|  |  |
| --- | --- |
| C:\Users\Plasma-PC\Desktop\logo.jpg | **MINISTRY OF EDUCATION AND TRAINING** |

**FPT UNIVERSITY**

**Capstone Project Document**

**Olives**

Project Code: Olives  
Document Code: **Olives\_PP\_v1.0**

|  |  |  |
| --- | --- | --- |
| **Olives Team** | | |
| **Group member** | Phạm Đức Thắng | SE03055 |
| Trần Ánh Dương | SE02797 |
| Phạm Minh Tuấn | SE02875 |
| Bùi Quốc Trọng | SE03418 |
| Nguyễn Văn Hưng | SE02582 |
| Nguyễn Duy Linh | SE03150 |
| **Supervisor** | Lecturer: Phan Trường Lâm | |
| **Capstone Project code** | Olives | |

**Hoa Lac, 18th May 2016**

**CONTENT**

[**1.** **Problem Definition** 2](#_Toc452678367)

[1.1 Name of this Capstone Project 3](#_Toc452678368)

[1.2 Problem Abstract 3](#_Toc452678369)

[1.3 Project Overview 3](#_Toc452678370)

[1.3.1 The Current System 3](#_Toc452678371)

[1.3.2 The Proposed System 3](#_Toc452678372)

[1.3.3 Boundaries of the System 3](#_Toc452678373)

[1.3.4 Development Environment 3](#_Toc452678374)

[**2.** **Project organization** 4](#_Toc452678375)

[2.1 Software Process Model 4](#_Toc452678376)

[2.2 Roles and Responsibilities 5](#_Toc452678377)

[**3.** **Project Management Plan** 6](#_Toc452678378)

[**3.1** **Task** 6](#_Toc452678379)

[**3.2** **Project Schedule** 6](#_Toc452678380)

[**3.3** **Meeting Minutes** 6](#_Toc452678381)

[**4.** **Resource** 6](#_Toc452678382)

[**5.** **Risk Management** 6](#_Toc452678383)

[**6.** **Communication Management** 6](#_Toc452678384)

[**9.** **Project Infrastructure** 8](#_Toc452678385)

[9.1 Software and Techniques 8](#_Toc452678386)

[9.2 Hardware 9](#_Toc452678387)

[9.3 Other Infrastructure 9](#_Toc452678388)

[**10.** **Coding convention** 9](#_Toc452678389)

[**11.** **Version Numbering Rule** 9](#_Toc452678390)

[**12.** **Directory Structure** 10](#_Toc452678391)

[**13.** **Other Olives Rules** 11](#_Toc452678392)

1. **Problem Definition**
   1. Name of this Capstone Project
   * Project name: **Olives**
   * Project code: **Olives**
   * Product type: **Web and iOS Application**
   * Timeline: **From 9th May to 27th August**
   1. Problem Abstract

Nowadays, the humman health issues always are intersted. But hospitals services can’t meet the needs of patients. Many doctors have good specialize but they can’t approach to patients. Patients having difficulty looking for good doctors. Between them don’t have connections, the only solution is go to the hospital. Patients take long time with hospital procedures and doctors can’t take care them. Olives is a sotfware for doctors. It’s a connection between doctors and patients. Olives helps doctors take care their patients.

* 1. Project Overview
     1. The Current System

The system has not been developed previously.

The system is growing up as begin at the moment.

* + 1. The Proposed System

Objects using the system: Admin, doctors, patients

Function:

* Admin: Add doctor, View information, Disable/ Enable Doctors, Disable/ Enable Patients, Statistic business.
* Doctors: View information, Create/Update/Delete medical record, Contact, Appointment, View/Create/Update/Delete prescription, Set status of medical record.
* Patients: View information, Create/Update/Delete medical record, Contact, Appointment, View/Create/Update/Delete prescription, Set status of medical record, Find doctors, Rate doctors, Create family relation
  + 1. Boundaries of the System

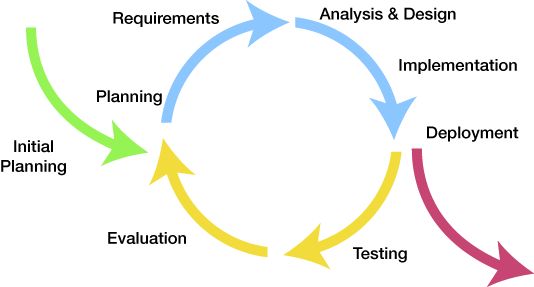
This document is for all members engage to <My Doctor>

It is used in: Software requirement, evaluation and acceptance standards are developed

* + 1. Development Environment
* Programming languages:
* C#, HTML5, Objective C
* Framework:
* AngularJS v1.2.22 (JavaScript)
* ASP.NET Framework 4.6
* ASP.NET 5
* Software Architecture:
* Web Service
* Process Model:
* Iterative and Incremental Software Process Model.
* Version Control:
* Github.
* IDEs: VisualStudio 2015, Xcode, Resharper 10.0.2 ultimate.
* DBMS:
* MySQL v5.5, Redis v3.0, MySQL Workbench v6.2.
* UML tools:
* Enterprise Architect v12, Astah Professional v6.9.0
* Hosting:
* Heroku
* AppHarbor
* Manage tools:
* Jira
* Web server:
* Apache2.0.

Other: Microsoft Office 2013, Microsoft Visio 2013, Microsoft Project 2013

1. **Project organization**
   1. Software Process Model

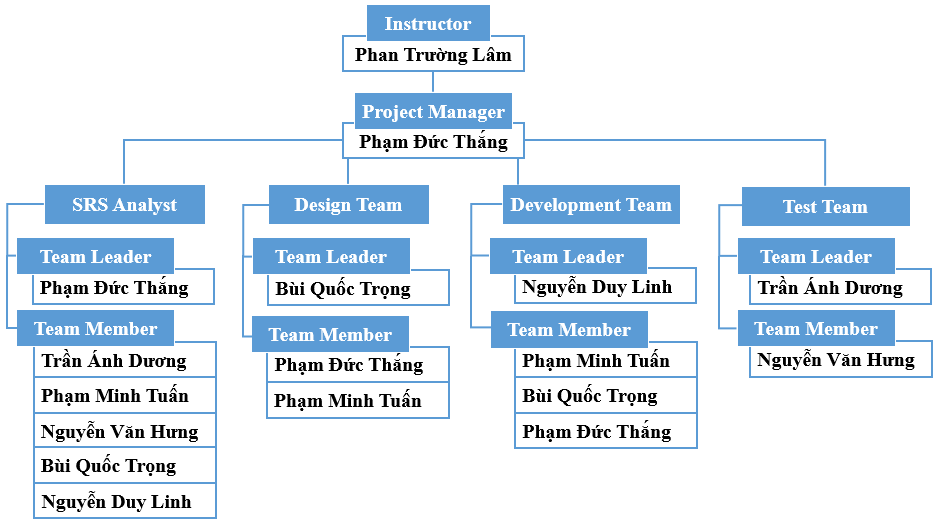
  
*Figure 2.1: Iterative and Incremental Software Process Model*

This figure above describes the information and products flow lifecycle process model. Olives project uses the Iterative and Incremental Software Process Model.

The Iterative and Incremental Software Process Model is most use when the scope of the project is big, the major requirements were defined clearly, some more detail will be added in time, and for the newbie group in software development. By using this software process model, we break down the developing system task into series of smaller tasks which be completed separately, evaluated, and subsequently re-worked until the system’s performance adequately. In addition, the iterative model is easier than other models when the issues are discovered. They are fed back to the team, and solutions found while the project is still in development.

* 1. Roles and Responsibilities

2.2.1 Organizational Structure



*Figure 2.2: Organization Structure Chart*

2.2.2 Project Role

|  |  |  |
| --- | --- | --- |
| Name | Role | Quality Responsibility |
| ThangPD | Project Management | - Planning and defining scope, developing schedules, allocating resources, coordinating communication, generally responsible for keeping the team’s focus on main goal, and tries to keep the project team focused on the right goal at a time. |
| TrongBQ | Developer, Designer | - Involve to code product.  - Involve to design product. (Build system architecture, coding. Review codes. Fix bug) |
| LinhND | Developer | - Involve to code product.  (Building common framework, Investigate solution. Build system architecture, coding. Review codes. Fix bug) |
| TuanPM | Developer | -Involve to code product. (Colecting database. To be in charge of data issues. Review code. Fix bug) |
| DuongTA | QA and Tester | - Responsible for test execution, including test set-up and test run, evaluation of test run and error recovery, defect logging and test results recording.  - Create test cases, Execute test cases  - Manage the Quality Assurance function |
| HungNV | QA and Tester | - Audits and approve project deliverables from QA perspective. Review plans and deliverables for compliance with applicable standards. Provides guidance and assistance on process matters  - Create test cases, Execute test cases |

1. **Project Management Plan**
   1. **Task**

**Planning**

**SRS**

**Design**

**Code**

**Test**

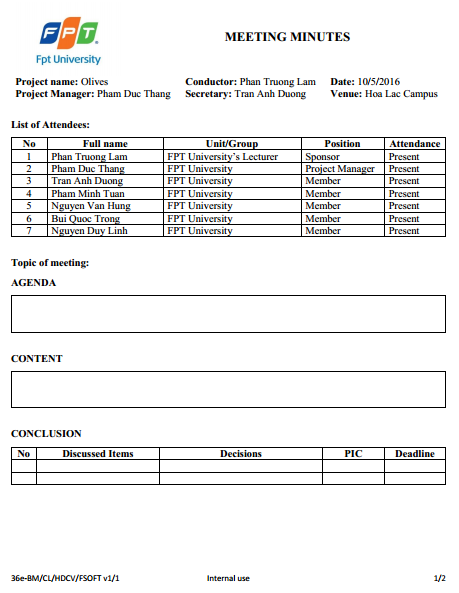
* 1. **Project Schedule**

Below are the image of task list used to assign and tracking tanks.

Refer to OL\_ProjectPlan\_v1.0.mpp to see more detail.

* 1. **Meeting Minutes**

All meeting minutes will be written follow this template:

****

*Figure 3.3.1: Meeting minutes*

1. **Resource** 
   1. Human resource

Team member

* 1. Non-human resource

Equipment:

- 1 Dell Alienware Core i7

- 2 MacBook Pro (Retina, 13-inch, Early 2015)

- 1 Dell Inspiron 7548 Core i5

- 1 Dell Inspiron 15R 5537 core i5

- 1 iPhone 6 Plus 128 GB

- 1 iPhone 6 64GB

- 1 iPhone 5C 16GB

Building: Room 105R – Beta Building – FPT University

1. **Risk Management**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Description** | **Root Cause** | **Avoidance Plan** | **Contingency plan** | **Probability** | **Impact** | **Status** |
| **R1** |  |  |  |  |  |  |  |
| **R2** |  |  |  |  |  |  |  |
| **R3** |  |  |  |  |  |  |  |
| **R4** |  |  |  |  |  |  |  |
| **R5** |  |  |  |  |  |  |  |
| **R6** |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

1. **Communication Management**
   1. Communication between Team Members

**Weekly meeting schedule:** By using Iterative and Incremental Process Model, Olives project system will be divided into a series of small tasks, each task will be assigned to team members, and depend on difficulty, Project Manager will assign deadlines for each task. Olives team will have a meeting every Monday, and Friday from 19h00 to 21h30 to report the progress of the whole team’s tasks and review together all task and problem in project. Any member who doesn’t finish his/her task (without reasonable explanation) will be fined. The issues which we are having, we will discuss and find solution together. If it is too difficult and can’t be solved by ourselves, we will ask our supervisor for advises.

**Unscheduled meeting:** If someone has an important problem want to be solved immediately, he proposes to Project Manager and we will have a meeting for discussion.

**Communication channel:** Our main communication channels are face – to – face meeting, Email, Facebook and Skype. However, we sometimes can make a phone call or instant message if someone has problem.

* 1. Communication with Supervisor

• Face-to-face meeting: Weekly on every Tuesday to make sure that supervisor can keep tracking of the team’s progress

• E-mail: Gmail is the fastest way to get advice and document checking from supervisor

• Mobile phone: is used to get time and place arranged for the meeting every weeks

1. **Configuration Management Process**5.1 CI Identification and Naming Convention

|  |  |  |
| --- | --- | --- |
| **No** | **Configuration Items** | **Naming convention** |
| **Project Management** | | |
| 1 | Project Plan | OL\_ProjectPlan\_v[version number]  For example: Olives\_ProjectPlan\_v1.0 |
| **Requirement & Design** | | |
| 2 | SRS | OL\_SRS\_v[version number]  For example: OL \_SRS\_v1.0 |
| 3 | Architectural Design | OL\_AD\_v[version number]  For example: OL\_ArchitecturalDesign\_v1.0 |
| 4 | Screen Design | OL\_SD\_ v[version number]  For example: OL\_ScreenDesign\_v1.0 |
| 5 | Data Design | OL\_DD \_v[version number]  For example: OL\_DataDesign\_v1.0 |
| **Source Code** | | |
| 6 | Source Code | OL\_SourceCode\_ v[version number][Tested/Untested]  For example: OL\_SourceCode\_v1.0Tested |
| **Support Document** | | |
| 7 | User Manual | OL\_UserManual\_v[version number]  For example: OL\_UserManual\_v[version number] |
| **Test** | | |
| 8 | Unit Test Plan | OL\_UnitTestPlan\_v[version number]  For example: OL\_UnitTestPlan\_v1.0 |
| 9 | Integration Test Plan | OL\_ITP\_v[version number]  For example: OL\_IntegrationTestPlan\_v1.0 |
| 10 | System Test Plan | OL\_STP\_v[version number]  For example: OL\_SystemTestPlan\_v1.0 |
| 11 | Unit Test Case | OL\_UTC\_v[version number]  For example: OL\_UnitTestCase\_v1.0 |
| 12 | Integration Test Case | OL\_ITC\_v[version number]  For example: OL\_IntegrationTestCase\_v1.0 |
| 13 | System Test Case | OL\_STC\_v[version number]  For example: OL\_SystemTestCase\_v1.0 |
| 14 | Test Result | OL\_TR\_v[version number]  For example: OL\_TestResult\_v1.0 |
| 15 | Test Data | OL\_TD\_v[version number]  For example: OL\_TestData\_v1.0 |
| **Process** | | |
| 16 | Guideline | OL\_[Name Of Guideline]Guideline\_v[version number] For example: OL\_UnitTestGuideline\_v1.0 |
| 17 | Convention | OL\_[Name Of Convention]Conventions\_v[version number]  For example: OL\_CodingConventions\_v1.0 |
| 18 | Template | OL\_Template-[Name Of Template]\_v[version number]  For example: OL\_Template-ChangeRequestForm\_v1.0 |
| 19 | Checklist | OL\_[Name Of Checklist]Checklist\_v[version number]  For example: OL\_Review Checklist\_v1.0 |
| **File Type** | | |
| 20 | MS Word | \*.doc |
| 21 | MS Excel | \*.xls |
| 22 | MS PowerPoint | \*.ppt |
| 23 | MS Project | \*.mpp |

*Table 9.1: CI Identification and Naming Convention*

1. **Project Milestones**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Milestones** | **Completion Date** | **Verification** |
| 1 | Project Start |  | Instructor approval |
| 2 | Submit report 1 |  | Instructor approval |
| 3 | Submit report 2 |  | Instructor approval |
| 4 | Submit report 3 |  | Instructor approval |
| 5 | Submit report 4 |  | Instructor approval |
| 6 | Submit report 5 |  | Instructor approval |
| 7 | Submit report 6 |  | Instructor approval |
| 8 | Submit report 3 update |  | Instructor approval |
| 9 | Submit report 4 update |  | Instructor approval |
| 10 | Submit report 5 update |  | Instructor approval |
| 11 | Submit Final Report |  | Instructor approval |

*Table 6.1: Project Milestones*

1. **Project Infrastructure**
   1. Software and Techniques

|  |  |  |
| --- | --- | --- |
| **Category** | **Software name** | **Version** |
| Operating System | Microsoft Windows 8,10 | Professional |
| Mac OS |  |
| Office Tools | Microsoft Office | 2010, 2013 |
| Task tracking | Microsoft Project | 2013 |
| Atlassian Jira Software | 7.1.7 |
| Design Tool | Astah Community | 7.0 |
| Developments tools | Xcode | 7.3 |
| Visual Studio 2015 Enterprise | 2015 |
| Resharper Ultimate | 9.0 |
| Database tool | SQLite | 3.13.0 |
| Source version control | Xcode | 7.3 |
| Documentation | Microsoft Word | 2010, 2013 |
| Microsoft PowerPoint | 2010, 2013 |
| Microsoft Excel | 2010, 2013 |
| UI design tool |  |  |
|  |  |

* 1. Hardware

Personal computer for developing and testing with the minimum configuration: 2GB Ram, 8 of hard disk, Intel Core I3.Internet network connection with the minimum speed 512kbit/s.

* 1. Other Infrastructure

Rooms for meeting and working. Internet and mobile phone services for communication.

1. **Coding convention**

Follow Objective C coding convention. See the reference for detail: [https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/CodingGuidelines/CodingGuidelines.html](https://l.facebook.com/l.php?u=https%3A%2F%2Fdeveloper.apple.com%2Flibrary%2Fmac%2Fdocumentation%2FCocoa%2FConceptual%2FCodingGuidelines%2FCodingGuidelines.html&h=KAQFIHl7V)

Follow C# coding convention. See the reference for detail

[https://msdn.microsoft.com/en-us/library/ff926074.aspx](https://l.facebook.com/l.php?u=https%3A%2F%2Fmsdn.microsoft.com%2Fen-us%2Flibrary%2Fff926074.aspx&h=KAQFIHl7V)

1. **Version Numbering Rule**

* **For Document:** Each file has a version number as part of its identity. This version number is physically represented as a 2-part string with the following format: <version>.<revision> For example, version 1.0 indicates 1 as the version, and 0 as the revision number. The original version will be numbered 0.1. Subsequent revisions will be numbered 0.2, 0.3 and so on. The approved version will be 1.0.
* **Version number**: appears to the left of the decimal. It is changed only when the core content of the item is significance changed. For example: when an item is completely overhauled, with substantial internal changes, the version 1.0 would become version 2.0.
* **Revision number**: appears to the right of the decimal. It is changed when the existing content is changed, but the main (or core) content is remained. The normal sequence of revision is 1.1, 1.2, and so on.

**For Software source files:** Software executable and support files are generally identified by name and version number. The version number is physically represented as a 3-part string with the following format: <Version>.<revision><update> For example, version 1.1a indicates 1 as the version, 0 as the revision number, and a as the update level.

* **Version number:** appears to the left of the decimal. It is changed only when the core content of the item is significance changed, as when moving from one are of the development tool to another, when an application is completely overhauled, or the user interface changes fundamentally. In this case, version 1.1a would become version 2.0.
* **Revision number:** appears to the right of the decimal. It is changed when new features, functionality or other content are added or significantly changed. In normal case, the core architecture or user interface have been extended or limited in some manner. The most common reason for changing the revision number is adding a new module or other functionality to the software. The normal sequence of revision is 1.0, 1.1 and 1.2 and so on.
* **Update level:** is appended or incremented when the only change to the software item is to correct one or more defects, without the addition of any new function. Version 1.1 would become v1.1a, 1.1b and so on. This updating is overridden when a combination revision, involving bug fixes and new feature additions, is performed. In such a case, the software revision number is incremented and any update indicator is dropped, as in v1.1b to 1.2.

1. **Directory Structure**

|  |  |  |
| --- | --- | --- |
| **Main folder** | **Sub-folder** | **Purpose** |
| Documents | Meeting minutes | Store project meeting minutes |
| Q&A | Store QA Management Sheet |
| Final deliverable | Report 1 | Store final deliverables of report 1 |
| Report 1 | Store final deliverables of report 1 |
| Report 2 | Store final deliverables of report 2 |
| Report 3 | Store final deliverables of report 3 |
| Report 4 | Store final deliverables of report 4 |
| Report 5 | Store final deliverables of report 5 |
| Report 6 | Store final deliverables of report 6 |
| Report 3 Update | Store final deliverables of report 3 Update |
| Report 4 Update | Store final deliverables of report 4 Update |
| Report 5 Update | Store final deliverables of report 5 Update |
| Report 6 Update | Store final deliverables of report 6 Update |
| Final Report | Store final deliverables of Final Report |
| Plan |  | Store project plan, Task list |
| Resource | Template | Store template needed in project |
| Tool | Store tool needed in project |
| Working Space | Each team members has a folder | Team member’s working area |
| Reference |  | Store reference needed in project |

*Table 10.1 Directory Structure*

1. **Other Olives Rules**

* **Email subject naming convention:** All email related to the Olives project must have prefix [Olives]. For example: [Olives] Progress Report 1.
* **Document changing rule:** When a member wants to modify a document, he/she must update version of that document with appropriate description for the modification.