**Temasek Polytechnic**

**School of Informatics and IT**

**Diploma in Information Technology (IT)**

**Assignment 1**

**Project Particulars**

|  |  |
| --- | --- |
| **Tutor** | Mel Goh |
| **Class** | P02 |
| **Project Title** | Delonix Regia Hotel Management System |

**Project Team’s Particulars**

|  |  |
| --- | --- |
| **Matric Number** | **Student Name** |
| **1502698B** | **Gideon Ler** |
| **1506489B** | **Tan Hai Kang** |
| **1500632C** | **Eugene Sim** |
| **1502041E** | **Brandon Chew** |

Table of Contents

[1. Software Development Lifecycle – Gideon Ler 3](#_Toc482102980)

[1.1 Introduction 3](#_Toc482102981)

[1.2 Software development models 4](#_Toc482102982)

[1.3 References 7](#_Toc482102983)

[2. Software Development Lifecycle – Brandon Chew 8](#_Toc482102984)

[2.1 Introduction 8](#_Toc482102985)

[2.2 Software development Models 9](#_Toc482102986)

[2.3 References 10](#_Toc482102987)

[3. Software Development Lifecycle – HaiKang 11](#_Toc482102988)

[3.1 Introduction 11](#_Toc482102989)

[3.2 Software Development Models 12](#_Toc482102990)

[3.3 Reference 15](#_Toc482102991)

[4. Software development Lifecycle – Eugene 16](#_Toc482102992)

[4.1 Introduction 16](#_Toc482102993)

[4.2 Software development models 16](#_Toc482102994)

[Terms of Reference 18](#_Toc482102995)

[1. Introduction 18](#_Toc482102996)

[2. Objectives of the Project 18](#_Toc482102997)

[3. Scope of the Project 19](#_Toc482102998)

[4. Distribution of Workload 19](#_Toc482102999)

[5. Constraints 20](#_Toc482103000)

[6. Resources 20](#_Toc482103001)

[7. Product Positioning in the Market/Company 20](#_Toc482103002)

[8. Approach and Methodology of the Project 21](#_Toc482103003)

[Project Plan 25](#_Toc482103004)

[1 Introduction 25](#_Toc482103005)

[1.1 Objectives and scope of the project 25](#_Toc482103006)

[1.2 Assumptions and constraints 26](#_Toc482103007)

[1.3 Definitions and acronyms 26](#_Toc482103008)

[2. Roles and responsibilities 26](#_Toc482103009)

[3. Estimates and project schedule 27](#_Toc482103010)

[3.1. Work breakdown structure 27](#_Toc482103011)

[3.3. Budget Summary 31](#_Toc482103012)

[4. Risk Management Plan 32](#_Toc482103013)

[Possible risks & Reduction strategies 32](#_Toc482103014)

[Likelihood of the risks 33](#_Toc482103015)

**1. Software Development Lifecycle – Gideon Ler**

## **1.1 Introduction**

A software development lifecycle (SDLC) is a cycle that is made up of 4 key phases, requirement engineering, analysis & design, implementation, testing and finally deployment of an information system.

Requirement Gathering:

This is the first phase of SDLC, and the main aim of it is to find out what the user wants the software to be doing. The features that the users provide should be quantifiable, relevant and as detailed as possible. If not done properly, it might lead to:

* Incomplete & unstable requirements
* Misunderstanding in user goals
* Difficulty in specifying precisely

Analysis & design:

In the second phase, analysing is used by a combination of texts and diagrams forms which are meant to depict the requirements. After that, developers would need to produce a representation of the entity which would then be built. Examples of designs include the architecture design, database design as well as user interface design. If failed to do so, it might lead to:

* Poorly designed work
* Not meeting design requirements
* Compromise of design due to lack of money/time
* Having wrong designs

Implementation:

In the third phase, the details of the phase in part two of the designs are converted into detailed codes and instructions to be written in the desired programming language. Developers need to follow the coding guidelines defined by their organization and tools needed such as visual studio or compilers to make the code work. However, some problems might arise such as:

* Being tied down to certain technologies
* Bad compromises due to deadline
* Conflicting choices between cost vs time & speed vs memory usage

Testing:

In the fourth phase, testing ensures that the software does not have any bugs and is reliable, and it also ensures that the software is up to the user’s expectations and needs. However, problems regarding testing includes:

* Testing not done by developers
* Testing not done properly

Deployment:

In the last phase, the application would almost be ready and would be distributed to selected customers, prior to be released to receive user feedback for anything regarding the application. However, some challengers the developers face regarding the app might include:

* Incorrect expectation of product environment
* Not testing out production environment
* Mismatch between development and production environment

SDLC should be used as a guideline used for Mr & Mrs Wang’s system as it incorporates all the key values of while building a system. Having to know and understand how it feels like to be a customer and seeing the challenges that customers might face.

## **1.2 Software development models**

**Waterfall Model**

The waterfall model is one of the process model’s developers use to develop their websites. It step by step process, and below are the following steps:

1. Requirements
2. Analysis/Design
3. Coding
4. Testing
5. Deployment

The waterfall model is useful as it helps to maintain control of the software development process, since it can divide many of the complex procedures into smaller sub-tasks making it more manageable for developers. Each of the task also produces a well-defined deliverable, making it easy for developers and users to understand the process and what is going on. And being broken down into several tasks, it would be easier for developers to control and monitor the progress since they would need to deal with one activity at a time.

However, using the waterfall model might also have some downfalls. For example, if a problem in the development process is discovered at an early stage, nothing can be done about it. Not to mention the product can only be seen at the end, hence there would be no opportunity to validate the user’s requirements.

Hence, the waterfall model should only be used when requirements are very well known, fixed and clear, technologies used are understood, the project is short and when there are much resources and required expertise available freely.

Mr & Mrs Wang might find this model to be useful for the hotel management system if he already knows what challenges customers face when using the old hotel management system, and the budget given is a generous amount.

**Prototyping Model**

A prototype is a scaled down initial version of the system, and is not suitable as the final software product. The goal of the prototype is to clarify the requirements of the system. Users can use the prototype to get an actual feel of how the system might look like and how will it work, and it can also be used to verify if new technologies would work. Prototyping models should be used whenever there is a lot of interaction with end users, the end system would have a lot of interaction with the end users. However, continual changes could potentially corrupt the software, and the changes can also be expensive.

Examples of prototypes-

Throwaway prototype:

When the initial specifications of the software have some parts that are poorly understood by developers, the throwaway prototype is implemented. Once clients see and validated the prototype, a full specification would be written down and the prototype would be thrown away. Base on the specifications, the full-scale system would then be built.

Evolutionary prototype:

Evolutionary prototype is when an initial system is already being built and is well understood. This prototype is then shown to the users to be commented on and validated. The system would finally be refined and the process is repeated by adding more parts to the system until the system is completed.

Prototyping can be useful as it clarifies the user requirements, and since the specifications are being developed one at a time, it gives opportunity to users if they choose to change their minds.

This model might be useful for Mr & Mrs Wang’s system if are still unsure of what do the challenges customers face when using the old system and they wish to research more about it, as this model makes it easy for modifications.

**Unified Process Model**

A Unified Process (UP) model is made of 4 phases of development. Inception, elaboration, constructed and transition. It uses iterations to accommodate any ongoing risks with the system. Each phase can consist of one or more iterations, and each iteration consists of analysis, design, coding and testing. The good thing about using the UP method is that it takes the best part of a waterfall model and makes it into an iterative process, which allows room for any changes. However, it relies heavily on the stakeholder feedback, and it is also process heavy making it slower. Below are the following phases of the UP explained:

Inception:

Before any development has taken place, developers would study and discuss about the system to see whether the proposed system would be feasible and worthwhile developing by using tools such as business models.

Elaboration:

After the system has passed the inception phase, the elaboration phase is to be used for planning the project plan and following up with the requirement in a project plan. Analysis and design is taken place in this phase, and it also takes notes of the cost and the risks while developing the project.

Construction:

After all the planning is done at the elaboration phase, the development of the system is done in this phase which follows up with the requirements. Further implementations might be added at this phase, and lots of testing would be done here.

Transition:

Once the construction of the system is done, the system can finally be ready for users and clients to use.

This model might be useful for Mr & Mrs Wang if they plan on building the system based on solely what the customer expects of the system to be like instead of what they had envisioned the system to be like.

## **1.3 References**

L04-a

L04-b

<http://searchsoftwarequality.techtarget.com/definition/requirements-analysis>

<https://www.tutorialspoint.com/sdlc/sdlc_overview.htm>

Waterfall Model:

<https://airbrake.io/blog/sdlc/waterfall-model>

<http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/>

Prototyping Model:

<http://istqbexamcertification.com/what-is-prototype-model-advantages-disadvantages-and-when-to-use-it/>

UP:

<https://www.workflowmax.com/blog/choose-your-project-management-methodology-pros-and-cons-of-agile-waterfall-prism-and-more>

**2. Software Development Lifecycle – Brandon Chew**

## **2.1 Introduction**

Software Development Lifecycle (SDLC) is a process to ensure each software is well developed. There are a total of 5 key phases in each cycle. They are Requirements engineering, Design, Coding (Implementation), Testing and finally deployment. Each of these key phases has to go in order before entering the next phase to ensure good software development.

The first phase is known as “Requirements Engineering”. This is the phase where by the software developer would identify the requirements of the software. This is normally the expectations and the needs and wants of the stakeholder. Identifying the stakeholder’s requirement using interviews or surveys may help developers to understand their client better.

However, if the first phase is not well done, it may lead to multiple negative impact such as having incomplete requirements, misunderstanding between developer and client which results to unstable requirements and misunderstood goals.

The second phase is known as the “Design” phase. This is the phase where developers will design the user interface and architectural and database design into the system. If this phase is not well done, it will lead to incomplete requirements, various designs which results in confusion on which design is better or multiple derived requirements.

The third phase is known as the “Coding” phase. This is the phase where developers will program functions into their user interface, architect and database. This is written in many various programming language such as java and C#. If this phase is not well done, it will lead to being tied down to certain technology, conflicts in choices and bad results in the final product.

The fourth phase is known as the “Testing” phase. This is the phase where developers will create testing units in order to ensure that there are minimal bugs in the system. This is to ensure that the software will be reliable and will meet the clients’ needs. If this phase is not well done, it will lead to testing not done properly with people who are not developers or testing being done late.

Finally, the last phase is known as the “Deployment” phase. This is the phase where the application will be distributed among a group of selected customers. These customers may require training in how to use the new software. If this phase is not well done, it will lead to an upset due to incorrect expectation as the product was not tested in the production environment.

## **2.2 Software development Models**

There are multiple models that aids in planning the SDLC steps. Using these models, developers can predict and analyse their current development process and make adjustments.

The three models that I will be talking about are “Waterfall model”, “Unified process” and “Agile methods”.

**Waterfall Model**

The first model is the Waterfall model. This model divides the software development into different separated phases. The phase are lineal and each phase’s outcome will be used as input for the next phase. Therefore, each phase must be completed in order entirely.

Some benefits of using the waterfall model is that deliverables are well defined and it is easy for developers to understand each process. With each phase being broken down into several tasks, developers will be able to monitor each progress properly.

However, there are some downsides to the waterfall model. This model should only be used if all requirements are well understood and fixed. Not only that, the model should be short to create in order to avoid complexity. This is because that since each phase are lineal and completed completely, it will cost a lot if any requirements were to be changed.

Therefore, this model should only be used as long as requirements are well understood and fixed and the software is short to create.

**Unified Model**

The second model is known as the Unified model. This model is a development process which goes by iteration. This model can be used for any scale of project such as small or large scale. This process has a total of 4 phases: Inception, Elaboration, Construction and Transition.

The first phase is Inception where developers has to ensure that the proposed system is worthwhile before development.

The second phase is elaboration is where a complete list of requirements, design architecture and development plans is planned.

The third phase is construction whereby the developers will begin to develop the system and finally the last phase is putting the system to use for outside use.

Some benefits of using the unified process is that it focuses on showing client demonstrable products. It also reduces cost as it sets up a single team rather than several ones.

**Agile Method**

The agile development model develops software in rapid iterations known as a sprint. Each of these sprints will follow the phase on the SDLC however at the end of each sprint a working product will be displayed to the clients.

Some benefits of using the Agile Methods is that it is flexible and it provides a fast and high quality delivery. This is because that since each developments is split into many different iteration it allows for easy, flexible and high quality software developments.

However, using the agile methods will require high amount of commitment. This is because it requires a large amount of teamwork.

## **2.3 References**

Introduction:

<https://airbrake.io/blog/sdlc/what-is-the-software-development-life-cycle>

Waterfall:

<https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm>

Unified Process:

<http://www.methodsandtools.com/archive/archive.php?id=32>

Agile Method:

<https://www.versionone.com/agile-101/>

**3. Software Development Lifecycle – HaiKang**

**3.1 Introduction**  
Software Development Life Cycle is a structured procedure for developing of software. It describes the phases of software cycle to be executed when developing a software.

The following are the six phases in SDLC: -

1. **Requirement gathering**

This phase is used to gather the business requirements from the client, project manager and stake holders of the project. Some examples of general information that we should gather are; who will be using system? How are they using the system?

1. **Analysis and Design**

After requirement gathering these requirements will be taken to analyse by using a combination of text and diagrammatic forms to depict the requirement and to validate the possible possibility of incorporating it into the system. In design phase, the software design and system will be prepared from the requirement specifications. Which include architectural design, specifying hardware and system requirements and defining the overall system architecture.

1. **Implementation/ coding**

After receiving the system design documents, developer would split the work into units and start convert the instruction into programming language.

1. **Testing**

After the code is developed it will be tested to check if it met the client requirements to ensure that that needs are solved. It will also be tested on reliable to ensure that users would not encounter any errors.

1. **Deployment**

If the testing is a success, the application will be distributed to group of customers for beta testing. This is to find bug in the application or if any changed are required. Once the customers are satisfied with the changes and bug are fixed then they will move on to the final deployment.

## **3.2 Software Development Models**

Some examples of Software development process models in SLDC are Waterfall model, Agile methods and prototyping. These are approaches used during the development of software which help to plan the step required to develop the system.

**Waterfall Model** is the first Process Model that was introduced in SLDC. It is the simplest to understand and use. It consists of six phases requirements analysis, design, Implementation, testing, Installing and Maintenance. In Waterfall Model, we must complete the first phase before we can continue to the next. Waterfall model does not overlap, there will be a review taken at the end of each phase to ensure that the project is on the right path. Waterfall model is often use for smaller project.

The following are the pros and cons of Waterfall Model: -

Pros:

* **Well-documented** as Waterfall Model requires a clear documentation of each phases.
* **Clear deadlines** as waterfall are static making it easier to stick to the timeline.
* **Easy to Monitor** the progress of project as it is based on milestones.

Cons:

* **Costly to change** as the testing is usually at the end of the life cycle which make it difficult to make changes if client is not satisfy with the product.
* **Slow delivery times** as all the phases must be completed before implementation, client will not be able to see a working product until late in the life cycle
* **Neglect testing** as the testing are at the end of life cycle which could be a risk because poorly tested product can lead to a fail launch.

**Agile development model** develops software in incremental, rapid cycles. Agile method break the requirements into various builds and are work in iterations. Functional teams will work simultaneously on various phases like planning, requirement analysis, design, coding, testing and acceptance testing. The focus is on customer satisfaction and adaptability of software. At the end of each iteration, a working product will be displayed to the clients. Each subsequent iteration will add function to previous release till the system is completed.

The following are the pros and cons of Agile Methods: -

Pros:

* **Flexibility** as changing the project requirements is easier and less costly.
* **Fast, high quality delivery** as risky builds can be identified early and solved during each iteration.
* **Teamwork** as agile approach require frequent communication between the team members. Team work will benefit pair programming improving the efficiency.

Cons:

* **Neglect documentation**, hence transferring of technology to new team member is difficult due to lack of documentation.
* **High Commitment** as agile approach require teamwork where every member must be committed to the project.
* **Skill-dependent team** are not structured like Waterfall model; hence each member must be self-disciplined in their role.

**Prototype** is a scaled down version of the initial product which does not hold the exact logic of the actual product but as way to clarify the requirements. By using prototype, the client get to try the system which enable the client to know what is their desired system. It allows the designer to validate the client desired before moving on to the design or coding.

The following are approach to design a software prototype: -

* **Throwaway prototyping** build a simple prototype on poorly understood requirements. Once actual requirements are understood, the prototype will be discarded and system will be developed with a much clearer requirement.
* **Evolutionary prototyping** is based on developing part of the actual functional that are well understood. This allow the client to explore the requirements. The well understood requirements will be included in the prototype. This process will continue until the system is fully developed.

The following are the pros and cons of prototyping: -

Pros:

* **Reduce cost and time** as the development is faster and less expensive when changes are detected early in the development.
* **Improve user involvement** when prototyping require involvement with users to through a prototype system to understand the complete specification and feedbacks. This would prevent miscommunication where the developer and client think the other understand what they said.

Cons:

* **Insufficient analysis** as the developer might get too focus on prototype and are not able to analyse properly on the complete project. This would distract the developer from better solution.
* **Excessive time spent on developing prototype** could be an issue when developer is stuck in debates over details in prototype which could lead to a delay in the complete product.

In conclusion, the 3 SLDC approach above have their own pros and cons which depending upon the size of the project. In the case of Mr. and Mrs Wang, they wanted a hotel management system which could help them in improve their business. Based on the expectation from Mr. and Mrs Wang, I think that Agile model as the software development model met the requirement of them. The following are justifications pertaining to my recommendation: -

* **Time** is one factor why Agile is the best approach. As we are only given 8 weeks to complete the system, Agile model allow us plan quickly and develop quickly.
* **Flexibility** as Agile method allow changes to be made easier as compared to waterfall and prototyping model, it is an important factor as we are uncertain of the specification and making chances are expected.
* **Tracing progress is easier** as Agile method work in iterations, it allows us to display working product at each end of iteration. Hence, easier to update the progress to Mr. and Mrs. Wang periodically.

## **3.3 Reference**

Agile vs Prototyping. *Yodiz Project Management Blog*. N.p., n.d. Web. 07 May

2017.

Development Models.*Website design and development company Rns Informatics Pvt*

*Ltd*. N.p., n.d. Web. 07 May 2017.

Shiotsu, Yoshitaka. "Agile vs. Waterfall."*Hiring | Upwork*. N.p., 15 Jan. 2017. Web.

07 May 2017.

Tutorialspoint.com. "SDLC Agile Model."*Www.tutorialspoint.com*. N.p., n.d. Web.

07 May 2017.

Tutorialspoint.com. "SDLC Software Prototype Model."*Www.tutorialspoint.com*.

N.p., n.d. Web. 07 May 2017.

Tutorialspoint.com. "SDLC Waterfall Model."*Www.tutorialspoint.com*. N.p., n.d.

Web. 07 May 2017.

# **4. Software development Lifecycle – Eugene**

## **4.1 Introduction**

A system development life cycle is a term used in system engineering which describe a process for planning, analysis and design, implementation, testing, deployment. There some development models that can be used for the life cycle. A Process Model is a plan of steps to develop a project

## **4.2 Software development models**

**Waterfall**

The first model is the Waterfall model. The process of waterfall model starts from Requirements 🡪 Analysis/Design 🡪 Coding 🡪 Testing 🡪 deployment. Using Waterfall model is efficient as the tasks is divided into smaller task to maintain control over the work, thus each thus produced would be well-defined. However, viewing the product while in products would be impossible as the task is too split up.

**Prototyping**

Another Model used is Prototyping. By using prototyping, the team start with a prototype which is a scaled down version of the initial version. The goal of this is to demonstrate the feasibility and verify that the technology will work. There are 2 types of prototyping,

* **Throwaway Prototyping**

The Throwaway prototyping is usually used whenever the initial specification for the software contains vague requirements. After the showing and validating with the client, the team will write a full specification and the prototype will be discarded. A full-scale system is then built based on this specification

* **Evolution Prototyping**

Evolutionary Prototyping is used when certain parts of the system is understood well and will be developed in its initial state. The developed system will then be shown to users for comments to further improve on the system. After the feedback has been analyse, the team will then improve the system and repeat the process until the full system is complete

**Agile method**

Lastly, the agile method is a flexible process that is used to produce potentially shippable product increment at the end of each sprint. Each sprint refers to a working process. Each sprint must have a significant improvement from the initial stage. To implement Agile method, SCRUM is needed. As SCRUM is a subset of Agile, the aim of SCRUM is to maximize the productive time by keeping the overhead of the process as small as possible.

**Terms of Reference**

**Content**

[**Terms of Reference** 19](#_Toc482079457)

[1. Introduction 19](#_Toc482079458)

[**2**. Objectives of the Project 19](#_Toc482079459)

[3. Scope of the Project 20](#_Toc482079460)

[4. Distribution of Workload 20](#_Toc482079461)

[5. Constraints 21](#_Toc482079462)

[6. Resources 21](#_Toc482079463)

[7. Product Positioning in the Market/Company 21](#_Toc482079464)

[8. Approach and Methodology of the Project 22](#_Toc482079465)

# **Terms of Reference**

## **1. Introduction**

We are a group of 4 students from Temasek Polytechnic, diploma in Information Technology currently interning at a software firm called Pantheon Systems. Recently, we have been given a new project, from a hotel called Delonix Regia. The hotel is owned by Mr & Mrs Wang, and they haven't been doing well despite being in a good district. Therefore, we have been tasked to develop a Hotel Management System to attract more customers.

## **2. Objectives of the Project**

Based on the problems that Mr & Mrs Wang has faced, we have come up with features in the system that might improve the hotel’s business. We would want to create a software system which is user friendly and has a good user interface to manage the hotel easily.

Firstly, we would need to create a secure system that has anti-virus, firewall protection and encryption to protect the system from potential hackers and any malicious viruses. This way, we would be able to protect the customer’s private information and prevent the system from being hacked by hackers.

Secondly, the user interface should be simple to navigate around. On the homepage, users should login based on their privileges. Staffs should not be allowed to modify anything, while the administrators should be able to be able to modify the system such as cancelling bookings or adding in new bookings. Admins should also be able to see the full guests list as well as their information. Staff should be able to keep track of their shifts as well as be able to have a floorplan of all the rooms be it available or taken in the hotel, and housekeeping should be kept track of. The system should also be able to track down and collect data such as the number of monthly guests in the hotel, and staff information should also be accessed easily. Admins can update the staff’s information anytime.

Lastly, users should be able to access the system quickly without having to constantly wait for the page to load, and the system should run as smoothly as possible with no bugs.

## **3. Scope of the Project**

The software will firstly be able to manage the number of accommodations left available for easy identifications of which rooms are free. This will allow users who wishes to reserve a room with ease and staff members will not have a difficult time figuring out whether the room is available or not.

Secondly, the software will have an in-built list with an event management system. This will allow employees to keep track on which rooms are required for housekeeping services. Staff members can add into the list to update housekeepers on additional rooms needed for housekeeping. This allows for an easy method of communication among staffs.

Finally, a finance management system will be available. This will display transactions, taxes and earnings in a chart for easier identifications of sales. This reduces the time spent for accountants when recording profits.

In conclusion, the Hotel management software will have an accommodation management system, an event management system and a finance management system as key features in the system.

## **4. Distribution of Workload**

|  |  |
| --- | --- |
| **Objectives/Deliverables** | **Members** |
| * Introduction * Objectives of the project | **Gideon Ler** |
| * Approach and Methodology of the Project | **Tan Hai Kang** |
| * Scope of the project * Constraints | **Brandon Chew** |
| * Resources * Product Positioning in the Market/Company | **Eugene Sim** |

## **5. Constraints**

These are the following constraints:

* Time
* Limited meetup days
* Other project assignments due date
* Limited customer interaction

## **6. Resources**

* Microsoft project
* Printer
* Microsoft Visual Studio
* Rational function tester
* Macromedia Dreamweaver
* Manpower

## **7. Product Positioning in the Market/Company**

As there are many software implemented, there are bound to be similar software created by others. However, our software is unique as it makes accommodation easier whether to customers or employee. From customer’s perspective, they can view rooms that are available and also book a room in advance so as to ensure a room without making a physical appearance. There is also a language translator which helps international customers to ease their difficulty when booking a room.

From employee’s perspective, they can manage rooms that are empty which helps room cleaners accommodate with the room cleaning. Furthermore, the software also allows employee to calculate their profit and losses with having to go through the trouble to calculate them. Therefore, these conclude the understanding and position for our product into market.

## **8. Approach and Methodology of the Project**

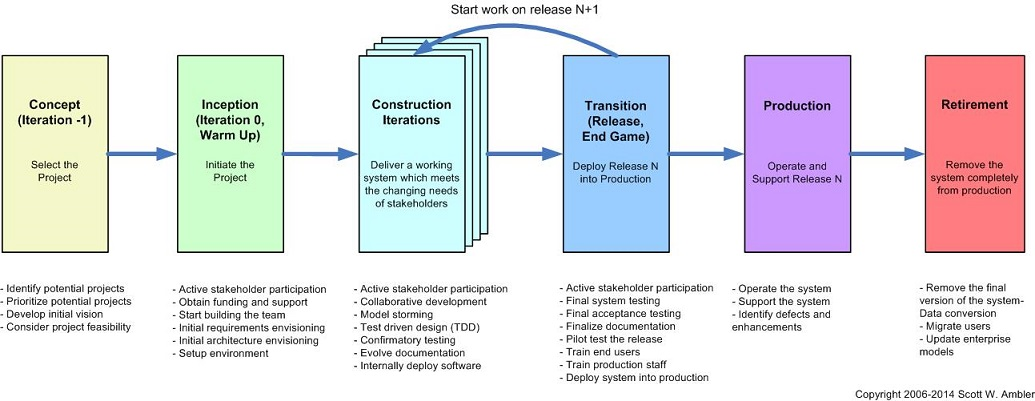
The approach that our team have selected is Agile methodology.

Figure 1: Illustration of Agile system life cycle

Agile SDLC model combine the use of iterative and incremental model. The focus is on processing adaptability and by delivery of working software product in the early stage of life cycle. Agile process break the requirements into various builds and work in iterations. Every iteration requires the functional teams to work simultaneously on various areas. At the end of each iteration, a working product will be shown to the client. This process will continue until the entire system is completed.

As agile methods have many different specific methods, the method we would be using is Scrum approach. Scrum emphasizes on decision making from real-world results than peculation, and it has simple sets of roles, responsibilities and meetings that would never change, making it clearer for our team to tackle the project without interfering with each other.

Scrum is made of 3 roles:

* Scrum master

To act as the facilitator for the team

* Product Owner

To continuously communicate the team’s vision and priorities.

* Team

Responsible for organizing and completing the work

The scrum master should be someone who is able to remove anything that is obstructing the team from achieving its sprint goal. The product owner should be someone who has vision and authority, and the team should be able to accomplish the goal of designing, building and testing the product.

Some of the benefits of Agile methods includes ease of making changes and rapid delivery of project. However, there are bound to be some issues that we might face while adopting this development model. The following are some potential risks that our team might face: -

* **Over budget and late deliverable** can happen as our team are inexperience in Agile practices. Agile method requires skilled developer who are adaptable and able to think independently as there is minimum documentation. Hence, to overcome this problem the team leader must ensure that every member of the team understood the requirements and working on the right track. This would prevent unnecessary changes which can save cost and time.

* **Lack of customer interaction** is a major problem for our team as Agile methods depend heavily on customer interaction. If the instructions is not clear the whole team may get driven to the wrong direction and the final product might gross differently from what the client want. One way to overcome this problem is by doing analysis on other hotel management software that are currently available in the market. This allow us to depict the functionality and how final product work.

**Project Plan**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 26/5/17 | 1.1 | Started on introduction and objective of project. | Gideon  Hai Kang  Brandon  Eugene |
| 29/5/17 | 1.2 | Listed our assumption and some potential constraint. | Gideon  Hai Kang  Brandon  Eugene |
| 5/5/17 | 1.3 | Completed a draft of project plan. | Gideon  Hai Kang  Brandon  Eugene |
| 9/5/17 | 1.4 | Made some changes on few areas, finalising the project plan. | Gideon  Hai Kang  Brandon  Eugene |

**Contents**

[**Project Plan** 27](#_Toc482080053)

[**1 Introduction** 27](#_Toc482080054)

[**1.1 Objectives and scope of the project** 27](#_Toc482080055)

[**1.2 Assumptions and constraints 28**](#_Toc482080056)

[**1.3 Definitions and acronyms 28**](#_Toc482080057)

[**2. Roles and responsibilities** 29](#_Toc482080058)

[**3. Estimates and project schedule** 30](#_Toc482080059)

[**3.1. Work breakdown structure** 30](#_Toc482080060)

[**3.2. Project Schedule** 32](#_Toc482080061)

[**3.3. Budget Summary** 34](#_Toc482080062)

[**4. Risk Management Plan** 35](#_Toc482080063)

[**Possible risks & Reduction strategies** 35](#_Toc482080064)

[**Likelihood of the risks** 36](#_Toc482080065)

# **Project Plan**

## **1 Introduction**

### **1.1 Objectives and scope of the project**

This project plan would provide the definition of the project and outline what are the project’s objectives and goal would be, based on the terms of reference. Below is the list of features we wish to include in our project: -

* **Login system** to identify the privileges of user there are Admin module and staff module.
* **Home Page of HMS (Admin)** allow the administrator to access the feature to manage staff, guest, accommodation and finance.
* **Home Page of HMS (Hotel Staff Members)** allow the hotel staff to check the hotel status how many rooms are occupied and vacant, and the types of rooms that are currently available, be it single rooms, double rooms or suites.
* **Staff management** showcase the list of staff in the hotel as well as their information such as name, DOB, NRIC number, salary and their work schedule. Administrators are given the option of adding or removing staffs, and have the ability to edit staff information.
* **Guest management** display the listof guest information, which shows their name, age, check in and checkout dates, contact information, number of days staying, id, address and room type. Administrators are able to add new guests and remove guests once they have finished staying.
* **Accommodation management** keep track on which rooms are required for housekeeping services, whether they are occupied or not. It would also show the occupants name beside it. House cleaning staff can use this information to verify which rooms need house cleaning and are able to add into the list to update housekeepers on additional rooms needed for housekeeping. This allows for an easy method of communication among staffs.
* **Finance management** display the total amount of money made in that day, transactions, taxes and earnings in a chart for easier identifications of sales. This reduces the time spent for accountants when recording profits.
* **Database** to storedata of the HMS. The database being used would be a MySQL database.

### 

### **1.2 Assumptions and constraints**

**Assumptions**

* Completion of project 1 week before deadline
* No upgrading of software needed.
* Software stated is sufficient to complete project
* High competition in the market

**Constraints**

* Project to be completed in 3 weeks’ time
* Unable to change design once completed
* Limited meetup days

### **1.3 Definitions and acronyms**

HMS - Hotel Management System

DOB - Date of birth

NRIC - National Registration Identity Card

CRUD- Create, read, update, delete

* CRUD function refers to the 4 functions implemented in a database, which permits users create, view, update and delete data.

DB – Database

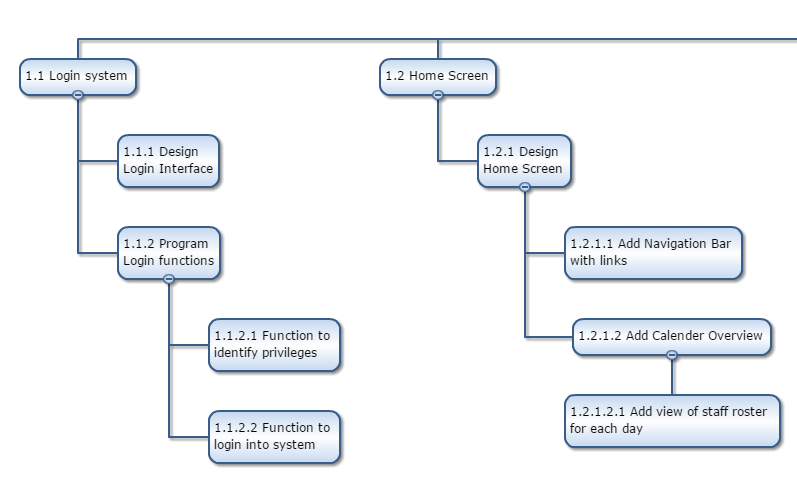
SLDC - Software Development Lifecycle

## **2. Roles and responsibilities**

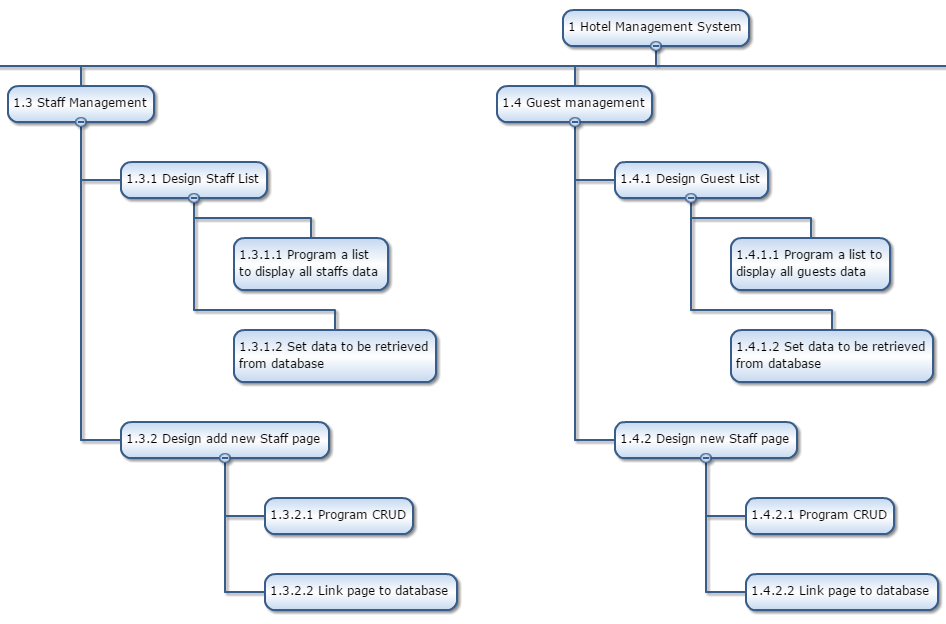
|  |  |
| --- | --- |
| Members | Workload |
| Gideon Ler | Objectives and scope  Work breakdown structure  Project schedule  Definition and acronyms |
| Tan Hai Kang | Budget Summary  Roles and responsibilities  Definitions and acronyms  Objective and scope |
| Eugene Sim | Assumption & Constraint  Risk Management Plan |
| Brandon Chew | Work breakdown structure  Project schedule  Risk Management Plan  Compilation |

## **3. Estimates and project schedule**

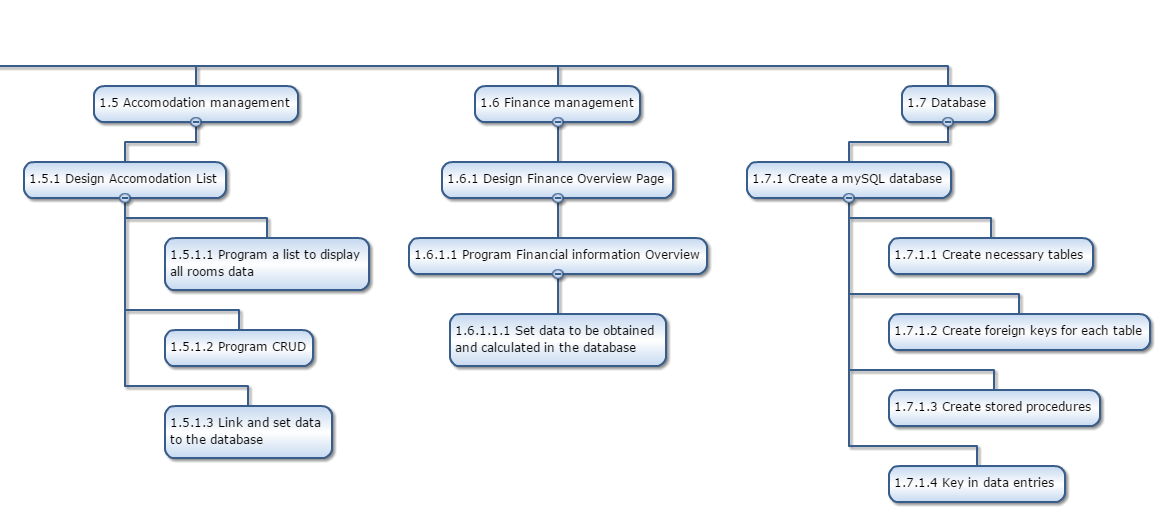
### **3.1. Work breakdown structure**



(Figure 3.1 shows the first login page and the home page of the system)

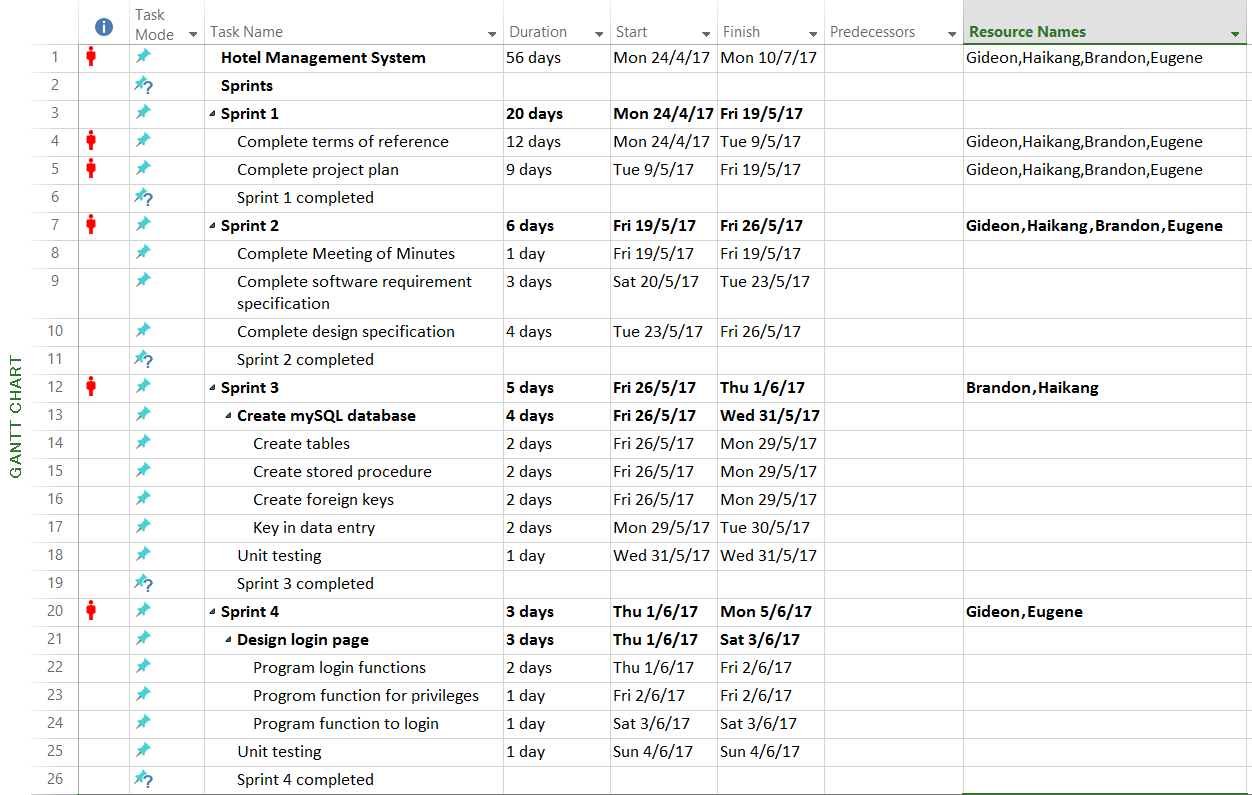


(Figure 3.2 shows the staff page and the guest management page)

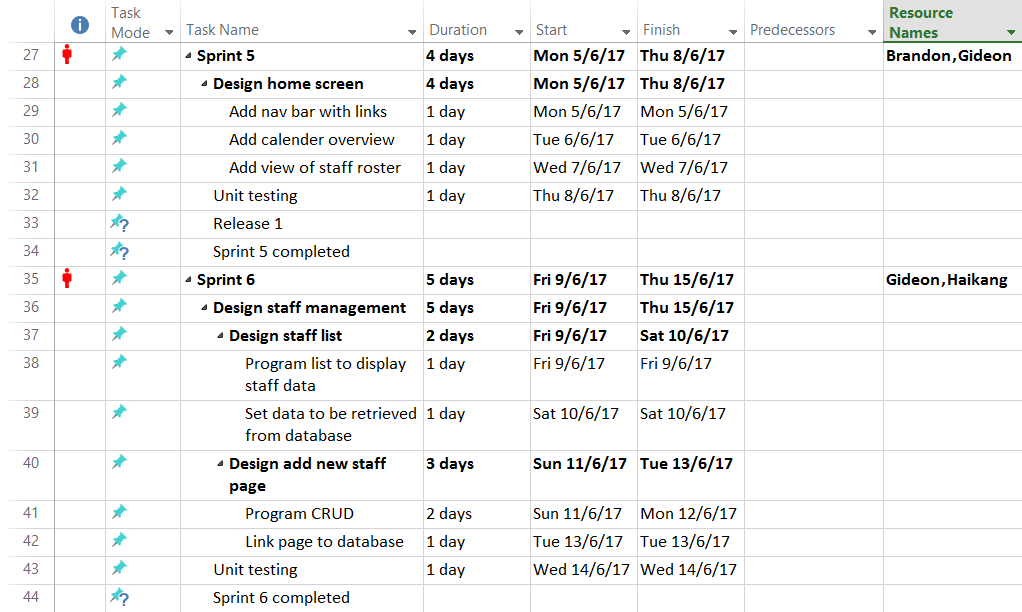


(Figure 3.3 shows the accommodation and finance management page. As well as the steps to create the database)

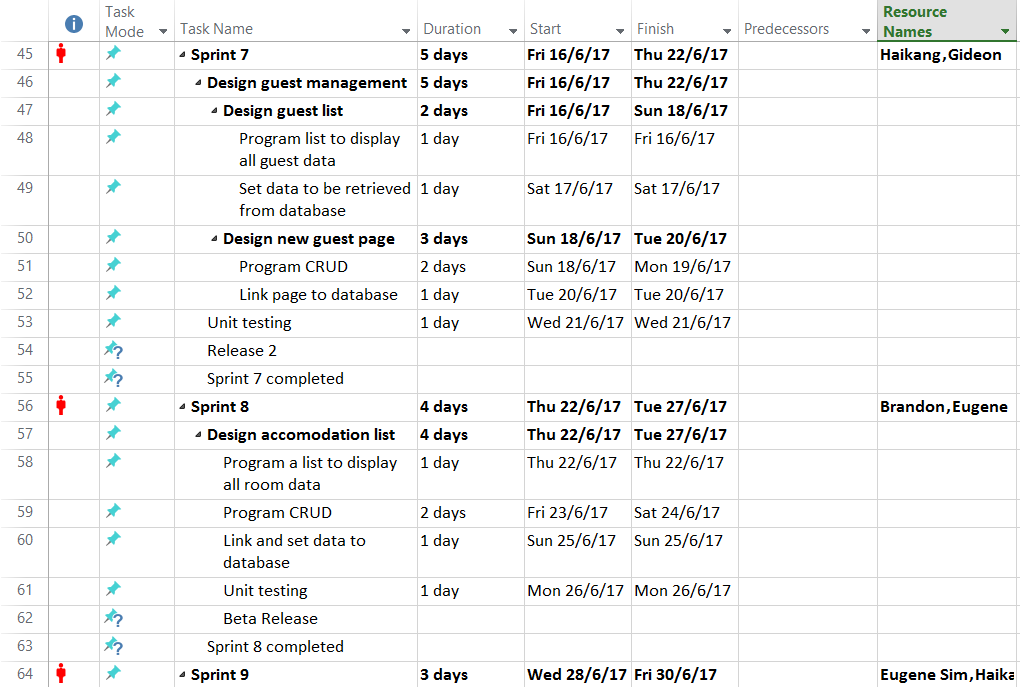
**3.2. Project Schedule**



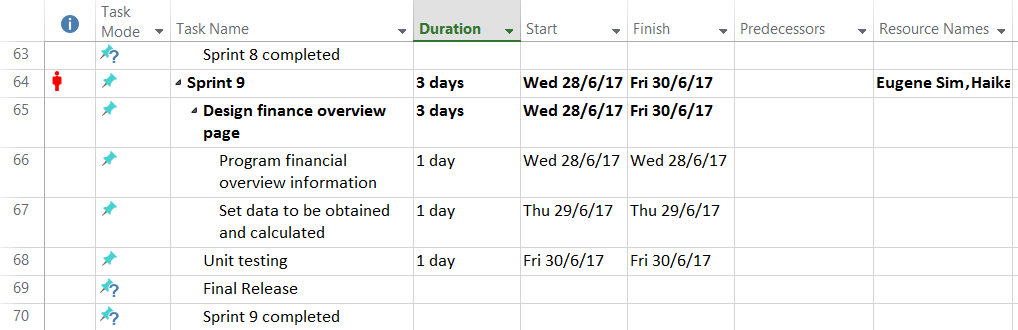
(Figure 3.4 shows the first 4 sprints in the Project Schedule)



(figure 3.5 shows sprint 5 and 6 and the first release)



(Figure 3.6 shows sprint 7 and 8 with the next two releases)



(Figure 3.7 shows sprint 9 with the final release)

### **3.3. Budget Summary**

*Table for Manpower cost*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *No.* | *Name* | *Project duration* | *Salary per month (SGD)* | *Total Salary*  *(SGD)* |
| *1.* | *Brandon Chew - Agile project manager* | *2month* | *6500* | *13000* |
| *2.* | *Gideon Ler - C# Software engineer* | *2month* | *4000* | *8000* |
| *3.* | *Tan Hai Kang - C#Software engineer* | *2month* | *4000* | *8000* |
| *4.* | *Eugene Sim - C#Software engineer* | *2month* | *4000* | *8000* |
|  | *TOTAL* |  |  | *40,000* |

*Table for Hardware and Software cost*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *No.* | *Item* | *Unit* | *Unit cost*  *(SGD)* | *Total Cost*  *(SGD)* |
| *1.* | *PCs/ Servers*   * *1 as server with Win2k8 OS* * *3 as desktop with Win8 OS* | *4* | *1500* | *6000* |
| *2.* | HP PageWide Pro 577z Multifunction Printer | *2* | *1750* | *3500* |
| *3.* | *Microsoft Project 2016* | *1* | *830* | *830* |
| *4.* | *Microsoft Visual Studio enterprise* | *1* | *4000* | *4000* |
| *5.* | *Rational functional tester (IBM)* | *1* | *1671.75* | *1671.75* |
| *6.* | *Macromedia dreamweaver 8.0* | *1* | *280* | *280* |
| *7.* | *MySql enterprise Edition* | *1* | *5000* | *5000* |
| *8.* | *Keyboard* | *4* | *10* | *40* |
| *9.* | *Mouse* | *4* | *10* | *40* |
|  | *TOTAL* |  |  | *19,857.75* |

Total budget needed: $40000 + $19,857.75= $59,857.75

## **4. Risk Management Plan**

### **Possible risks & Reduction strategies**

* **System may be non-reusable [Business Risk]**

The system is designed under the needs & wants of Mr & Mrs Wang’s Hotel. Therefore, the programmed system may have unrequired needs & wants from other hotels which may results in the system to not be able to be reused again. One way to reduce the risk is by accepting it. This is because that this risk is expected to happen for any system that is programmed.

* **Over budget and Late deliverable [Business Risk]**

This is possible due to the fact that the team is inexperienced with using the agile method. The Agile method requires member to be adaptable and think independently due to minimal documentation. Hence, in order to overcome this problem, team members has to communicate with one another to ensure that everyone is on the right track. This will prevent additional costs and time.

* **Lack of customer interaction [Project Risk]**

As we are using the agile method for our methodology, it depends heavily on customer interaction. If any instructions were given unclear, the final product may turn differently from how it is expected to be. One way to reduce the severity of this problem is to mitigate it by analysing different hotel management systems. This will improve our understanding of different functions and will improve the final product.

* **Conflict between team members [Project Risk]**

Different people have different ideal perspective of the finalize project. Voicing harsh comments on the project might cause tension between the team members and thus starting a conflict. One strategy to overcome this risk is to mitigate the situation. By taking action such as compromising with one another will help reduce the negative impact of the risk.

* **Misunderstanding of project requirements [Project Risk]**

As English has vast amount of meaning, misunderstanding words are common when

reading specification, which therefore, working on a wrong project. However, by changing the scenario after confirmation of the project specification, the team will be able to avoid this risk.

* **Inexperienced team members [Project Risk]**

With a growing number of fresh graduates into this industry, inexperienced members often make for mistake which lead to decrease in productivity. Thus, it is often avoided by changing the inexperienced member to a more experienced member at a higher cost

* **Out of support for legacy components [Technical Risk]**

As many people prefer to use a components which they are comfortable with, the support for these components would eventually end when is deemed as obsolete. This would result in low completion rate or even inability to complete the work. However, by picking up modern software, this will increase the flexibility of the individual worker and also able to complete more tasks with different variation.

### **Likelihood of the risks**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type of Risk** | **Severity of Impact** | **Likelihood of Occurrence** | **Risk Exposure** |
| a) Over budget & Late delivery | High(3) | Medium(2) | 6 |
| b) Lack of customer interaction | High(3) | Medium(2) | 6 |
| c) Out of support components | High(3) | Medium(2) | 5 |
| d) Inexperience members | Medium(2) | Medium(2) | 4 |
| e) Misunderstood requirements | High(3) | Low(1) | 3 |
| f) Non reusable System | Low(1) | High(3) | 3 |
| g) Team member conflicts | Medium(2) | Low(1) | 2 |
|  |  |  | 29 |