# A

# **Software Project Management Plan**

## **Problem Definition**

* 1. **Name of this Capstone Project**
* Official name: **Discovery Year (DY) Emergency Response.**
* Vietnamese name: hệ thống thông báo khẩn cấp.
* Abbreviation: DYER.
  1. **Problem Abstract**

Nowadays, in many of University in the world they have a program for their student to go to outside their country for many kinds. Example: exchange students, internship and so on. And the number of students who go abroad keep increase by years. That’s why they need a tool, help them to contact with their students as fast as they can. Help them to warning their student in case of emergency, and quickly know their status, quickly find missing student and quickly find their information.

* 1. **Project Overview**
     1. **Current Situation**

Some problems encountered in this project.

Discovery Years Coordinators (DYC) behaviors: depend on the number of students going abroad in the moment, the work of managing students who go abroad right now is handled manually. Example: using whatsapp, messenger, gmail, … to contact with students. And they have to count the number of student manually, it took long time to report back to the manager.

* + 1. **The Proposed System**

The proposed system is an emergency response system. This system contains a website application for Admin and DYC from different faculty in the school. And the mobile applications for students and both Admin and DYC.

In details, the system provides following features:

**For students:**

* Only student who will go abroad can log in to this application.
* The account will be expired in the end of DY time.
* See basic information of host organization they will come.
* Give report in case emergency.
* Log in one time.
* Receive notification from their teacher.
* Give answer about their status.
* Student can see their tip-line.
* Update new phone number in new country.

**For DYC:**

* Log into both web application.
* Log into mobile application one time.
* List of their students by country.
* Only can see information of student inside their faculty.
* Create notification to group of student.
* See number of student answered and list of them.
* See number of student not answered and list of them.
* Receive notification from Admin.
* Import student information form Excel file.

**For Admin:**

* Log into both web application.
* Log into mobile application one time.
* View report.
* Create notification for student and DYC.
* List of all student by country.
* Manage account and information of DYC and Students: create, edit, activate or deactivate.
* Manage country and host organization: create, edit, activate or deactivate.
  + 1. **Boundaries of the System**

This section supposes that users of the system have stable Internet connection while using

**The system can:**

* Allow DYC know list of their faculty’s students and the country they go abroad.
* Allow Admin and DYC to send the notification to student’s device.
* Allow Admin to manage DYC’s staff and student abroad and host organization.
* Allow DYC quickly know about their student’s status in case emergency.
* Allow Admin to see the report from DYC.

**The system cannot:**

* Run without internet connection and database.
  + 1. **Boundaries of the System**

In the future, we want to upgrade and develop the system with following features:

* Support Malaysian languages.
* Admin can see the location of student on Google map.
* Admin and DYC can chat with their student.
* Can handle with more than 1000 students.
* Can know the last location of student in case no internet connection.
  1. **Development Environment**

#### **Hardware Requirements**

|  |  |  |
| --- | --- | --- |
| **Server** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| **Operating System** | Window Server 2008 | Window Server 2008 |
| **Computer Processor** | Intel® Xeon ® 1.4GHz | Intel® Xeon ® Quad Core (12M Cache, 2.50 GHz) |
| **Computer Memory** | 1GB RAM | 2GB or more |

Table 1. Hardware requirements for server

|  |  |  |
| --- | --- | --- |
| **PC** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| **Operating System** | Window 7 | Window 7 or more. |
| **Computer Processor** | Intel® Core i3 1.4GHz | Intel® Core i5 2.50GHz |
| **Computer Memory** | 1GB RAM | 2GB RAM or more |
| **Web Browser** | Firefox (v52 or higher), Chromes (v28 or higher) | Chrome latest stable version |

Table 2. Hardware requirements for PC

|  |  |  |
| --- | --- | --- |
| **Mobile Phone** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Wi-Fi (4 Mbps) | Wi-Fi (8 Mbps) |
| **Operating System** | Android, IOS | Android, IOS lates verison |
| **Phone Memory** | 1GB RAM | 2GB RAM or more |

Table 3. Hardware requirements for Mobile Phone

#### **Hardware Requirements**

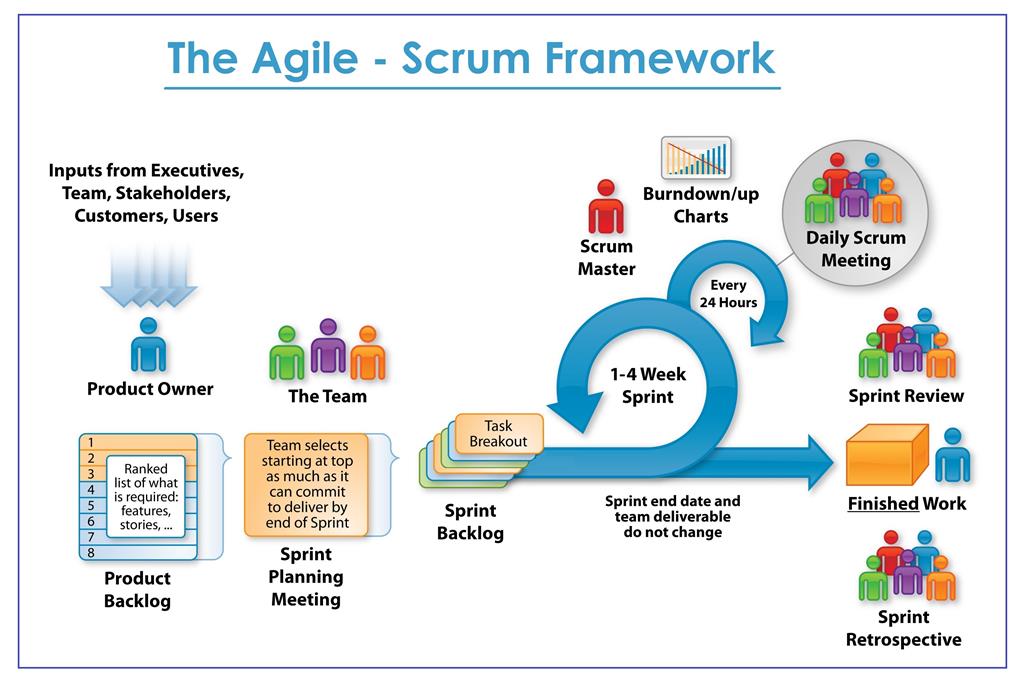
|  |  |  |
| --- | --- | --- |
| **Software** | **Name/ Version** | **Description** |
| **Operating System** | Windows 8 or above | Operating system and platform for development |
| **IDE** | Visual Studio 2017 | Used for implementing website and phone mobile |
| **Design Model Tool** | StarUML v2.5.1 | Used for creating model and diagram |
| **DBMS** | SQL Server 2014 | Used for creating & managing database |
| **Document storage** | Google Drive | Used for storing documents |
| **Store and manage source code** | Github with Source Tree | Used for storing source code |
| **Communication** | Facebook, Whatsapp, Email, | Used for exchanging information, online meeting, communication |
| **Web browser** | Chrome 28 or higher | Testing browser |
| **Mobile Phone** | Iphone SE, Sony Phone | Testing mobile application |

Table 4. Software requirements

1. **Project Organization**
2. **Software Process Model**

In this project, our team choose Scrum, an Agile framework that allows us step by step to set up a plan successfully. We choose this model because of the following reasons:

* Time-saving: daily meetings ensure that the process is at correct stage, as established at the beginning of the project.
* Easy to use: suitable model for small and medium project.
* Fast response to changes: product owner may change requirements or extend / reduce scope and we can adapt better right on time.
* Encourage teamwork: roles and tasks are divided and assigned efficiently.
* Functional tests are frequent in the entire process.



*Figure 1. Scrum Framework*

1. **Roles and Responsibilities.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Full name** | **Role in Group** | **Responsibilities** |
| 1 | Ngô Đăng Hà An | Supervisor,  Product owner | * Specify user requirements * Control the development process * Support in technique and business logic. |
| 2. | Ngô Thục Thiên Bình | Team Leader,  BA,  Developer,  Tester | * Manage process * Plan Scrum implementations * Clarify requirements * Design database * Code on Web Application * Write document and report * Test * Contact with client * Research technology * Design GUI Website * Provide API for Mobile |
| 3. | Mohammad Firdaus bin Haji Jais | Team member,  Developer,  Tester | * Clarify requirements * Design database * Code Web Application * Write document and report * Test * Contact with client * Research technology * Design GUI Website * Provide API for Mobile |
| 4. | Phạm Thị Xuân Hạ | Team member,  Developer,  Tester | * Design GUI Mobile * Clarify requirements * Design database * Code on Mobile Application * Test * Create test plan * Write document and report * Research technology |
| 5. | Muhd. Wafi Nur Arif bin Shamdi | Team member,  Developer,  Tester | * Design GUI Mobile * Clarify requirements * Design database * Code on Mobile Application * Test * Write document and report * Research technology |
| 6. | Hoàng Nhựt Vũ | Team member,  Developer,  Tester | * Design GUI Mobile * Clarify requirements * Design database * Code on Mobile Application * Test * Write document and report * Research technology |

*Table 5. Roles and responsibilities details*

1. **Tools and Techniques.**

|  |  |
| --- | --- |
| **Tool/Technique** | **Name/Version** |
| Front-end | HTML5, CSS3, Bootstrap, Javascript/jQuery, AdminLTE |
| Back-end | C#, .NET Framework |
| Managing Database | Microsoft® SQL Server® 2014 Management Studio |
| Source Control | Sourcetree (server github.com) |
| Modeling Tool | StarUML v2.8.1 |
| Web Browser | Google Chrome |
| Mobile Application | Xamarin Cross-Platform/ lates version |
| Mobile Phone | Iphone, Android phone |

Table 6. Tools and techniques

## **Project Management Plan**

1. **Product Backlog**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story**  **ID** | **Features** | **Task**  **ID** | **Task description** | **Sprint** |
| 1 | Create Product Backlog | 1.1 | Create product backlog | 1 |
| 2 | Create Introduction document | 2.1 | Create introduction document | 1 |
| 2.2 | Review introduction document | 1 |
| 3 | Learning Xamarin | 3.1 | Learning Xamarin Cross-Platform | 1 |
| Learning MVC API, Entity Framework | 3.2 | Learning MVC API, Entity Framework | 1 |
| Learning Firebase Push Notification | 3.3 | Learning Firebase Technology | 1 |
| 4 | Create UI mockups | 4.1 | Search for suitable UI | 2 |
| 4.2 | Login Mobile UI | 2 |
| 4.3 | Admin Mobile UI | 2 |
| 4.4 | DYC Mobile UI | 2 |
| 4.5 | Student Mobile UI | 2 |
| 4.6 | Login Website UI | 2 |
| 4.7 | Admin Website UI | 2 |
| 4.8 | DYC Website UI | 2 |
| 5 | Create Software Project Management Plan | 5.1 | Problem definition | 2 |
| 5.2 | Project organization | 2 |
| 5.3 | Project management plan | 2 |
| 5.4 | Coding convention | 2 |
| 6 | Create Software Requirement Specifications | 6.1 | User requirement specification | 2 |
| 6.2 | External interface requirement | 2 |
| 6.3 | Use case diagram | 3 |
| 6.4 | Software system attributes | 2 |
| 6.5 | Database diagram | 2 |
| 7 | Create Software Design Description | 7.1 | Design overview | 3 |
| 7.2 | System architectural design | 3 |
| 7.3 | Component diagram | 3 |
| 7.4 | Detailed description of components | 3 |
| 7.5 | Sequence diagram | 3 |
| 7.6 | User interface design | 2 |
| 7.7 | Database design | 3 |
| 7.8 | Coding | 3 to 6 |
| 8 | Create Software Test Documentation | 8.1 | Test Plan | 5 |
| 8.2 | Test Cases | 5 |
| 8.3 | Checklists | 6 |
| 9 | Software User’s Manual | 9.1 | Installation Guide | 6 |
| 9.2 | User’s Guide | 6 |

Table 7. Product backlog

1. **Deliverables**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Deliverable** | **Delivery date** | **Delivery location** | **Note** |
| 1 | Introduction Document, Task list | 13/10/2018 | Supervisor’s office | Report No.1 |
| 2 | Software Project Management Plan | 13/10/2018 | Supervisor’s office | Report No.2 |
| 3 | Software Requirements Specification | 20/10/2018 | Supervisor’s office | Report No.3 |
| 4 | Software Design Description | 10/11/2018 | Supervisor’s office | Report No.4 |
| 5 | Software Test Documentation Guide Implementation (Coding) | 20/11/2018 | Supervisor’s office | Report No.5 |
| 6 | Software User’s Manual | 27/11/2018 | Supervisor’s office | Report No.6 |

Table 8. Deliverables

\* For each sprint, deliverables are potentially shippable products, which can be a part of documents or runnable software application which is implementation of some program features.

\* Each sprint is completed in 1 week.

## **Coding Convention**

* **Naming conventions:**
  + Use pascal case for class names and method names.
  + Use camel case for method arguments and local variables.
  + Do not use underscore in identifiers. Except: prefix private static variables with an underscore.
  + Use noun or noun phrases to name a class.
  + Prefix interfaces with letter ‘I’.
* **Layout conventions:**
  + Tabs must be set exactly 4 spaces.
  + Avoid lines longer than 80 characters.
  + Vertically align curly brackets.
  + Write only one statement per line.
  + Write only one declaration per line.
  + Add one blank line between method definitions and property definitions.
* **Declaration:**
  + Use implicit type ‘var’ for local variable declarations. Except: primitive types (int, string, double, etc.) use predefined names.
  + Organize namespaces with a clearly defined structure.
* **Commenting conventions:**
  + Place comment on a separate line.
  + Begin comment text with an uppercase letter and end with a period.
  + Add one space between comment delimiter (//) and comment text.