|  |  |
| --- | --- |
| **Logo_FPT_University_doc** | **MINISTRY OF EDUCATION AND TRAINING** |

**FPT UNIVERSITY**

**CAPSTONE PROJECT DOCUMENT**

**Fstay**

**Report #2 – Software Project Management Plan**

|  |  |
| --- | --- |
| **SWP490\_G7** | |
| **Group Member** | Phạm Minh Dương – SE05785 |
| Lý Văn Thiên – SE05754 |
| Trần Đức Anh – SE06025 |
| Võ Đại Dương Phi – SE05784 |
| Phạm Mạnh Đạt – SE05386 |
| **Supervisor** | Mr. Nguyễn Văn Sang |
| **Project Code** | Fstay |

- Hanoi, 07/2020 -

Table of Contents

[1 INTRODUCTION 4](#_Toc49467960)

[1.1 Purpose 4](#_Toc49467961)

[1.2 Definition and Acronyms 4](#_Toc49467962)

[1.3 References 4](#_Toc49467963)

[2 PROJECT OVERVIEW 4](#_Toc49467964)

[2.1 Project Description 4](#_Toc49467965)

[2.2 Scope 4](#_Toc49467966)

[2.3 Standard Objectives 5](#_Toc49467967)

[2.4 Milestones and Deliverables 5](#_Toc49467968)

[3 PROJECT ORGANIZATION 6](#_Toc49467969)

[3.1 Software Process Model 6](#_Toc49467970)

[3.2 Project Life Cycle 7](#_Toc49467971)

[3.3 Roles and Responsibilities 8](#_Toc49467972)

[3.3.1 Team Organization Structure 8](#_Toc49467973)

[3.3.2 Roles and Responsibilities 8](#_Toc49467974)

[4 TOOLS AND INFRASTRUCTURES 9](#_Toc49467975)

[4.1 Hardware 9](#_Toc49467976)

[4.2 Software 9](#_Toc49467977)

[5 SCHEDULES 10](#_Toc49467978)

[5.1 Detailed Schedules 10](#_Toc49467979)

[5.2 Meeting minutes 12](#_Toc49467980)

[6 RISKS MANAGEMENT 13](#_Toc49467981)

[7 CODING CONVENTION 15](#_Toc49467982)

[7.1 Code Layout 15](#_Toc49467983)

[7.1.1 Indentation 15](#_Toc49467984)

[7.1.2 Maximum Line Length 15](#_Toc49467985)

[7.1.3 Blank Lines 15](#_Toc49467986)

[7.2 Imports 15](#_Toc49467987)

[7.3 Comments 16](#_Toc49467988)

[7.3.1 Block Comment 16](#_Toc49467989)

[7.3.2 Inline Comment 16](#_Toc49467990)

[7.4 Naming Conventions 16](#_Toc49467991)

[7.5 Wrapping Lines 16](#_Toc49467992)

# INTRODUCTION

## Purpose

This document is the project management plan of Fstay Project – our Capstone Project in FPT University. It is included the project overview, project organization, tools and infrastructures, schedule, risk management, quality management and some coding convention of this project.

## Definition and Acronyms

|  |  |
| --- | --- |
| Acronym & Abbreviation | Definition |
| Fstay | Fstay Homestay Management System |
| ERD | Entity Relationship Diagram |
| SRS | Software Requirement Specification |
| SAD | Software Architecture Design |
| FU | FPT University |
| RUP | Rational Unified Process |

Table 1-1: Definitions and Acronyms

## References

[1] Report #1 – Fstay Project Introduction v1.0.

[2] CapstoneProject\_Report2\_Template.docx.

[3] Wikipedia, 2020, IBM Rational Unified Process, viewed 02 June 2020, <https://en.wikipedia.org/wiki/Rational_Unified_Process>

# PROJECT OVERVIEW

## Project Description

Nowadays, the idea of creating a website for homestay renting is not new; there are many existing websites share this idea now. However, each website has its own advantages and disadvantages. Our target when developing this system is included some main functions of current websites and add to some special features that limited in existing website. It provides intuitive interface, Vietnamese language supporting, ease to use. Therefore, the project group aims at bringing a better, faster and easier website that suitable for a very active homestay service in Vietnam.

## Scope

Due to the time constraint of a capstone project, this project scope is included:

* Developing user requirement and software requirement specification.
* Developing architecture and detailed design documents.
* Coding and unit test.
* Developing test case and execute system test.
* The developing software will have these following functions:

|  |  |  |
| --- | --- | --- |
| No | Group of functions | Function |
| 1 | Common module for customers (renter and owner) | Account Management |
| 2 | Renter Module | Renter – Homestay Management |
| Renter – Order Management |
| 3 | Owner Module | Owner – Account Management |
| Owner – Homestay Management |
| Owner – Order Management |
| 4 | Admin Module | Admin – Account Management |
| Admin – Renter Management |
| Admin – Owner Management |
| Admin – Homestay Management |
| Admin – Promotion Management |
| Admin – Utility Management |
| Admin – Commission Fee Management |
| Admin – Stafff Management |

Table 2-1: Function group

## Standard Objectives

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Metrics | Unit | Committed | Actual | Deviation |
| Start Date | dd-mm-yyyy | 18-05-2020 | 18-05-2020 | 0 day |
| End Date | dd-mm-yyyy | 28-08-2020 | 28-08-2020 | 0 day |
| Duration | Elapsed days | 102 days | 102 days | 0 day |
| Maximum Team Size | Person | 5 | 5 | 0 |

Table 2-2: Project Standard Objectives

## Milestones and Deliverables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Stage | Deliverable/Milestone | Delivery Date | Delivery Location |
| 1 | Inception | Business case | 20-May-2020 | Team |
| 2 | Inception | Feasibility study | 22-May-2020 | Team |
| 3 | Inception | Complete draft screen prototypes | 28-May-2020 | Supervisor |
| 4 | Inception | Deliver Report #1 | 04-Jun-2020 | Supervisor |
| 5 | Inception | Project plan | 22-Jun-2020 | Supervisor |
| 6 | Inception | Deliver Report #2 | 22-Jun-2020 | Supervisor |
| 7 | Elaboration | User requirement specification | 22-Jun-2020 | Supervisor |
| 8 | Elaboration | Complete ERD | 22-Jun-2020 | Supervisor |
| 9 | Elaboration | Final prototype | 22-Jun-2020 | Supervisor |
| 10 | Elaboration | Database Model | 22-Jun-2020 | Supervisor |
| 11 | Elaboration | Software requirement specification | 22-Jun-2020 | Supervisor |
| 12 | Elaboration | Deliver Report #3 | 22-Jun-2020 | Supervisor |
| 13 |  | Deliver Report #4 | 22-Jun-2020 | Supervisor |
| 14 | Construction | Coding training Report | 22-Jun-2020 | Supervisor |
| 15 | Elaboration | Software Architecture Design | 22-Jun-2020 | Supervisor |
| 16 | Construction | Complete Coding | 22-Jul-2020 | Supervisor |
| 17 | Construction | System Test Package | 22-Jul-2020 | Supervisor |
| 18 | Construction | Deliver Report #5 | 22-Jul-2020 | Supervisor |
| 19 | Construction | Deliver Report #6 | 22-Jul-2020 | Supervisor |
| 20 | Transition | Project Review – Lesson Learned | 22-Jul-2020 | Supervisor |
| 21 | Transition | The last document and CD source code | 22-Aug-2020 | FU |
| 22 | Transition | Project completed | 10-Aug-2020 | FU |

Table 2-3: Milestones and Deliverables

# PROJECT ORGANIZATION

## Software Process Model

Due to the short time of project and the volume of work that we have to do, after discussing, our project team chooses Iterative and Incremental Software Process as our software project model.

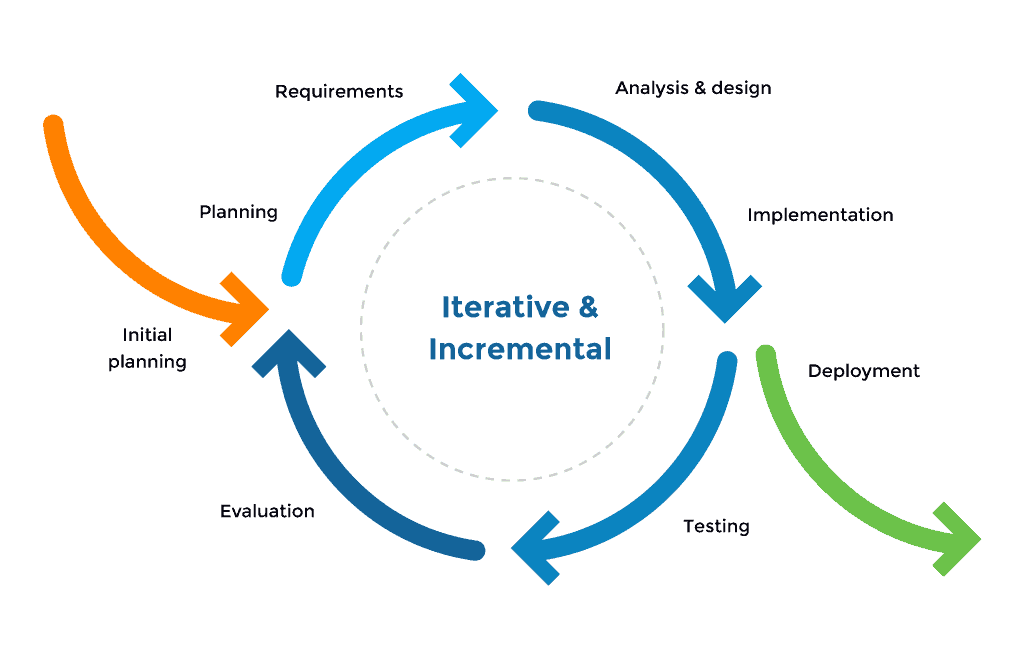


Figure 3-1: Iterative and Incremental Software Process Model

The Iterative and Incremental Software Process Model is mostly used when the scope of the

project is big, the major requirements are defined clearly, some more details will be added later in software development. By using this software process model, we break down the developing system task into series of smaller tasks which will be completed separately, allowing us to take advantage of what was learned during development of earlier parts of the system. In addition, the iterative model is easier than other models when the issues are discovered. They are fed back to the team, and solutions will be found while the project is still in development.

## Project Life Cycle

Project lifecycle has four phases:

* Inception Phase: Inception is the first phase of process. In this startup phase, we should provide business case of the system and determine the scope of project. Besides, we have to create the project management plan that has project schedule, effort estimation and risk management etc. At the end of this phase, we should check the objectives of project and decide whether to continue development or not. Hence, Inception phase must be planned and be done. Based activities of this phase:
* Study business case and feasibility study of project
* Complete draft ERD of system.
* Complete draft screen prototypes.
* Complete draft requirements.
* Determine project scopes.
* Complete project management plan.
* Elaboration Phase: The objectives of this phase are to determine appropriate architectural and construction plan for the project. The architectural decision needs to be made for the entire system, and to describe most of the requirements of system. At the end of this phase, we must examine the objectives and scopes, the choice of architecture and decide whether to proceed to the next phase. Based activities of this phases:
* Complete user requirement specification.
* Complete ERD, final prototypes.
* Complete software requirement specification.
* Complete database model.
* Complete system architecture design.
* Construction Phase: Construction is the third phase of RUP lifecycle. In this phase, we must have done all the coding and testing work. After coding, developers will do unit test themselves, then test team will do functional test and regression test when finishing all. Based activities of this phase:
* Complete coding and unit test.
* Complete functional and user interface test.
* Complete user manual.
* Transition Phase: Transition is the final phase of the RUP lifecycle. In this phase, project team has to deploy the application and give it to users. The next step is receiving feedback from users to identify the problems and then complete the system. Based activities of this phase:
* Deploy the system.
* Deliver source code.
* Complete all reports and documents.

## Roles and Responsibilities

### Team Organization Structure

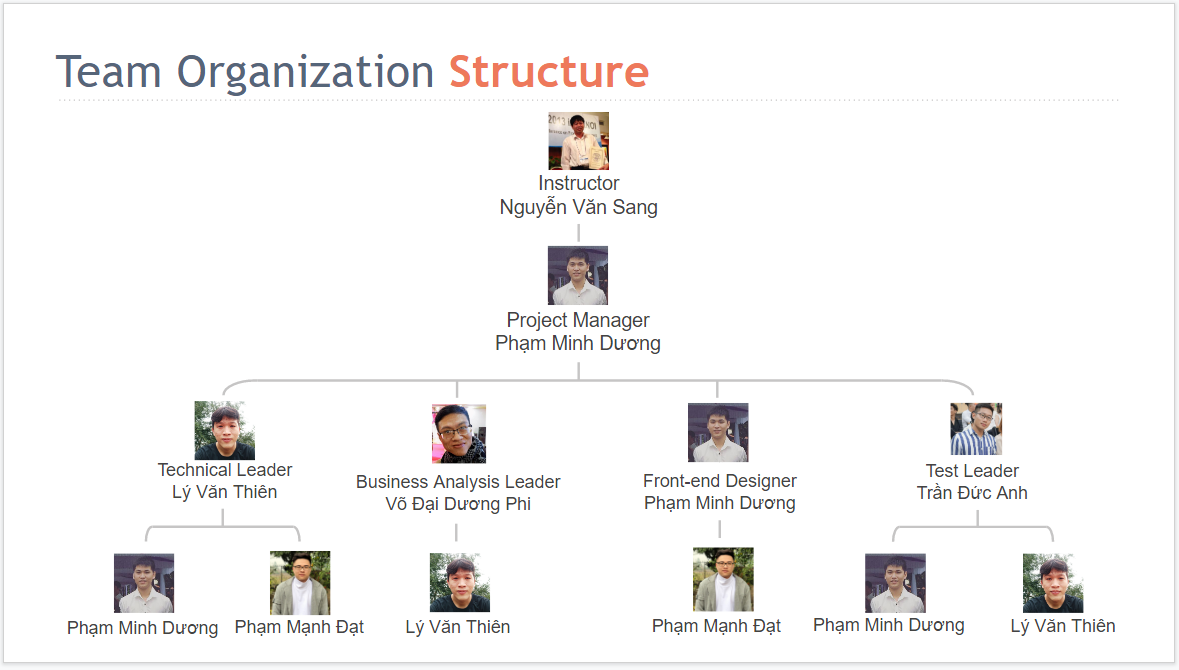


Figure 3-2: Team organization Structure overview

### Roles and Responsibilities

|  |  |  |
| --- | --- | --- |
| Team Member | Role & Responsibility | |
| Role | Responsibility |
| Pham Minh Duong | Project Manager | Planning, developing schedules, coordinating communication, generally responsible for keeping the team’s focus on the main goal |
| Developing Leader | Responsible for choosing and deciding what technologies should be used, as well as for overseeing the work being done by other develop |
| Business Analysist | Analyzes an organization or business domain and documents its business or processes or systems |
| Ly Van Thien | Documentation Leader | Resonsible for planning plan, dividing tasks and schedules of reports, documentation. |
| Business Analysist | Analyzes an organization or business domain and documents its business or processes or systems |
| Developer | Involve to code the product and review code of other developers |
| Pham Manh Dat | Design Leader | Responsible for designing product’s user interface |
| Documentation Member | Involve to write documentation |
| Tester | Responsible for involving test the product |
| Developer | Involve to code the product |
| Tran Duc Anh | Testing Leader | Responsible for test execution, including test set-up and test run, evaluation of test run and error recovery, defect logging and test results recording |
| Documentation Member | Involve to write documentation |
| Design Member | Involve to design product’s user interface |
| Developer | Involve to code the product |
| Vo Dai Duong Phi | Business Analysis Leader | Analyzes an organization or business domain and documents its business or processes or systems; ensure the product meets the certain standards of quality from requirements |
| Design Member | Involve to design product’s user interface |
| Tester | Responsible for involving test the product. |

Table 3-1: Roles and Responsibilities

# TOOLS AND INFRASTRUCTURES

## Hardware

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Purpose | Detail |
| 1 | Laptop Acer/Dell/Asus | Developing  Testing  Deploying | RAM: 8Gb  Hard disk: 256Gb  Processor: Core i7 2.50 GHz |
| 2 | Server  Computer | Keeping project’s resource | Service from Google Code:  <https://github.com/> |
| 3 | Dell-Inspiron D5559 | Testing  Deploying | RAM and Processor are fixed for each specific device.  Memory Storage: 16Gb or higher |

Table 4-1: Hardware

## Software

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | | Version |
| Software | Microsoft Office | | 2016 |
| Microsoft Project | | 2016 |
| Microsoft Visio | | 2016 |
| Operating System | Microsoft Windows | 10 |
| Balsamiq Mockups | | 2.2.4 |
| Framework | Spring Boot | | 8 |
| Visual Code | | 8 or higher |

Table 4-2: Software

# SCHEDULES

## Detailed Schedules

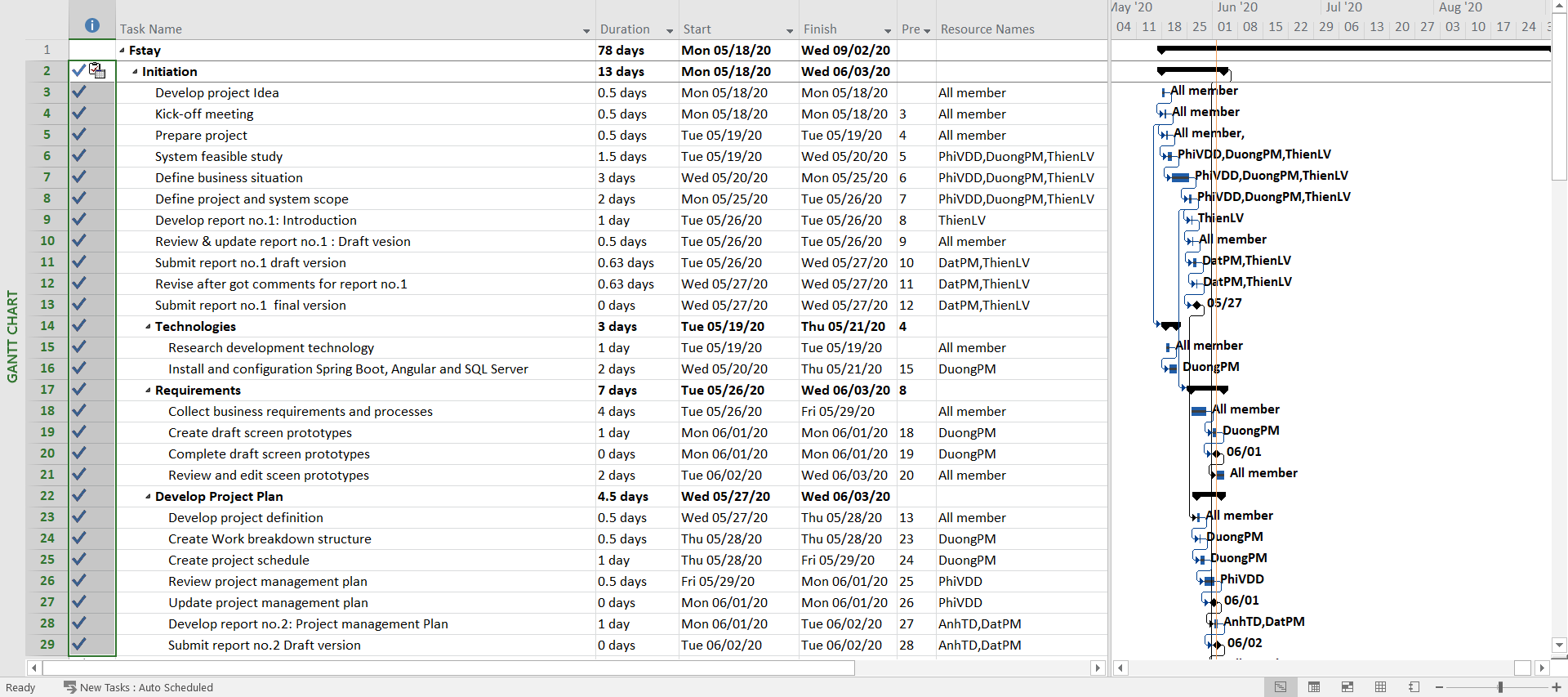


Figure 5-1: Detailed schedule 1

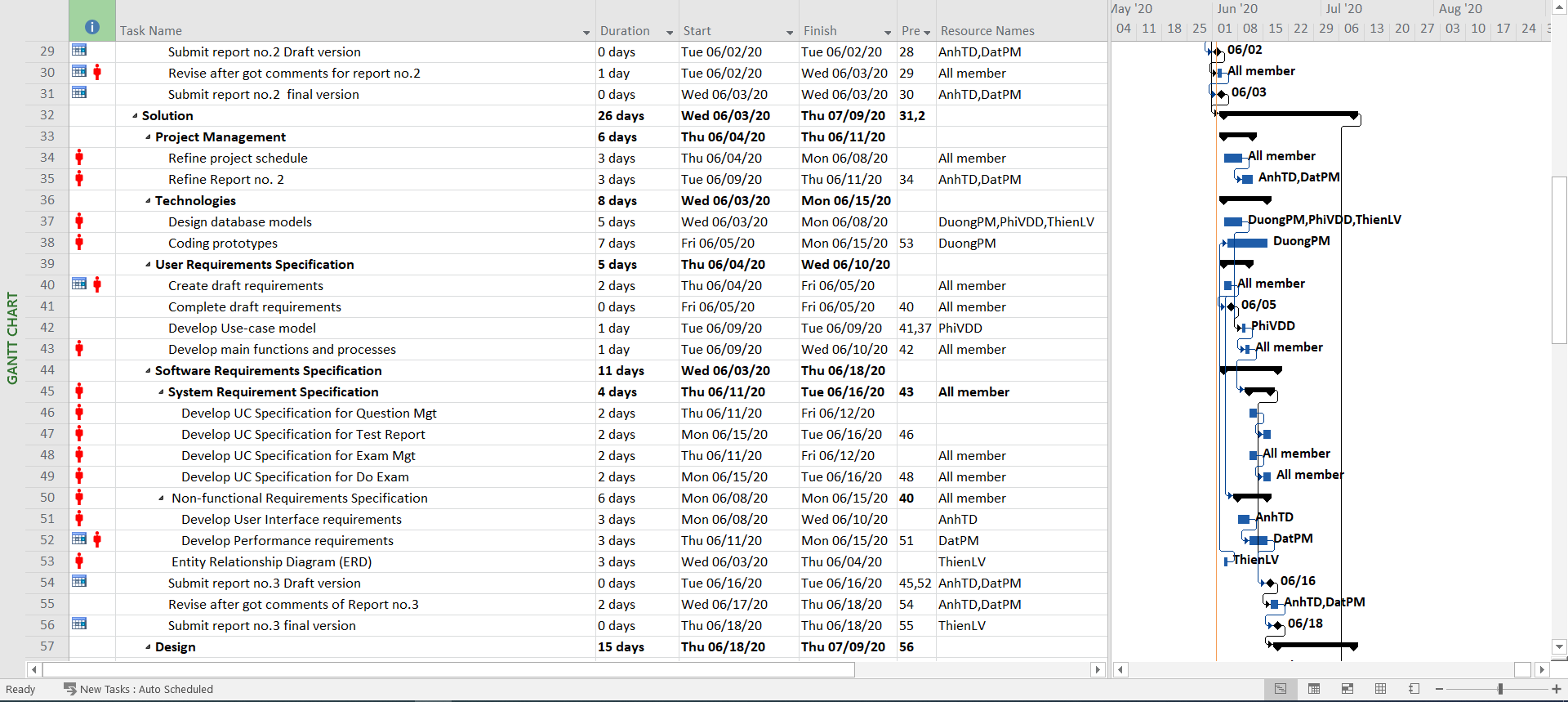


Figure 5-2: Detailed schedule 2

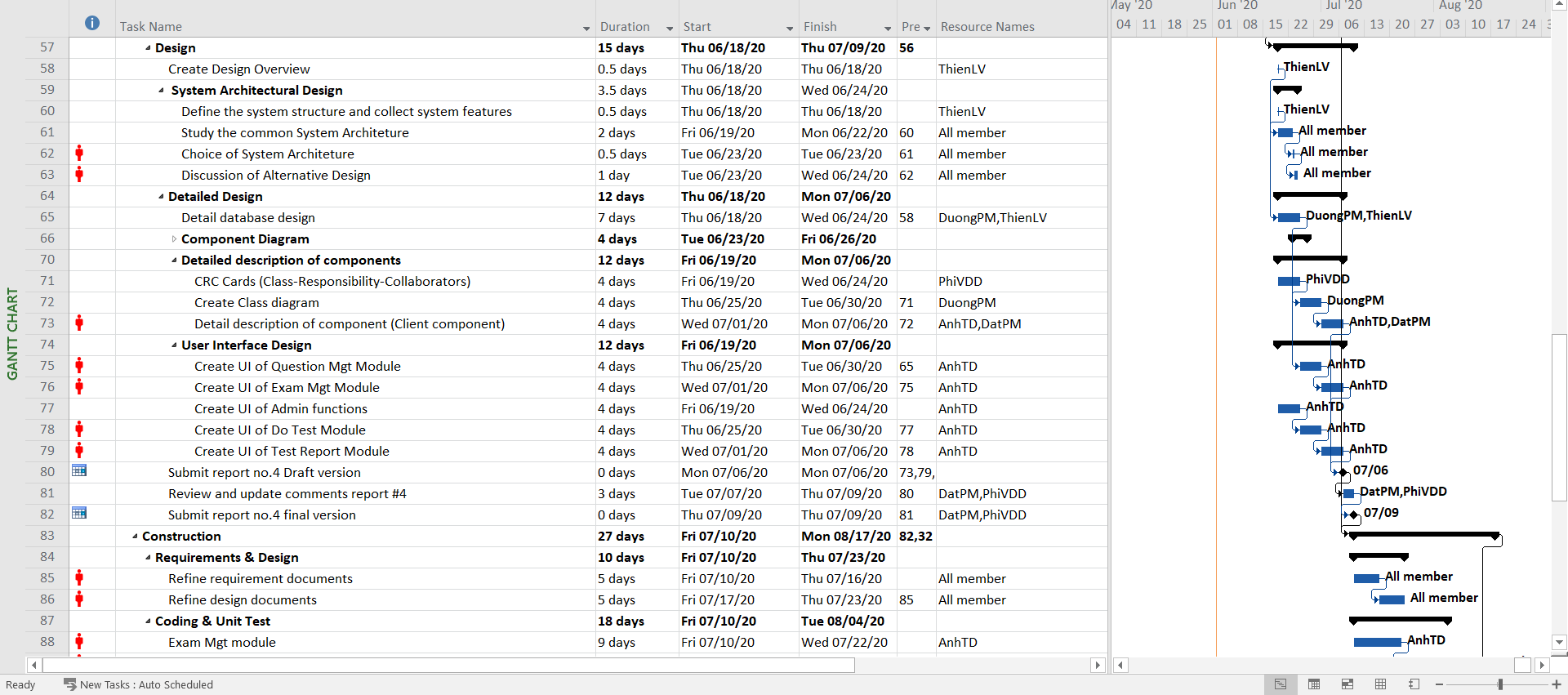


Figure 5-3: Detailed schedule 3

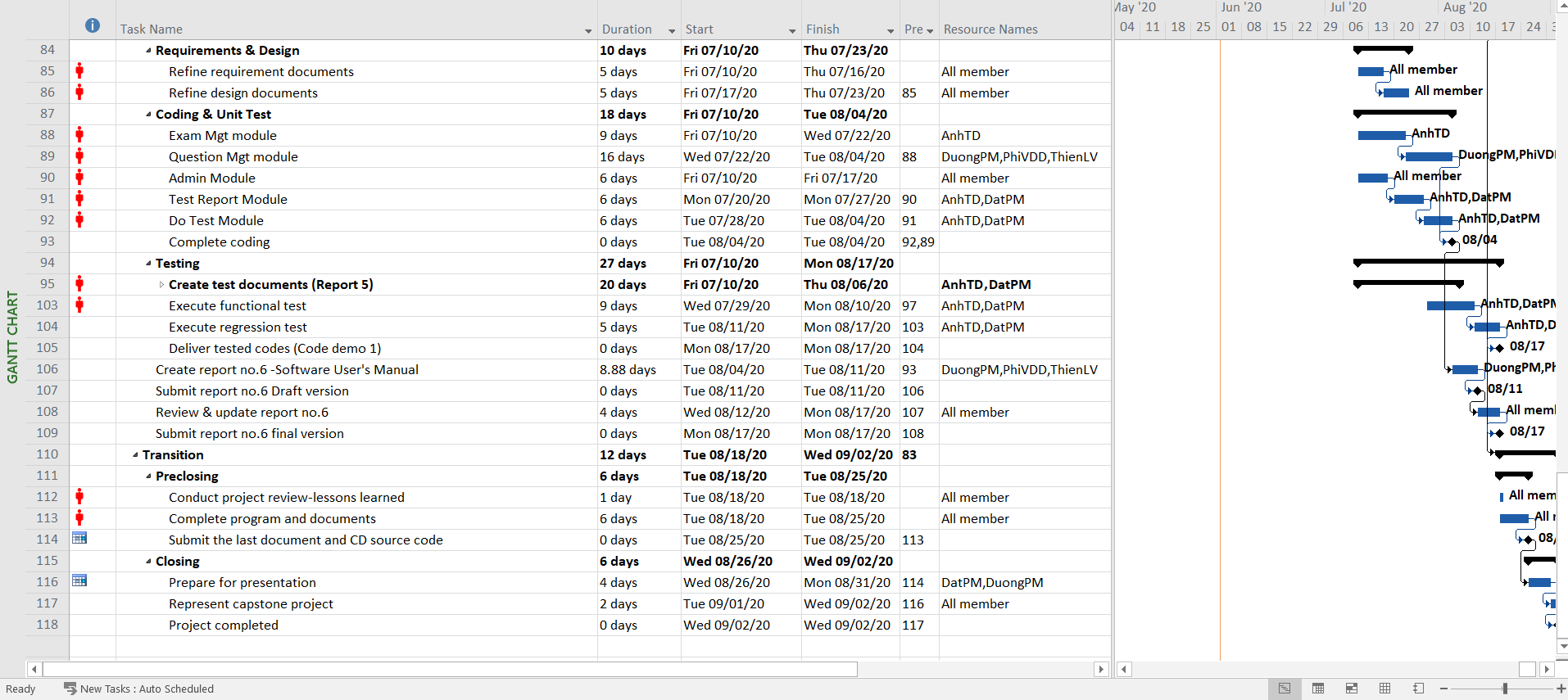


Figure 5-4: Detailed schedule 4

## Meeting minutes

During our developed time, we have two types of meeting:

* Meeting with supervisor: we have meeting with our supervisor once a week in 1.00PM Thursday. In this meeting, we will report what we have done in last week, raise issues to discuss with supervisor and supervisor will give us advices to deal with these problems.
* Meeting with team member: we have team meeting five times a week (Monday, Tuesday, Wednesday, Thursday and Saturday) from 8:00AM to 11:00AM. In this meeting, each member will report what he has done last week to others and the problems he meets (if have). All members will discuss about these problems and PM or technical leader will give the final solution for each. Finally, PM will give the tasks for all members to do and report in the next meeting.

Here is the meeting minute template will be used for the project:

M E E T I N G M I N U T E S

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Project: | | Fstay | | Project Code: | | Fstay | | |
| Project Manager: | | DuongPM | | Conductor: | | Mr.SangNV | | |
|  | | Secretary: | | DuongPM | | |
| Date: | | 21-05-2020 | | Time: | | 1:00PM – 2h20PM | | |
| Venue: | | Room Alpha-401L | | | | | | |
| Topic of meeting: | | Weekly meeting | | | | | | |
| Attendees: | | | | | | | | |
| No | Full name | | Unit/Group | | Position | |  | Attendance |
| 1 | Nguyen Van Sang | |  | | Supervisor | | Present | |
| 2 | Pham Minh Duong | |  | | PM | | Present | |
| 3 | Vo Dai Duong Phi | |  | | Dev | | Present | |
| 4 | Ly Van Thien | |  | | Dev | | Present | |
| 5 | Pham Manh Dat | |  | | Dev | | Present | |
| 6 | Tran Duc Anh | |  | | Dev | | Present | |

Table 5-1: General information of meeting minute

|  |  |  |
| --- | --- | --- |
| Key Points Discussed | | |
| No | Topic | Highlights |
| 1. | Team member problems | All the activities, the issues will be discussed and reported with supervisor in last week.  Discuss all issues & problems on the job last week  Discuss the working plan next week |
| 2. | Project introduction | Instruct the direction for project team to deal with the problems  Support & comments the working plans next week |

Table 5-2: Key points discussed of meeting minute

|  |  |  |  |
| --- | --- | --- | --- |
| Action Plan | | | |
| No |  | Owner | Target Date |
| 1 | Business Case | DuongPM | 23-May-2020 |
| 2 | Feasibility study | PhiVDD | 26-May-2020 |

Table 5-3: Action plan of meeting minute

# RISKS MANAGEMENT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Risk Type | Risk Description | Effect | Potential Response |
| **#** | **People Risks** | | | |
| 1 | Over deadline (Cause by team member) | - Team member is not hard working, do not know what to do in their task, do not finish their tasks before deadline.  - The task and schedule is overload for members. | SERIOUS | - Leader must review carefully the project, give tasks for member clearly, surely they know exactly what they have to do.  - Members who miss deadlines must be punished according to the team rules. |
| 2 | Over deadline  (Cause by team member) | Team member are sick, have personal work so that they cannot complete task under deadline | SERIOUS | - Allow member have break day to restore health, solve their personal work so that they can focus on project. |
| 3 | Unrealistic project schedule | Leader creates a unrealistic plan so that members can not follow it to do their tasks | NORMAL | -Meeting with all team to create project schedule more realistic.  -Leader should learn more about creating a project schedule. |
| 4 | Communication risks | Members misunderstand others, leads to the wrong actions and decisions | SERIOUS | - All members must talk to each other more often, careful in writing reports or email.  - Leader need to discuss about member’s issues in every meeting so that he can solve members problem immediately. |
| 5 | Conflictions between team members. | Members have argued, conflicted with others, leads to stressful working environment | SERIOUS | -Set up open talk between all members to share about their uncomforted things.  -Set up team-building for members to relax and better understand each other. |
| 6 | Poor experience | It’s makes plan late.Study new technologies have many difficult to apply for project | SERIOUS | - Encourage members to study harder about new technologies, sharing about it at the meeting to develop better project together.  -List tasks and check continuously. |
| **#** | **Technical Risk** | | | |
| 7 | Web technique knowledge | Team have not much knowledge in the framework and technique of web. Therefore, team have to study all of these things from the beginning. This work may takes a lot of times or team may not resolve some technical problems. | SERIOUS | -All members must research for web technique, sharing knowledge for others, exchange information and problem.  - Send technical issues to supervisor who has experience to get support |
| 8 | Eclipse | Eclipse is very error-prone. | SERIOUS | - Record the occurred error and how to fix it. |
| **#** | **Structure/ Process Risk** | | | |
| 9 | Project scope, task and risks | Underestimate project scope, tasks’ difficulty level and risks’ effectiveness. | SERIOUS | - Estimate project scope with supervisor and experience persons.  - Assign task weight value to make task evaluation easier. Discuss in group about tasks’ difficulty level.  - Involve all team members in risk management process and reference to instructor’s opinions. |
| **#** | **Requirement Risk** | | | |
| 10 | Not understanding requirement | Not understanding the system'sprocess, so we can have mistakes in describing the essential functions | SERIOUS | - Develop prototypes and review prototypes with experts and supervisor |

Table 6-1: Risks Management

# CODING CONVENTION

## Code Layout

### Indentation

* Use 4-spaces tab per indentation level.
* Never mix tab and space.

### Maximum Line Length

Limit all lines to a maximum of 79 characters.

*There are still many devices around that are limited to 80 character lines; plus, limiting windows to 80 characters makes it possible to have several windows side-by-side. The default wrapping on such devices disrupts the visual structure of the code, making it more difficult to understand. Therefore, please limit all lines to a maximum of 79 characters.*

### Blank Lines

* Separate top-level function and class definitions with two blank lines.
* Method definitions inside a class are separated by a single blank line.
* Extra blank lines may be used (sparingly) to separate groups of related functions.
* Use blank lines in functions, sparingly, to indicate logical sections.

## Imports

* Imports should usually be on separate lines, e.g.:

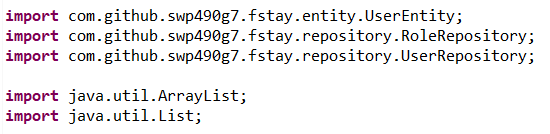


Figure 7-1: Import example

* Imports are always put at the top of the file.
* Imports should be grouped in the following order:
* Standard library imports
* Related third party imports
* Local application/library specific imports

Notes: *Put a blank line between each group of imports.*

## Comments

### Block Comment

Block comments generally apply to some (or all) code that follows them, and are indented to the same level as that code. Paragraphs inside a block comment are separated by /\* and \*/.

### Inline Comment

Use inline comments sparingly. An inline comment is a comment on the same line as a statement. Each line of a block comment starts with a //.

## Naming Conventions

* Classes and Interfaces:
* Class name should be **nouns**, in mixed case with the **first** letter of each internal word capitalized. Interfaces name should also be capitalized just like Class name.
* Use whole words and must avoid acronyms and abbreviations.
* Methods: Methods should be **verbs**, in mixed case with the **first letter lowercase** and with the first letter of each internal word capitalized.
* Variables:
* Should **not** start with underscore (‘\_’) or dollar sign ‘$’ characters.
* Should be mnemonic, designed to indicate to the casual observer the intent of its use.
* **One-character variable names should be avoided** except for temporary variables.
* Common names for temporary variables are i, j, k, m, and n for integers; c, d, and e for characters.
* Constant variable:
* Should be **all uppercase** with words separated by underscores (“\_”).
* There are various constants used in predefined classes like Float, Long, String etc.
* Packages:
* The prefix of a unique package name is always written in **all-lowercase ASCII letters** and should be one of the top-level domain names like: com, edu, gov, mil, net, org.
* Subsequent components of the package name vary according to an organization’s own internal naming conventions.

## Wrapping Lines

When an expression will not fit on a single line, break it according to these general principles:

* Break after a comma.
* Break before an operator.
* Prefer higher-level breaks to lower-level breaks.
* Align the new line with the beginning of the expression at the same level on the previous line.
* If the above rules lead to confusing code or to code that's squished up against the right margin, just indent 8 spaces instead. Align the new line with the beginning of the expression at the same level on the previous line.