

**CAPSTONE PROJECT REPORT**

**Report 2 – Project Management Plan**

– Hanoi, August 2024 –

**Table of Contents**

[I. Record of Changes 3](#_heading=h.gjdgxs)

[II. Project Management Plan 4](#_heading=h.30j0zll)

[1. Overview 4](#_heading=h.1fob9te)

[1.1 Scope & Estimation 4](#_heading=h.3znysh7)

[1.2 Project Objectives 5](#_heading=h.2et92p0)

[1.3 Project Risks 5](#_heading=h.tyjcwt)

[2. Management Approach 6](#_heading=h.3dy6vkm)

[2.1 Quality Management 6](#_heading=h.1t3h5sf)

[2.2 Project Training Plan 7](#_heading=h.4d34og8)

[3. Project Deliverables 8](#_heading=h.2s8eyo1)

[4. Responsibility Assignments 8](#_heading=h.17dp8vu)

[5. Project Communications 9](#_heading=h.3rdcrjn)

[6. Configuration Management 9](#_heading=h.26in1rg)

[6.1 Document Management 9](#_heading=h.lnxbz9)

[6.2 Source Code Management 10](#_heading=h.35nkun2)

[6.3 Tools & Infrastructures 11](#_heading=h.1ksv4uv)

# I. Record of Changes

| **Date** | **A\* M, D** | **In charge** | **Change Description** |
| --- | --- | --- | --- |
| 25/5/2024 | A | Yen | Project Deliverables,Responsibility Assignments, Scope & Estimation |
| 25/5/2024 | A | Kỳ Anh | Project Objectives |
| 25/5/2024 | A | Hiếu | Project Risks |
| 25/5/2024 | A | Huy | Quality Management, Source Code Management |
| 25/5/2024 | A | Dũng | Project Training Plan,Tools & Infrastructures |
| 6/6/2024 | M | Huy,Hieu | 1.1 Scope & Estimation |
| 7/6/2024 | M | Yen | Responsibility Assignments |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

\*A - Added M - Modified D - Deleted

# II. Project Management Plan

## 1. Overview

### 1.1 Scope & Estimation

| # | WBS Item | Complexity | Est. Effort  (man-days) |
| --- | --- | --- | --- |
|  | **Guest** |  |  |
| **1** | View Home Page | **Simple** | **5** |
| 2 | Make Online Reservation | Medium | 12 |
| 3 | View Columbarium Building | Simple | 4 |
| 4 | View Floors within Building | Simple | 4 |
| 5 | View Sections within Floor | Simple | 4 |
| 6 | View Niches within Section | Simple | 5 |
| **7** | Schedule Contract Signing | Medium | **10** |
|  | **Customer** |  |  |
| 8 | Login | Simple | 3 |
| 9 | Reset Password | Simple | 3 |
| 10 | Register for Visit | Simple | 5 |
| 11 | View List of Services | Simple | 4 |
| **12** | View Service Details | **Simple** | **5** |
| 13 | Register for Service Booking | Medium | 10 |
| 14 | Pay Service Invoice | Medium | 8 |
| 15 | View Purchased Niches | Medium | 8 |
| 16 | View Service Orders | Medium | 8 |
| **17** | View Visit Orders | **Medium** | **8** |
| 19 | Edit Service Order | Medium | 7 |
| 20 | Edit Visit Order | Medium | 7 |
| 21 | Delete Service Order | Medium | 7 |
| 22 | Delete Visit Order | Medium | 7 |
| 23 | View Contract | Medium | 8 |
| 24 | Register for Contract Renewal | Medium | 8 |
| 25 | View Personal Information | Simple | 4 |
| 26 | Edit Personal Information | Medium | 8 |
|  | **Staff** |  |  |
| 27 | View Columbarium Building | **Simple** | **4** |
| 28 | View Floors within Building | **Simple** | **4** |
| 29 | View Sections within Floor | **Simple** | **4** |
| 30 | View Niches within Section | **Simple** | **5** |
| 31 | View Niche Details | **Medium** | **7** |
| 32 | View List of Services | **Simple** | **4** |
| 33 | View Service Details | **Simple** | **5** |
| 34 | Add New Service | **Medium** | **10** |
| 35 | Edit Service Information | **Medium** | **8** |
| 36 | Delete Service | **Medium** | **8** |
| 37 | View List of Service Orders | **Medium** | **8** |
| 38 | View Service Order Details | **Medium** | **8** |
| 39 | Edit Service Order | **Medium** | **7** |
| 40 | View List of Visit Registrations | **Medium** | **8** |
| 41 | View Visit Registration Details | **Medium** | **8** |
| 42 | Edit Visit Registration Information | **Medium** | **8** |
| 43 | View List of Reservation Orders | **Medium** | **8** |
| 44 | View Reservation Order Details | **Medium** | **8** |
| 45 | Update Reservation Order Information | **Medium** | **8** |
| 46 | View List of Customers | **Medium** | **8** |
| 47 | View Customer Details | **Medium** | **7** |
| 48 | Edit Customer Information | **Medium** | **10** |
| 49 | Add New Customer | **Medium** | **8** |
| 50 | View List of Contracts | **Medium** | **8** |
| 51 | View Contract Details | **Medium** | **10** |
| 52 | Add New Contract | **Medium** | **8** |
| 53 | Remind Customers for Payment | **Medium** | **8** |
| 54 | View List of Contract Renewal Requests | **Medium** | **8** |
| 55 | View Contract Renewal Request Details | **Medium** | **8** |
| 56 | Edit Contract Renewal Request | **Medium** | **7** |
|  | **Manager** |  |  |
| 57 | View Contract Status Reports | **Medium** | **8** |
| 58 | View Service Status Reports | **Medium** | **8** |
| 59 | View Income Reports | **Medium** | **8** |
| 60 | Add New Staff | **Medium** | **8** |

### 1.2 Project Objectives

### 

| **#** | Testing Stage | Test Coverage | No. of Defects | % of Defect | Note |
| --- | --- | --- | --- | --- | --- |
| 1 | Reviewing | 100% | 20 | 45% |  |
| 2 | Unit Test | 100% | 9 | 20.5% |  |
| 3 | Integration Test | 100% | 9 | 20.5% |  |
| 4 | System Test | 100% | 3 | 7% |  |
| 5 | Acceptance Test | 100% | 3 | 7% |  |

### 

### 1.3 Project Risks

| **#** | **Risk Description** | **Impact** | **Possibility** | **Response Plans** |
| --- | --- | --- | --- | --- |
| 1 | Shortage of human resources during project development | High | Medium | - Cross-training so that members can support each other when members are lacking  - Perform regular security testing.  - Apply strong security measures such as data encryption and multi-factor authentication (MFA). |
| 2 | The system encountered a security error and was attacked | High | Low | - Cross-training so that members can support each other when members are lacking  - Perform regular security testing.  - Apply strong security measures such as data encryption and multi-factor authentication (MFA). |
| 3 | Request changes from customers during development | Medium | High | - Establish a clear change management process.  - Evaluate the impact of changes on project time and costs before implementation. |
| 4 | System performance is not satisfactory | High | Medium | - Optimize source code and database structure.  - Perform periodic performance and load testing to detect and fix problems promptly. |
| 5 | Delay in completing milestones due to dependence on third parties | Medium | Medium | - Make contingency plans for tasks that depend on third parties.  - Regularly monitor progress and update plans when necessary. |
| 6 | Difficulty in integrating backend and frontend | Medium | Medium | - Establish clear and consistent API communications.  - Organize regular meetings between teams to resolve integration issues promptly. |
| 7 | Risk of inaccurate or lost data during data migration | High | Low | - Perform data testing before migration.  - Develop a data backup and recovery plan. |
| 8 | Customer satisfaction level is not achieved after system implementation | Medium | Medium | - Collect feedback from customers regularly during the development process.  - Adjust and improve based on customer feedback. |

## 2. Management Approach

### 2.1 Quality Management

***Approach to Improve Project Quality:***

***Defect Prevention*** :

***Training:*** Organize training courses for the development team on secure programming methods and techniques.

***Standards & Guidelines:*** Apply coding standards and detailed guidelines for coding, testing, and documentation.

***Reviewing*** :

***Code Review:*** Organize periodic source code reviews to detect and fix errors early.

***Peer Review:*** Peer review to ensure source code quality and detect potential issues.

***Unit Testing :***

***Automated Unit Tests:*** Write automated unit tests for each smallest function of the system.

***Continuous Integration :*** Use a CI (Continuous Integration) tool to automatically run unit tests every time a code change occurs.

***Integration Testing :***

***Integration Tests:*** Integration tests to ensure modules operate consistently and without conflicts.

***Test Environments:*** Set up separate test environments to test integration before deploying to the real environment.

***System Testing :***

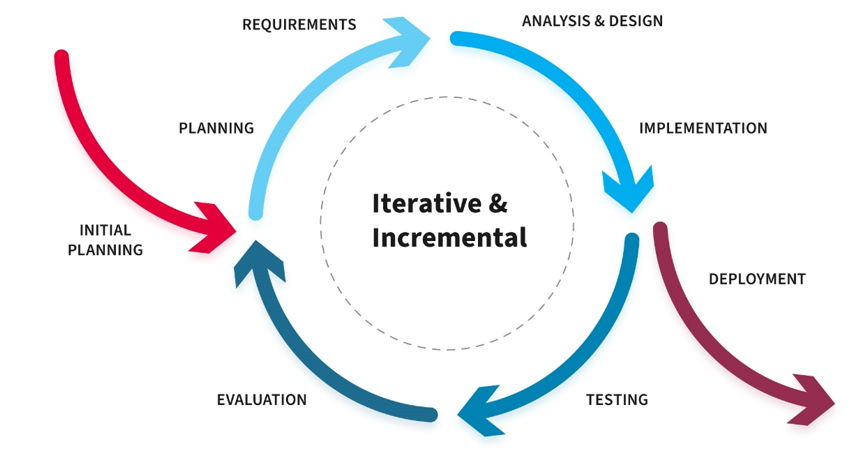
***Functional Testing:*** Ensuring all system functions work as required.

***Performance Testing:*** Performance testing to ensure the system operates stably under heavy load.

***Security Testing:*** Security testing to ensure the system has no security vulnerabilities.

## 2. Management Approach

### 2.1 Project Process

**

*Figure 2-X: Iterative and Incremental Software Process Model*

Based on recommendation of supervisor and actual situation of the project, our Project team chooses to use the Iterative and Incremental Software Process Model. In an Iterative and Incremental model, initially, a partial implementation of a total system is constructed so that it will be in a deliverable state. The Iterative and Incremental 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, which is the case of this project. By using this software process model, we break down the developing system task into a series of smaller tasks which will be completed separately, allowing us to take advantage of what was learned during the development of earlier parts of the system.

The reasons for the project to choose this model are:

· You can develop some main functions that prioritise requirements first.

· Requirements changes can be easily accommodated.

· Testing and debugging during smaller iterations is easy.

· Client can give input to every item increment, accordingly maintaining a strategic distance from differences toward finish of improvement.

### 2.2 Quality Management

#### 2.2.1 Code Review

· Any contribution of team members will be reviewed by team leader before merging into main branch.

· If team leader finds any defect or ambiguity in code or document, he will discuss directly with the author the clarify problems and give instruction on how to fix them.

#### 2.2.2 Unit Testing

· As each team member will be not only a developer but also a tester, each person will be testing for what the part they code.

· Any defects found will be recorded on GitHub Issues as a bug tracking software, with details and images for evidence.

· That team member will then be responsible for repairing those defects.

#### 2.2.3 Integration Testing

· All Integration test cases will be recorded into a spread sheet, divided by feature. The team member who developed the feature will perform Integration test for that feature.

· Any defects found will also be recorded on GitHub Issues and will be fixed by that team member.

#### 2.2.4 System Testing

· In the team meeting at the end of each Iteration, all team will discuss to write some System test cases that can cover all features.

· Any defects found will also be recorded on GitHub Issues and will be fixed by the team member who is responsible for the feature to which it belongs.

#### 2.2.5 Acceptance Testing

· At the end of each Iteration, system will be deployed on the Internet and team members will contact directly with actual landlords and students to use.

· Any feedbacks and comments from the client will be recorded and bring to discuss with whole team and supervisor to decide what should be done to improve the system.

### 

### 

### 

### 

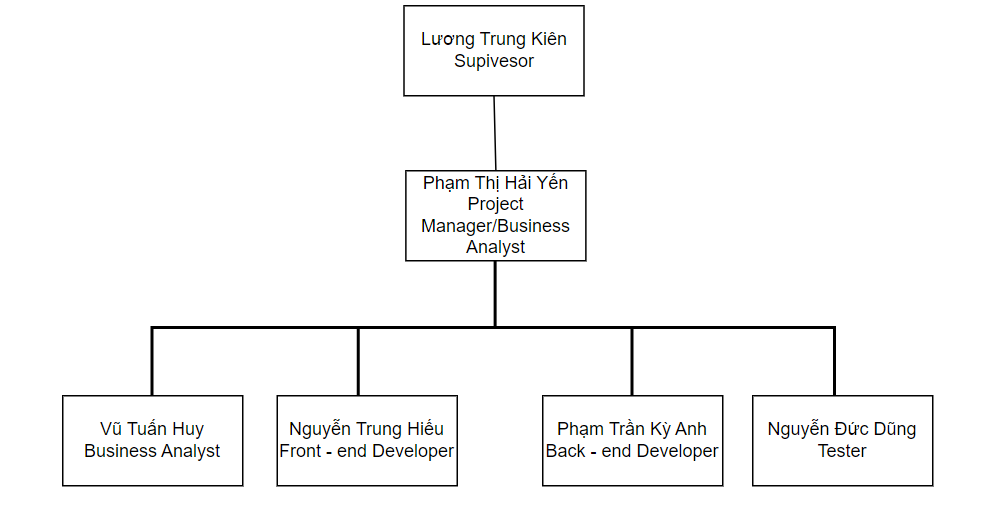
### 2.2 Project Training Plan

| **Training Area** | **Participants** | **When, Duration** | **Waiver Criteria** |
| --- | --- | --- | --- |
| ASP.NET Core | Backend Developers | Week 1-2, 5 sessions (2 hours each) | Mandatory |
| Entity Framework | Backend Developers | Week 2-3, 4 sessions (2 hours each) | Mandatory |
| SQL Server | Backend Developers | Week 3-4, 3 sessions (2 hours each) | Mandatory |
| NextJS | Frontend Developers | Week 1-2, 5 sessions (2 hours each) | Mandatory |
| RESTful API Design | All Developers | Week 2-3, 3 sessions (2 hours each) | Mandatory |
| Git, GitHub | All Developers | Week 1, 2 sessions (2 hours each) | Mandatory |
| Testing and QA | QA Team | Week 3, 3 sessions (2 hours each) | Mandatory |

## 3. Project Deliverables

| **#** | **Deliverable** | **Due Date** | **Notes** |
| --- | --- | --- | --- |
| 1 | Product Vision  Basic Use case  System actors  Report 1 |  | Initiate Project |
| 2 | Project Scope  Requirements  Business Rules |  | Customer Meeting  Functional Requirements  Non-functional Requirements |
| 3 | Plan and Schedule  Report 2 |  | Project Schedule  Project Management Plan |
| 4 | System Design  Report 3  Report 4 |  | Screen Design  Architectural Design & Detail Design  Database Design |
| 5 | Code & Implement  Iteration 1 |  | Code  Unit Test  Integration Test |
| 6 | Code & Implement  Iteration 2 |  | Code  Unit Test  Integration Test  System Test  User Acceptance Test |
| 10 | Final Report  Final Product  User Guides  Presentation |  | Final Report  User Guides  Presentation Slides |

## 4. Responsibility Assignments

**

*D~Do; R~Review; S~Support; I~Informed; <blank>- Omitted*

| **Responsibility** | **HieuNTHE130901** | **HuyVThe130028** | **DungNDhe130758** | **Anhptkhe130580** | **yenpthhe141292** |
| --- | --- | --- | --- | --- | --- |
| Project Planning & Tracking | S | R | R | S | D |
| Prepare Project Introduction Document | S | R | R | S | D |
| Prepare SRS Document (Overview Part) | S | D | R | D | S |
| Prepare SRS Document (User Requirements) | S | D | S | D | R |
| Software Design Description (SDD) | D | R | R | S | D |
| Software Test Documentation (STD) | R | D | S | R | D |
| Software User Guides (SUG) | R | D | S | D | R |
| Software Product (SP) | D | R | S | S | D |

## 5. Project Communications

| **Communication Item** | **Who/ Target** | **Purpose** | **When, Frequency** | **Type, Tool, Method(s)** |
| --- | --- | --- | --- | --- |
| Daily Meeting | All team members | - Checking progress of assigned tasks  - Finding solutions for difficult problems | 9 p.m everyday | Discord |
| Assign Tasks | All team members | - Project Manager assigns tasks to other team members | Everyday | Trello |
| Weekly Meeting With Supervisor | All team members, supervisor | - Checking progress of project  - Plan upcoming tasks  - Update requirements | Once a week | Online, Offline |
| Meeting with Supervisor and University Staffs | All team members, supervisor, Staffs | - Clarify user requirements, roles & project scope | Once a week | Offline |

## 6. Configuration Management

### 6.1 Document Management

How to Manage Project Documents and Their Changes/Versions:

Centralized Storage:

Tools: Use Google Sheets to store and manage all project documents. Google Sheets allows for easy sharing, collaboration, and tracking of changes.

Access Control: Implement role-based access control to ensure only authorized people can change important documents.

Version:

Version Number: Assign a version number to each document and update the version number with each change (for example, v1.0, v1.1, v2.0).

Change Log: Maintain a change log for each document, detailing changes made, author of the changes, and date of change.

Document Review Process:

Peer Review: Implement a peer review process where documents are reviewed by one or more other team members to ensure quality and accuracy.

Evaluation Notes: Record comments and suggestions from evaluations and make necessary changes.

Backup and Restore:

Regular Backups: Set up regular backups for all Google Sheets documents.

Recovery: Ensure document recovery plans are in place in the event of data loss or failure.

### 6.2 Source Code Management

How to Manage Project Source Code and Their Changes/Versions:

Source Code Repository:

Tools: Use GitHub to store and manage project source code. GitHub offers powerful features for version management, code merging, and issue tracking.

Version Management Process:

Main Branch: Use the main branch to store stable and thoroughly tested source code.

Development Branch: Create development branches for each new feature or bug fix.

Pull Requests: Use pull requests to propose changes and request code reviews before merging into the master branch.

Source Code Testing and Review:

Source Code Review: Perform source code reviews through pull requests to ensure high quality source code and minimize bugs.

Automated Testing: Set up an automated testing process using CI/CD (Continuous Integration/Continuous Deployment) tools to check the source code every time there is a change.

Issue Tracking and Task Management:

Issues: Use the issues system on GitHub to track bugs, feature requests, and development tasks.

Milestones: Create milestones to manage project progress and ensure goals are completed on time.

Backup and Restore:

Regular Backups: Ensure regular backups of your source code repository on GitHub.

Recovery: Establish a source code recovery process in the event of a crash or data loss.

### 6.3 Tools & Infrastructures

| **Category** | **Tools / Infrastructure** |
| --- | --- |
| **Technology** | NextJS14 (FrontEnd), ASP.NET 8 (BackEnd) |
| **Database** | SQL Server |
| **IDEs/Editors** | Visual Studio, Visual Studio Code,MSSQL |
| **Diagramming** | StarUML, DrawIO,Figma |
| **Documentation** | Microsoft Office, Google Docs/Sheets/Slides |
| **Version Control** | GitHub (Source Codes), Google Drive (Documents) |
| **Deployment Server** | Amazon Web Services (AWS) |
| **Project Management** | Trello (Schedule), GitHub (Tasks, Issues) |