

*Partial Example*

Assumptions:

1. Time for meetings will be limited.
3. The Unix server and RAID controller can be purchased and delivered by 7/1/2008.
4. Facilities will provide private office space for 3 outside contractors for the duration of the project.

Constraints:

1. The software must run on a Windows Mobile 6 device.
2. The database must be open source.
3. The software must be ready by 1/1/2008.
4. The software must be used with python 3

Note, the following is not a reasonable assumption for inclusion in this section: “We assume that our group has the necessary skills and knowledge needed to complete the project.” This might be something you are taking for granted, but it is not something worth documenting in the project plan. The assumptions you want to list here are those that are outside your control. Once the development team is established, it is their responsibility to possess or develop the skills and knowledge needed to complete the project. If there is a concern that the existing team doesn’t have the skills and knowledge needed to complete the project successfully, add it as a risk and develop a plan for mitigating the risk.

## Schedule

## Success Criteria

Success criteria spell out what has to happen before the project can be considered a success. Having explicit success criteria serve two purposes. First, during a project success criteria help to focus attention on what is important. Second, at the conclusion of a project (project closure) success criteria are used to assess whether or not the goals and objectives of the project have been achieved.

To be effective in both of these endeavors, success criteria must be defined in a way that is both quantifiable and verifiable.

For more advise on how to define the success criteria for a project, I recommend: *Success Criteria Breed Success*, by Karl Wiegers. It is available on the web.

*Partial Example*

* Total project cost does not exceed 20% of the post-requirements phase estimate.
* All high-priority use cases in the requirements specification are delivered before May 15.

## Definitions

This section should define potentially unfamiliar or ambiguous words, acronyms and abbreviations.

## Evolution of the Project Plan

This section describes plans for updating the project plan throughout the project.

*Partial Example*

Before the start of an iteration, the project plan will be updated to include a schedule of detailed tasks for the upcoming iteration. At the conclusion of an iteration, the project plan will be updated to include the actual effort for each completed task.

Risk mitigation efforts will be evaluated at the start of each iteration. Severe risks will be analyzed and added to the project plan as soon as they materialize.

# Startup Plan

## Team Organization

This section explains project roles and the authorities and responsibilities associated with these roles. Lines of communication, authority and reporting relationships are often shown with an org chart. If development team is known, actual names can be associated with roles.

*Partial Example*

Project Manager: The project manager is responsible for creating the project plan (with input from those doing the work), managing risks, running the weekly team meeting and providing monthly status reports to senior management.

Programmers (3): Programmers are primary responsible for coding and unit testing modules. They are also expected to take part in architecture planning and review meetings.

Build Coordinator: The build coordinator is responsible for setting up, running and distributing the results of the nightly build.

## Technical Process

This section describes the software development methodology or conventions the team agrees to live by. When following an organization standard process, this section will refer to the standard process and state any deviations that are planned for this project. In the absence of an organization standard process, this section will define planned phases, entry and exit criteria for each phase, major milestones, workflows, and other aspects of the proposed development process.

## Tools

* Programming Language – Python 3
* Version Control – source code and written artifacts will be stored on to the github repository
* Build tools – Visual studio 2017

# Work Plan

## Activities and Tasks

A work breakdown structure is an excellent tool for identifying a complete list of tasks.

Depending on the needs of the project, some or all of the following attributes will be recorded for each task:

* Task name
* Task Description
* Owner
* Effort estimate
* Actual effort
* Planned start and stop dates
* Actual start and stop dates
* Dependencies among other tasks

## Iteration Plans

An iteration plan is a short-term fine-grained plan that shows the tasks to be completed during an iteration.

# Control Plan

## Monitoring and Control

Include in this section plans and procedures for tracking progress and controlling performance. Included here will be the approximate dates of technical as well as managerial reviews. Typically each major milestone or project phase will end in a review.

For projects that don’t have a separate communication plan, this section may also include information on the timing and content of status reports and other project review documentation.

*Partial Example*

Weekly – Team meeting. Project participants report status, progress and potential problems.

3/1/2008 – Critical Design Review. Formal inspection of product architecture.

5/15/2008 – Executive Review. The project manager presents current project status to project sponsor and senior executives.