

MARK IV

Project Management Plan

Version 1.0.5

Jeevaneasan Pillay

Veingadason S Vadi Valu

Lau Zhe Yu

Ashard

Dr. Imran Ghaini

24/08/2017

# Change history

* Version 1.0.1
  + 20/8/2017
  + Template was designed
  + The file was formatted accordingly
  + Necessary topic names were added
* Version 1.0.2
  + 21/8/2017
  + 1. Overview was completed
  + Minor changes to template
  + Added definition into table of contents
* Version 1.0.3
  + 22/8/2017
  + The project budget and schedule summary was completed
  + Process model was completed
  + Additions to template:
    - 6.3 Infrastructure Plan and corresponding subheadings
* Version 1.0.4
  + 23/8/2017
  + Complete restructuring of PMP template
  + Added definition
    - Added information on PMP, PO, SM, PDLC and SQA
  + Completed section 4.2 Methods, tools and techniques
* Version 1.0.5
  + 24/8/2017
  + Added the risk management plan
  + Added the product acceptance plan
  + Added the analysis of alternatives

# Definition

1. PMP: Project Management Plan

* Formal document used to manage the execution of the software project
* Include necessary information for team members to start working on the project

2. PO: Product Owner

* Tells the team what to build for each iteration
* Customer, or a representative of customers, or an internal staff member
* One person
* Maintains the product backlog

3. SM: Scrum Master

* Guides and facilitates the team
  + When using scrum
  + Help team handle interference
  + Removes identified obstacles

4 .PDLC: Project Development Life Cycle

* Set of steps used in the development of a software
* Typically contains 6 phases

5. SQA: Software Quality Assurance

* Set of measures taken to ensure that the software is of high quality, reliability, functionality, maintainability and usability.
* SQA include the assurance of requirements, design of code and documentation

# List of Tables

**1. TABLE 1:** Schedule and budget summary for the software. **(PAGE 7)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Milestones | Description | Criteria | Planned Dates | Budget (RM) |
| M1 | **Project Start** | **Budget allocation** | 11th Aug 2017 | 2000 |
| * Project requirements and goals defined * Scope of project defined | SRS created and reviewed, stakeholders identified |  |  |
| M2 | **Project Planning** |  | 18th Aug 2017 | 1000 |
| * Process model discussion * Allocation of roles and responsibilities | PMP, Analysis of Alternatives and Risk Register was created |  |  |
| M3 | **Project Design** |  | 25th Aug 2017 | 4000 |
| * Software design * PDLC architecture defined | Resources were committed, PDLC reviewed |  |  |
| M4 | **Sprint** |  | 8th Sept, 22nd Sept, 13th Oct 2017 | 5000 |
| * Integration of software design by developers | Functional requirements coded |  |  |
| M5 | **Project Confirmation** | **Completion of project** | 13th Oct 2017 |  |
| * Project undergoes SQA * Test case reviewed | Documentation reviewed |  |  |

**2. TABLE 2:** Personal details and roles of all team members. **(PAGE 7)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Name | Phone number | Roles | Responsibilities |
| 1 | DR. IMRAN GHAINI |  | Product Owner | Communicates with the customer |
| 2 | JEEVANEASAN | +60179772900 | Tester | Creates test cases and checks output behaviour |
| 3 | VEINGADASON | +60176986077 | Scrum Master | Facilitator, ensures team works efficiently |
| 4 | NIZNI ASHARD | +94777714882 | Developer | Designs the process for the application |
| 5 | LAU ZHE YU | +60123757332 | Developer | Designs the process for the application |

**3. TABLE 3:** Definition of done**. (PAGE 17)**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Feature | Preview | Testing |
| 1. | Unit Name Input | User inputs the unit name | Be able to take in both strings and numbers |
| 2. | Assignment Name Input | User inputs the assignment name | Be able to take in both strings and numbers |
| 3. | Add Records button | Once button is clicked, a new page will be shown where user can input student ID, name and their marks. | Ensure that once clicked, a new page will be shown with relevant details (student ID, name and marks). |
| 4. | Student ID Input | User inputs the student ID name | Be able to only take in positive integers up to 8 digits |
| 5. | Name Input | User inputs the students’ name | Be able to only take in alphabets |
| 6. | Marks Input | User inputs the marks | Be able to only take in positive integer |
| 7. | Save button | Once button is clicked, a new page will be shown | Ensure that once clicked, a new page will be shown with relevant details. |
| 8. | Choosing a scale of (1-3) | User inputs a scale of 1-3 for the 3 questions | Ensure that the user can only choose 1 scale value for each question.  Ensure that all questions must be answered in order to save the records. |
| 9. | Factor of scales | Display of factor of scale | Ensure that the right formula is used (9 different combinations) |
| 10. | Final Marks | Display of final marks | Ensure that the right formula is used to calculate the final mark (mark\*factor of scale) |
| 11. | Exit button | Once button is clicked, it will go back to the first page | Ensure that once clicked, a new page will be shown with relevant details. |

Table of Contents

[Change history 2](#_Toc491441840)

[Definition 2](#_Toc491441841)

[List of Tables 3](#_Toc491441842)

[1. Overview 7](#_Toc491441843)

[1.1 Project summary 7](#_Toc491441844)

[1.1.1 Purpose, scope and objectives 7](#_Toc491441845)

[1.1.2 Assumptions and constraints 7](#_Toc491441846)

[1.1.3 Project deliverables 7](#_Toc491441847)

[1.1.4 Schedule and budget summary 8](#_Toc491441848)

[2. Project Organization 8](#_Toc491441849)

[2.1 Roles and responsibilities 8](#_Toc491441850)

[3. Managerial process plan 8](#_Toc491441851)

[3.1 Risk management plan 8](#_Toc491441852)

[4. Technical process plans 16](#_Toc491441853)

[4.1 Process model 16](#_Toc491441854)

[4.2 Methods, tools and techniques 16](#_Toc491441855)

[4.2.1 Definition of Done 17](#_Toc491441856)

[4.2.2 Analysis of Alternatives for platform 17](#_Toc491441857)

[4.2.3 Analysis of Alternatives for language 18](#_Toc491441858)

[4.3 Product acceptance plan 19](#_Toc491441859)

# 1. Overview

## 1.1 Project summary

Developing a software to help Software Engineering demonstrators mark work that has been done in groups. This software helps demonstrators’ mark group work, as it can be very tedious and challenging sometimes. In return, the marker is able to handle tricky situations in a timely manner which may reduce their stress levels. It also serves to be a fair and reliable marking software.

The vision for this software project is for demonstrators who needs assistance in marking group assignments, MARK IV, is a marking software that grades group work of students in respective assignments unlike traditional grading system, our product provides a fair and reliable system which is web based.

### 1.1.1 Purpose, scope and objectives

This section describes the purpose of this project. It also elaborates on the objectives of the project. The scope of this software project includes the planning, design, testing and development of a reliable marking software. This software will be used to mark group assignments only. It is able to access the students’ contribution fairly and keep track of students’ marks. The scope of the project will include the completion of all documentation used in conjunction with the software.

The scope of the project excludes the type of platform to run the software and the language to code on. Moreover, it is not intended to be used by anyone other than the software engineering faculties demonstrators.

### 1.1.2 Assumptions and constraints

This section describes the assumptions and constraints that have to be followed when designing the software.

* Maximum number of group numbers in a group must be five
* Team members have to be allocated in the same lab
* Extra hardware such as servers, database are not provided
* Software needed to program the application are not provided

### 1.1.3 Project deliverables

This section describes the details and the dates of completion of the project deliverables.

* Project Inception
  + Includes the project management plan, risk register and analysis of alternatives
  + 9pm Friday, 25/08/2017
* Project Iteration 1
  + Includes the proof of concept code, product backlog, product review and resource tracking
  + 9pm Friday, 08/09/2017
* Project Iteration 2
  + Includes the milestone code, product backlog, product review and resource tracking
  + 9pm Friday
* Project Iteration 3
  + Includes the completed system, product acceptance test and resource tracking
  + 9pm Friday
* Retrospective report
  + Includes the individual reflections on how well the project was managed
  + 9pm Friday

### 1.1.4 Schedule and budget summary

This section gives an overview on the schedule to be followed during the PDLC. It also provides a summary on the budget for this project. The budget allocated for the project is RM15000. Refer to List of Table section 1 for the schedule and budget summary table.

# 2. Project Organization

## 2.1 Roles and responsibilities

The roles and responsibilities of each team member are clearly stated in this section. It also provides the personal details of all team members partaking in this project. Refer to List of Table section 2 for the team members information.

# 3. Managerial process plan

## 3.1 Risk management plan

**Impact scale:**

1. Minimal

* Extremely minimal changes to the project is done.
* Changes can be made without obstructing the pace of the project or cost the organization at all.

1. Nominal

* Only slight changes to the project is done.
* Budget overrun exceeding 5%
* Project late by more than 5%

1. Moderate

* Significant changes on the project are unlikely.
* Budget overrun exceeding 10%
* Project late by more than 10%

1. High

* May result in significant changes on any of the project features, quality or functionality.
* Budget overrun exceeding 25%.
* Project late by more than 25%

1. Huge

* May result in project failure.
* Budget overrun could exceed 50%.
* Project late by more than 50%.
* Could affect the ability of the organization to continue functioning.

**Likelihood scale:**

1. 0 - 10% - Extremely unlikely to occur
2. 11 - 35% - Very unlikely to occur
3. 36 - 50% - Unlikely to occur
4. 51 - 65% - Likely to occur
5. 66 - 90% - Very likely to occur
6. 91 - 100% - Extremely likely to occur

**Risk Matrix**

Likelihood

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 6 | 6 | 12 | 18 | 24 | 30 |
| 5 | 5 | 10 | 15 | 20 | 25 |
| 4 | 4 | 8 | 12 | 16 | 20 |
| 3 | 3 | 6 | 9 | 12 | 15 |
| 2 | 2 | 4 | 6 | 8 | 10 |
| 1 | 1 | 2 | 3 | 4 | 5 |
|  | 1 | 2 | 3 | 4 | 5 |

Impact

**Mitigation strategy categories:**

1. Avoidance – Avoid the risk completely if possible or make it impossible to happen.
2. Reduction – Minimize the likelihood or impact of the risk.
3. Transfer – Make it somebody else’s problem.
4. Acceptance – Decide to embrace the cost if the worst happens.
5. **Team member**
   * 1. **Details:**

A team member might leave the organization or be transferred to another department or team handling a different project before the completion of the project. New team members need some time to understand the requirements of the project, get used to the team dynamics as well as understanding the source code even if they were well documented.

* + 1. **Estimated impact: 2**
    2. **Estimated likelihood: 2**
    3. **Monitoring Strategy:**

The Scrum Master should keep in touch with the team members of the project and notice if there are any problems faced by the team members.

* + 1. **Mitigation plan:**

1. Avoidance – Promise incentives and career opportunities to all team members for the success of the project. This helps the team to stay not only the course of the project.
2. Reduction – Make it a policy that before a resigning or transferring departments, the team member needs to guide the new staff for at least a week in the workings of the team as well as any leftover work to be done for the project in their place.
3. Transfer - The Scrum Master should keep in touch with the team members of the project and notice if there are any problems faced by the team members. If those problems are work related, like unfair salary, the Scrum Master should be quick to resolve them to prevent the problem from blowing up, leading to a resignation or transferring departments. However, if the problems faced by the team member is personal then not only the Scrum Master but the whole team should lend a helping hand and be supportive of them.
4. Acceptance – Accept the resignation or transfer of the team member because they must have a personal reason for doing so even after offering them incentives and support to stay.

1. **Natural disaster and accidents**
   * 1. **Details:**

Natural disasters and accidents might happen during the course of the project. This might cause the death or injury of team members and damage might also be done to the servers containing the source codes and stored documents or even levelling the infrastructure of the organization depending on the severity of the disaster or accident which would slow down or halt the project entirely. New team members need some time to understand the requirements of the project, get used to the team dynamics as well as understanding the source code even if they were well documented. If the source codes are completely lost or severely damaged then rewriting everything again would be time-consuming and costly.

* + 1. **Estimated impact: 5**
    2. **Estimated likelihood: 1**
    3. **Monitoring Strategy:**

Keep a close eye on disaster warnings and install surveillance on the staff, facilities and backup servers of the organization.

* + 1. **Mitigation plan:**

1. Avoidance - Keep a close eye on disaster warnings and install surveillance on the staff, facilities and backup servers of the organization. Besides that, make sure to keep strict policies for the safety of the staff and equipment inside the facilities of the organization.
2. Reduction – Have a secure online cloud to store the project’s data as natural disasters and most accidents cannot inflict harm on non-physical objects. Besides that, the organization could also have international servers for back up as natural disasters of a global scale ae extremely rare and even if they do happen the effects would be so cataclysmic that this project is not of a priority compared to survival.
3. Transfer – Buy insurances against natural disasters and accidents to at least cover up the cost of rewriting the source code, repairs and compensation to the families of the deceased.
4. Acceptance – Nobody can stop a disaster from happening especially if they are not detected by weather stations or announced by the media in time. Besides that, nobody can predict when, how and why an accident happens, one can only reduce their chances of happening.
5. **Collusion with rival competitor or a personal vendetta**
   * 1. **Details:**

A snitch in the team might steal the project’s source codes or damage the source code intentionally as they are affiliated to a rival competitor or a personal vendetta. This would cost not only a lot of time but also money depending on the amount of damage done or information siphoned. Also, the snitch might not even be caught throughout the project and the damage might be done to the organization at the last minute so a new team member replacing the snitch is unlikely. Even if they were caught early on, new team members need some time to understand the requirements of the project, get used to the team dynamics as well as understanding the source code even if they were well documented.

* + 1. **Estimated impact: 4**
    2. **Estimated likelihood: 1**
    3. **Monitoring Strategy:**   
       A platform that monitors and records every user’s viewing, copying or extraction of any part of the source code at any time should be used by the project team. An alert from this platform would be raised if there was any unauthorised copying of codes. Besides that, install surveillance on the staff, facilities and backup servers of the organization.
    4. **Mitigation plan:**

1. Avoidance – A platform that monitors and records every user’s viewing, copying or extraction of any part of the source code at any time should be used by the project team. An alert from this platform would be raised if there was any unauthorised copying of codes. Besides that, install surveillance on the staff, facilities and backup servers of the organization.
2. Transfer - Have all team members sign a contract where a breach in the agreement which includes any evidence of sabotage and collusion would result in a huge penalty and legal repercussions for both the snitch and the benefactor.
3. Reduction – Trustworthiness in new employees should be mandatory, therefore the Scrum Master should be present during the job applicant’s interview to gauge their trustworthiness and the Scrum Master should also call the job applicant’s past workplaces and extracting information about their trustworthiness. For example, if the person in question has ever assisted in collusion or has harmed previous organizations in other ways.
4. **Client is unclear of the requirements** 
   * 1. **Details:**

The customer gives vague requirements of the project. This leads to unnecessary features which might cost the organization more or too little features that the project team was unaware of and would stall the completion of the project further.

* + 1. **Estimated impact: 4**
    2. **Estimated likelihood: 4**
    3. **Monitoring Strategy:**

Hold regular meetings with the client themselves to confirm what features or requirement are needed and not needed by the client.

* + 1. **Mitigation plan:**

1. Avoidance – Hold regular meetings with the client for a better understanding on the requirements of the project that the client needs especially when there are unclear requirements from the last meeting until all of them are resolved.
2. Reduction – Have the customer representative come by whenever a core feature has been successfully completed to make sure that the feature was required and if not then changes can be easily made than when everything is done and it turns out to not fit the requirements of the client.
3. Transfer – Have a third party handle the features that the team is unclear of especially on features the team has little expertise in.
4. Acceptance – Embrace the cost should changes need to be done and ask of the client for a delayed deadline but this time ensure that every single detail has been sorted out as no more errors can be made at this point.
5. **No backup**
   * 1. **Details:**

If there exists no backup of source codes to a secure and safe storage like an online cloud or multiple backup servers then if a natural disaster damages the storage servers, a power outage occurs which causes the developers to lose all their work or a traitor sabotaging or stealing the source codes all or most of the source codes have to be painstakingly rewritten again, costing time and money as well.

* + 1. **Estimated impact: 3**
    2. **Estimated likelihood: 3**
    3. **Monitoring Strategy:**

Regularly check if there are any crucial progress in the project being made and making sure to back them up.

* + 1. **Mitigation plan:**

1. Avoidance – Store the source code to an online cloud which is secure. Make sure there are multiple backup servers around the world if possible to avoid the risk completely.
2. Transfer – Have a third party supply the online cloud, backup servers and handle the regular updates of the servers and online cloud so that the project team can focus more on other aspects of the project.
3. Reduction – Update the backup servers weekly or per important feature.
4. **Malicious third party code** 
   * 1. **Details:**

There might be hidden lines of code in the third party’s code that was bought in order to quicken the completion of the project. For example, an accurate search function. These lines of the code might harm the existing source code or siphon data to the third party which would be disastrous.

* + 1. **Estimated impact: 4**
    2. **Estimated likelihood: 2**
    3. **Monitoring Strategy:**

Have the testers on the project team check the third party codes thoroughly

* + 1. **Mitigation plan:**

1. Avoidance – Try not to rely on third party codes at all for any part of the project to make the risk completely impossible.
2. Transfer – Have the third party sign a contract before purchasing the codes, where a breach in the agreement which includes any evidence of sabotage and collusion would result in a huge penalty and legal repercussions for the third party.
3. Reduction – Have the testers on the project team check the third party codes thoroughly and if anything suspicious occurs when running the codes then do not integrate these codes to the source codes.
4. **Hacked / virus**
   * 1. **Details:**Attacks from hackers would harm, delete, and steal data from the project team’s computers. The budget would also be exceeded and the project would be further stalled in rewriting the source codes from scratch.
     2. **Estimated impact: 3**
     3. **Estimated likelihood: 3**
     4. **Monitoring Strategy:**

Ensure that every project team member does work on a secure computer that has an active firewall and a strong antivirus software and for the computer to be handed in every two weeks for testing of any present threats.

* + 1. **Mitigation plan:**

1. Transfer – Buy insurances against cyberattacks to at least cover up the cost of rewriting the source code. A third party security team could also be hired to handle all matters related to cyber security so that the project team can focus on other aspects of the project. Educate the team through seminars and workshops on cyberattacks so that they are more wary the next time they click an unknown link.
2. Avoidance – Provide every team member with a company laptop that doesn’t allow access to anything not work-related. Work can only be done on this laptop. This way, nobody would have a chance of downloading any malware, spyware or virus intentionally or unintentionally. On the chance that the team member connects the computer to a USB flash drive or any device, the computer would first scan the drive and if there are threats detected then an email would be sent to the security team. Besides that, install surveillance on the staff, facilities and backup servers of the organization.
3. Reduction - Ensure that every project team member does work on a secure computer that has an active firewall and a strong antivirus software and for the computer to be handed in every two weeks for testing of any present threats.

1. **Software failure**
   * 1. **Details:**

Operating system crashes and causes a loss in data. If there is no backup of data then the whole source code would need to rewritten.

* + 1. **Estimated impact: 3**
    2. **Estimated likelihood: 2**
    3. **Monitoring Strategy:**

Runmonthly tests on the company computers to check if there are any problems beforehand.

* + 1. **Mitigation plan:**

1. Avoidance - Run monthly tests on the company computers to check if there are any problems beforehand and send them for repair if there are any so that they are not used when the developers are done writing a big chunk of important codes only for the operating system to crash and losing the source codes.
2. Reduction - Make sure the software of the company computers are updated and maintained regularly. If the computer hardware gets is unaccommodating of new software used by the company then replace them with better computers.
3. Transfer – Hire a third party to handle all maintenance and update of company computers to prevent any software failure from occurring.

# 4. Technical process plans

## 4.1 Process model

The process models used for this plan is scrum process model. It incorporates an agile system whereby the team goes through multiple sprints to produce a working software. A sprint is generally done in 2-4 weeks depending on the story points of the sprint backlog. In the context of this project, the sprint duration chosen is 2 weeks.

The scrum methodology also requires the scrum master to conduct 15 minutes daily meetings, or daily scrum. The developers and testers discuss what has been done since the previous daily scrum and figures out how to restructure the sprint backlog. The scrum process model is chosen because there is an active discussion occurring weekly with the stakeholders. It helps the team to proceed with a more flexible yet effective working strategy.

## 4.2 Methods, tools and techniques

This section will discuss the tools and techniques to be used in the development of this software. One particular tool to highlight in the development of this software is KanbanFlow. KanbanFlow is an interactive website which assist the users in creating Kanbans to keep track of their progress. KanbanFlow provides the option of adding custom user stories which then translate into the todo, doing, and done sections. This is a simple yet reliable way of keeping track of the progress.

As for task management and allocation, the team will have weekly scrum meetings to allocate the task to team members. KanbanFlow will be used to manage the task allocated to the team members. Moreover, KanbanFlow’s todo list serves as the backlog of the software development. All user stories that contributes to the projects goal will be added into the todo list.

Our team will be using Kanban Flow to keep track of the time spent on project tasks. For each user story, we will implement the estimated time and the actual time spent on completing the specific task. That way we will have an efficient time tracking method so that we will be able to complete the whole project in the time frame given to us.

Word 2017 will be used to document the SRS, PMP and other related documents. Word provides easy and reliable way to modify the documents if there are any changes to be made during the software life cycle.

Furthermore, during the implementation phase GitHub will serve as the medium for any code related implementation. User will be able to pull, push and solve conflicts with ease by using GitHub. GitHub will be used in conjunction with GitHub desktop to create local repositories for the team members.

### 4.2.1 Definition of Done

This section talks about procedures taken to ensure that the requirements of the user stories are met. Refer to List of Tables section 3 for a full overview on the definition of done.

### 4.2.2 Analysis of Alternatives for platform

**Factors taken into consideration**

1. Time
2. Cost
3. Scope

The factor of greatest importance would be the scope of the project. In this case it is a very basic piece of software for grading group work. Therefore, it may not be suitable to host servers and purchase domains to create a web application.

Following closely at second would be time. In the short period of time allocated for this project, it is not possible to produce the software for various platforms, and thus one particular platform is chosen.

Cost is always considered. In this case, spending large amounts of money hiring 3rd party developers or testers is over the top as it is a very simple and small scoped software project.

**Options considered**

1. Mobile application
2. Web application
3. Desktop application

Producing a mobile application for this particular software may have been appropriate, however, the team lacked the experience and knowledge required to build a mobile application.

A web application is also not ideal for this purpose, as it is costly and more time consuming to learn to develop a web application, when compared to a desktop application.

The team members have the experience and knowledge required to develop a desktop application. Therefore, it seems like the most suitable option.

**Recommendations**

In the case of choosing a platform, after considering the options and the factors such as time, cost and scope of the project, building a desktop application seemed to be the best option.

### 4.2.3 Analysis of Alternatives for language

**Factors taken into consideration**

1. Time
2. Software requirements
3. Cost

The most important factor that is taken into consideration is time. As the project has to be completed within 3 months, it is not possible for the team members to learn entirely new languages.

Software requirements is important as well. Depending on the requirements of the project, a suitable language can be chosen.   
One other important factor to consider would be the cost. In the case of choosing a language, the cost would have to be considered when learning a new language. For example: Paying for tuition fees if the team member decides to enroll in a class or online course.

**Options considered**

1. JavaScript
2. Python

JavaScript is the go to language of when it comes to web development, but in this case, only 2 out of 4 team members are confident of their skills in JavaScript. This means that the other 2 members will need to spend an extensive amount of time to learn the language and use it efficiently to develop the software and eventually contribute.

However, all 4 team members are confident in their python programming skills. Therefore, this option is the least time consuming. Furthermore, python has an extensive range of libraries, for example, Django, thus fulfills the software requirements as well.

**Recommendations**

In this case, there is a clear winner when it comes to the language as to which the software will be developed with, which is Python.

## 4.3 Product acceptance plan

The purpose to have a product acceptance plan is to ensure that not only the functionality of the software is met but all other requirements too. We will ensure the software follows the intended scope which are to be able to mark group assignments fairly by accessing students’ contribution fairly and keep track of their marks. We will ensure all deliverables are met as well as meeting all the requirements for this project. The documentation of the software will be legitimate and will meet the requirements too.