

International School

**Capstone Project 1**

*CMU-SE 450*

**PROJECT PLAN DOCUMENT**

*Version: 1.0*

*Date: 31/08/2021*

**School Connect Application**

**Submitted by:**

Nguyen Thanh Phu

Nguyen Trung Hieu

Dang Nguyen Bao Hoai

Nguyen Van Thang

**Approved by**

Nguyen Minh Nhat

**Proposal Review Panel Representative:**

Name Signature Date

**Capstone Project 1- Mentor:**

Name Signature Date

# Project Information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project acronym** | SConA | | | |
| **Project Title** | School Connect Application | | | |
| **Start Date** | 22-Aug-2021 | **End Date** | | 18-Dec-2021 |
| **Lead Institution** | International School, Duy Tan University | | | |
| **Project Mentor & contact details** | Name: Nguyen Minh Nhat  Email: nhatnm2010@gmail.com  Tel: +84 905 125 143 | | | |
| **Scrum Master & contact details** | Name: Nguyen Thanh Phu  Email: [thangphu104@gmai.com](mailto:thangphu104@gmai.com)  Tel: +84 772 492 301 | | | |
| **Team members** | **Name** | **Email** | **Tel** | |
| Nguyen Trung Hieu | hnguyentrung20@gmail.com | +84 975 299 149 | |
| Dang Nguyen Bao Hoai | dangbhoai@gmail.com | +84 773 305 395 | |
|  | Nguyen Van Thang | thangnguyendien@gmail.com | +84 906 421 132 | |

|  |  |  |  |
| --- | --- | --- | --- |
| **DOCUMENT INFORMATION** | | | |
| **Document Title** | Project plan | | |
| **Author(s)** | Team C1SE.44 | | |
| **Role** | C1SE.44-Project Plan-SConAdocx-v1.0 | | |
| **Date** | 31-Aug-2021 | **Filename** | C1SE.44-Project Plan-SConAdocx-v1.0 |
| **URL** | **https://github.com/PhuNguyenThang/Capstone1** | | |
| **Access** | Project and CMU Program | | |

**DOCUMENT APPROVALS**

The following signatures are required for approval of this document.

|  |  |  |
| --- | --- | --- |
| Nguyen Minh Nhat  *Mentor* | Signature | Date |
| Nguyen Thanh Phu  *Scrum Master* | Signature | Date |
| Nguyen Trung Hieu  *Product owner* | Signature | Date |
| Dang Nguyen Bao Hoai  *Team Member* | Signature | Date |
| Nguyen Van Thang  *Team Member* | Signature | Date |

**RECORD OF CHANGE**

\*A - Added M - Modified D – Deleted

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Effective Date** | **Changed Item** | **A\* M, D** | **Change Description** | **Reason for Change** | **Revision Number** |
| 31-Aug | Create Document | A |  |  | 1.0 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Table of Contents

[Project Information 2](#_Toc81876440)

[*1.* Project Overview 7](#_Toc81876441)

[**1.1. Project Description 7**](#_Toc81876442)

[**1.2. Scope and Purpose 7**](#_Toc81876444)

[1.2.1 Scope 7](#_Toc81876445)

[1.2.2*.* Purpose 7](#_Toc81876446)

[**1.3. Assumptions and Constraints 8**](#_Toc81876447)

[1.4. Project Objectives 9](#_Toc81876449)

[1.4.1. Standard Objectives 9](#_Toc81876450)

[1.4.2. Specific Objectives 9](#_Toc81876452)

[1.4.3. Quality Attributes 9](#_Toc81876453)

[1.4.4. Business goals 11](#_Toc81876456)

[***1.5.* Dependencies 11**](#_Toc81876458)

[**1.6. Project Risk 11**](#_Toc81876460)

[*2.* Project Development Approach 12](#_Toc81876462)

[**2.1. Project Process 12**](#_Toc81876463)

[2.1.1. Reasons for selecting 12](#_Toc81876464)

[2.1.2. Agile Methodology 12](#_Toc81876465)

[2.1.3. Scrum Process 14](#_Toc81876466)

[**2.2. Requirement Change Management 15**](#_Toc81876467)

[**2.3. Product Integration Strategy 16**](#_Toc81876468)

[**2.4. Quality Management 16**](#_Toc81876469)

[2.4.1. Estimates of Defects to be detected 16](#_Toc81876470)

[2.4.2. Strategy for Meeting Quality Objectives 18](#_Toc81876472)

[2.4.3. Quality Control 18](#_Toc81876474)

[2.4.4. Measurements Program 19](#_Toc81876476)

[**2.5. Unit Testing Strategy 20**](#_Toc81876478)

[**2.6. Integration Testing Strategy 20**](#_Toc81876479)

[**2.7. System Testing Strategy 20**](#_Toc81876480)

[*3.* Estimation 21](#_Toc81876481)

[**3.1. Size 21**](#_Toc81876482)

[***3.2.* Effort 21**](#_Toc81876484)

[**3.3. Schedule 22**](#_Toc81876486)

[3.3.1. Project Milestone & Deliverables 22](#_Toc81876487)

[3.3.2. Work Breakdown Structure 23](#_Toc81876489)

[**3.4. Resource 27**](#_Toc81876491)

[3.4.1. Human Resources 27](#_Toc81876492)

[3.4.2. Equipment 27](#_Toc81876494)

[**3.5. Infrastructure 27**](#_Toc81876496)

[**3.6. Training Plan 28**](#_Toc81876498)

[**3.7. Finance 29**](#_Toc81876501)

[*4.* Project Organization 29](#_Toc81876503)

[**4.1. Organization Structure 29**](#_Toc81876504)

[**4.2. Project Team 30**](#_Toc81876506)

[*5.* Communication & Reporting 31](#_Toc81876508)

[***5.1.* Reporting Methodology 31**](#_Toc81876509)

[***5.2.* Communication Methodology 31**](#_Toc81876511)

[*6.* Configuration Management 31](#_Toc81876513)

[*7.* Security Aspects 32](#_Toc81876514)

[*8.* References 33](#_Toc81876515)

# Project Overview

## Project Description

## Table 1: Project Description

|  |  |  |  |
| --- | --- | --- | --- |
| **Project code** | C2SE.06 | **Contract type** | External Project |
| **Customer** | Nguyen Hien high school | **/End-User** | Student, Teacher |
| **Project Type** | External | **Project Manager/ Scrum master** | Nguyen Thanh Phu |
| **Project Category** | Development & Maintenance | **Business domain** | social network |
| **Application type** | Web Application |  |  |

## Scope and Purpose

* + 1. **Scope**

The backbone of the project:

* Manage all account in the system
* Create, modify, delete the post
* Comment in the post
* Sent message in the system
* Censorship the post
* Notification the new message or new post
* Delete comment or block box chat have message toxic
* Report the message or comment toxic
* See calendar event
* Manage member in forum

Language:

* English
  + 1. **Purpose**

School Connect Application is mean website application help connection between school and student in the context epidemic COVIC-19. Students do not need come to school to hear announcements about upcoming events or new policies, school can use this application for notify to students about policies of school. Teacher can keep contract with student without using another app and don’t worry about personal information network social network exposed, student can create post in the forum their join and see calendar of event will take

With intuitive visual interface will help the teacher and student easy to using system

This only for student and teacher in that school, so school can manage all account using in the system and censor is teacher can manage all post and member in the forum them manage

## Assumptions and Constraints

## Table 2: Assumptions and Constraints

|  |  |  |
| --- | --- | --- |
| **No** | **Description** | **Note** |
| **Assumptions** | | |
| 1 | Auto import data account and download report may not be able to finish on  time | Scope |
| 2 | Customer reviewers will get seven days to approve a milestone document. If no comments are received within this time period, it will be considered as approved. | External Interfaces |
| 3 | The system will be tested with pre-prepared standard data sources | Data Sources |
| **Constraints** | | |
| 1 | SConA must be done and delivered by December 15th according to the customer demonstration deadline | Schedule |
| 2 | The project must comply with information security rules | Security |
| 3 | The project must conform to the system architecture specification and end-user requirements | Consistency |
| Constraint | Constraints Description | Guidelines for Acceptance |
| Ethical | This is system for people of school so outsider can’t go into system destructive forum  Information of student or teacher can’t expose. | Ethical considerations can be broad. Areas that are typically addressed include intellectual property, reverse- engineering, privacy, security, and the conflict between cost and safety |
| Public health, safety, and welfare | In the context of the epidemic student and teacher don’t need meet face to face so limit the spread of disease | Includes safety standards as well as impact of the design on users (for example, electrical or physical hazards) |
| Cultural | This application can be change communication student can contact with school or teacher event they don’t at school or using many third app to hold contact | Which cultural characteristics could influence the approach?  How do the design from different cultures differ? |

1. **Project Objectives**

### Standard Objectives

## Table 3: Standard Objectives

|  |  |  |  |
| --- | --- | --- | --- |
| **Metrics** | **Unit** | **Committed** | **Note** |
| Start Date | dd-mmm-yy | 18-Aug-2021 |  |
| End Date | dd-mmm-yy | 15-Dec-2021 |  |
| Duration | days | 121 |  |
| Team Size | Person | 4 |  |
| Billable Effort / | Person-day | 4/day |  |
| Number of work hours per day for one engineer | Person-hour | 2 | Can overtime if project be progress late |

1. **Specific Objectives**

* Build website where student and teacher can sent message with together
* Build website where student, teacher, censor, admin can create post
* Build website where censor can manage post and member in forum
* Admin can manage all account in the system
* Student, teacher, censor see the post in the forum they join or manage
* Build a website with user-friendly interface.

1. **Quality Attributes**

## Table 4.1: Quality Attributes: Performance

|  |  |
| --- | --- |
| ID | QA01 |
| Quality Attributes | Performance |
| Stimulus | Sent message in the system |
| Source(s) of stimulus | Student or teacher, censor |
| Artifacts | System |
| Environment | Normal operating system |
| System response | The person receiving the message will receive the message |
| Response measure(s) | In 10 seconds |

## Table 4.2: Quality Attributes: Performance

|  |  |
| --- | --- |
| ID | QA02 |
| Quality Attributes | Performance |
| Stimulus | Create post in the forum they join or manage |
| Source(s) of stimulus | Student or teacher, censor |
| Artifacts | System |
| Environment | Normal operating system |
| System response | Another user receives notify from new post |
| Response measure(s) | In less 15 second |

## Table 4.3: Quality Attributes: Security

|  |  |
| --- | --- |
| ID | QA03 |
| Quality Attributes | Security |
| Stimulus | Using website application |
| Source(s) of stimulus | Student or teacher, censor |
| Artifacts | Information in system |
| Environment | Normal operating system |
| System response | Require user to login |
| Response measure(s) | Prevent access to the website without login |

## Table 4.4: Quality Attributes: Security

|  |  |
| --- | --- |
| ID | QA04 |
| Quality Attributes | Security |
| Stimulus | Using function delete posts that aren’t theirs, remove members from forum |
| Source(s) of stimulus | Student or teacher |
| Artifacts | System |
| Environment | Normal operating system |
| System response | Do not expose those functions to the user |
| Response measure(s) | Requires censor users to be able to use that function |

## Table 4.4: Quality Attributes: Modifiability

|  |  |
| --- | --- |
| ID | QA04 |
| Quality Attributes | Modifiability |
| Stimulus | Change some function in the future |
| Source(s) of stimulus | Customer |
| Artifacts | Developer |
| Environment | In the development or implementation phase |
| System response | System components can be easily changed or integrated with new features |
| Response measure(s) | Change time less than 10 days |

1. **Business goals**

## Table 5: Business goals

|  |  |
| --- | --- |
| **Business** | **Goals** |
| Cost | Design cost: Must be around $500. Production cost: Should be under $1700. Maintenance cost: Should be around $500. Operation cost: Should be under $200 |

* 1. **Dependencies**

## Table 6: Dependencies

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Dependency** | **Expected delivery date** | **Note** |
| 1 | Cloud Server | Dec 15th | Put all the necessary modules on the server cloud to ensure the continuity of the system |

## Project Risk

## Table 7: Risk

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk** | **Description** | **Probability** | **Severity** | **Mitigation Strategy** |
| Lack of coding experiences | No one in team member work with VueJS, spring MVC, Restfull API, Socket.io | 3 | 2 | Each team member has to learn and help the other to learn quickly. |
| Source Code conflict | Problems while merging code between members to master branch | 3 | 3 | Each team member must resolve conflicts by using git merge CLI before merging to the master branch. |
| Member conflict | Team member maybe conflict with each other while discussing | 3 | 2 | Team building, playing board games to get everyone together. |
| Less equipment | No machine or hosting for deploying the server. | 1 | 2 | Try free hosting for deployment. |
| Time management | Every member has to go to work or school. | 3 | 3 | Overtime |
| Language barrier | Most of the documents the are in English, sometimes it hard to understand clearly the articles and the information | 3 | 3 | Improve the individual English skills while doing the project. Asking the mentor technology for specific |

|  |  |  |  |
| --- | --- | --- | --- |
| **Probability** | | **Severity** | |
| 1 | Rarely happened. | 0 | Low damaged |
| 2 | Sometime happened | 1 | Medium damaged |
| 3 | Usually happened | 2 | Serious damaged |

# Project Development Approach

## Project Process

### *Reasons for selecting*

* Our team has 4 people
* The project will be continuously horizontally scaled up.
* There is only a short amount of time to finish the project.

So, based on those constraints, we decided to choose SCRUM as the project lifecycle

### *Agile Methodology*

**RINCIPLE AND DIFFERENT STAGES**

The SCRUM methodology relies on the incremental development of a software application while maintaining a completely transparent list of upgrade or correction demands to be implemented (backlog). It involves frequent deliveries, usually every four weeks, and the client receives a perfectly operational application that includes more and more features every time. This is why the method relies on iterative developments at a constant rhythm of 2-4 weeks. Upgrades can therefore be more easily integrated than when using a V-cycle.

This method requires four types of meetings:

* Daily meetings: the entire team meets for approximately 15 minutes every day in order to answer the following three questions, usually while standing: what did I do yesterday? What am I going to do today? Is there a cumbersome impediment today?
* Planning meetings: the entire team gathers to decide on the features that will make up the following sprint
* Work review meetings: during this meeting, every member presents what he has done during the sprint. They organize a demonstration of the new features or a presentation of the architecture. This is an informal meeting lasting for approximately 2 hours which is attended by the entire team.
* Retrospective meetings: at the end of each sprint, the team analyzes both successful and unsuccessful elements of their activity. During this meeting lasting between 15 and 30 minutes where everyone is invited and speaks on their own behalf, a vote of confidence is organized in order to decide on the improvements to be made.

The advantage of this method consists in reducing the documentation to the minimum  
in order to gain productivity. The idea is to write only the minimum documentation which allows to save the history of the decisions taken on the project and to easily perform interventions on the software when it goes into the maintenance phase.

**AGILE - SCRUM ORGANISATION**

The SCRUM methodology involves the following three main players:

* Product owner: In most projects, the product owner is the leader of the client's project team. He is the one who will define and prioritize the product features and choose the date and content of each sprint based on values (workloads) that the team communicates to him.
* Scrum Master: He is a genuine facilitator on the project as he makes sure that everyone works at their full potential by eliminating impediments and protecting the team from external interference. Moreover, he pays particular attention to the respect of the different SCRUM phases.
* Team: A team is typically made up of 4-10 people and groups together all the IT specialists who are necessary on a project, i.e. an architect, a designer, a developer, a tester, etc. The team is self-organizing and remains unchanged during an entire sprint

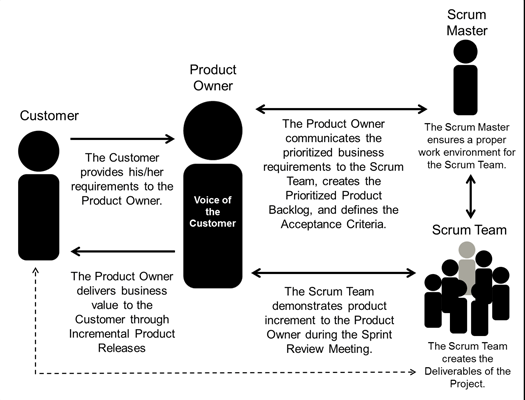


Figure 1: Scrum team members

**AGILE - SCRUM ADVANTAGE**

Scrum differs from other development methods through its advantages which turn it into a pragmatic response to product owners' current needs:

Iterative and incremental method: this allows to avoid the "tunnel effect", i.e. the fact of seeing the result only at the final delivery, and nothing or almost nothing during the entire development phase, which is so frequent with V-cycle developments.

Maximum adaptability for product and application development: the sequential composition of the sprint content allows the addition of a modification or a feature that was not initially planned. This is precisely what renders this method "agile".

* Participatory method: every team member is asked to express his opinions and can contribute to all the decisions taken on the project. He is therefore more involved and motivated.
* Enhancing communication: by working in the same development room or being connected through different communication means, the team can easily communicate and exchange opinions on the impediments in order to eliminate them as early as possible.
* Maximizing cooperation: daily communication between the client and the team enables them to collaborate more closely.
* Increasing productivity: as it removes certain "constraints" of the classical methods, such as documentation or exaggerated formalization, SCRUM allows for increased team productivity. By adding to this the qualification of each module which allows determining an estimation, everyone can compare their performance to the average team productivity.

1. **Scrum Process**

Scrum is an agile method, so it follows the principles of Agile Manifesto (see also Agile Manifesto). In addition, Scrum operates on three core values, also known as Scrip Scripps, including Scrutiny, Inspection, and Adaptation.

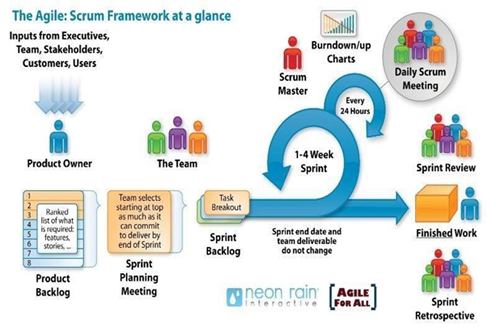


Figure 2: Scrum team members

Based on the empirical process control theory, Scrum uses iterative and incremental algorithms to optimize efficiency and control risk. Scrum is simple, easy to learn, and has wide applicability. To be able to use Scrum, we need to understand and apply the elements that makeup Scrum include the core values (also known as the "three legs", or the three pillars of Scrum), roles, Events, and Scrum-specific artifacts.

1. **Requirement Change Management**

We use Ince’s Change Process Model to handle the required changes. Ince’s model focuses on how software configuration management relates to software change management. This model has two main sources of change requests, i.e. customer and development team as shown in Figure 1. In order for the change process to be initiated, a change request must be initiated in a software project. All such change requests are recorded in a change request note. The change control board then considers the suggested change. The change control board can reject the change (the change will not take place), batch the change (the change will take place but not immediately), or accept the change (the change is to be implemented at the earliest possible time). If the request for the change is successful, a change authorization note must be filled. After this, the change can be implemented and a system’s documentation is modified. After implementation, the change is validated. Validation and test records are then produced to document the changes that have taken place.

Finally, the configuration records are updated and the staff is informed about the new changes.

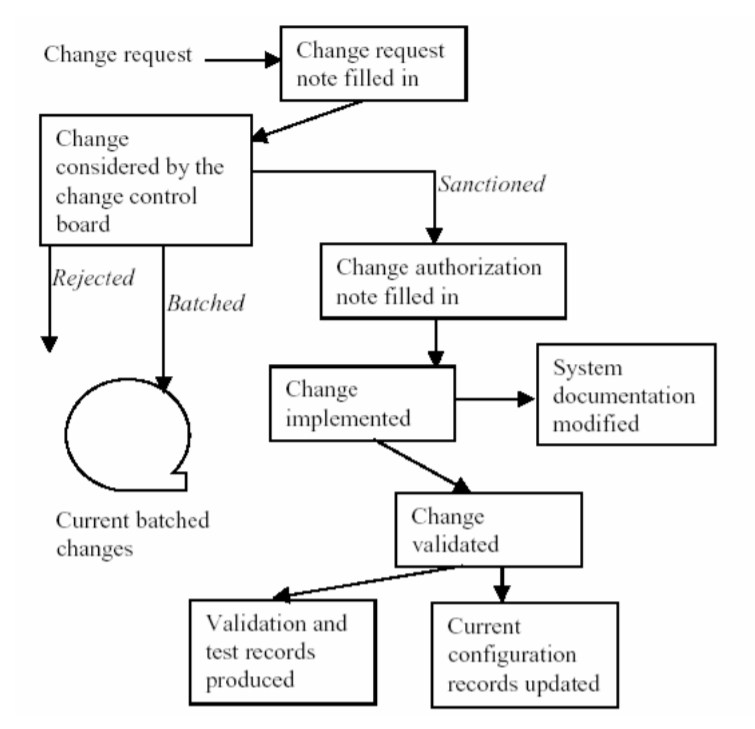


Figure 3: Ince’s change process model

1. **Product Integration Strategy**

The project is integrated with Continuous Integration (CI) methods.

* Develop describes the practices necessary to implement stories and commit the code and components to version control.
* Build describes the practices needed to create deployable binaries and merge development branches into the trunk.
* Test end-to-end describes the practices necessary to validate the solution.
* Stage describes the practices necessary to host and validate the solution in a staging environment before production.

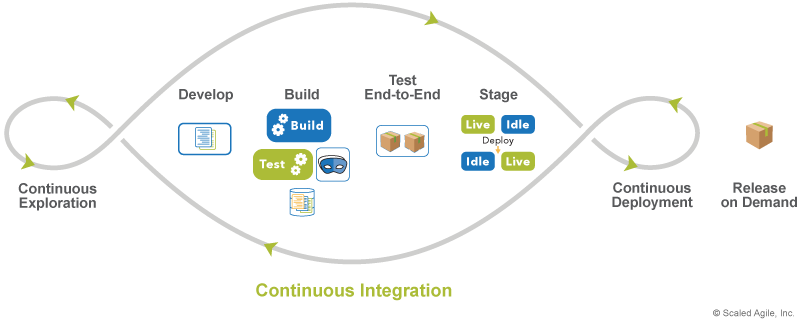


Figure 4: Continuous Integration

We decided to implement the CI method for SConA with Bitbucket Pipeline. Bitbucket Pipelines is an integrated CI/CD service, built into Bitbucket. It allows users to automatically build, test, and even deploy code based on a configuration file in them repository

## Quality Management

* + 1. **Estimates of Defects to be detected**

**Pre-release review defects**

## Table 8: Pre-release review defects

|  |  |  |
| --- | --- | --- |
| **Process** | **Planned found by review** | **Actual found by review** |
| Requirement | - Change business requirement - Change project architectural design requirements - Shortcomings in defining use cases and user stories | - Change business requirement - Change project architectural design requirements - Shortcomings in defining use cases and user stories - Change requirements on data processing and storage methods |
| Amount of work | - A lot of work causes overload for programmers who have a job | - A lot of work causes overload for programmers who have a job |
| Programming tools | - Inappropriate use of programming tools |  |
| Technologies stack | - Inappropriate use of technologies stack - Some technologies are no longer supported - Lack of new technology | - Inappropriate use of technologies stack - Some technologies are no longer supported - Lack of new technology - Difficulty in accessing paid technologies |
| Reference studies | - Shortage of research articles related to the system - Articles often have a large academic volume and are difficult to apply in practice | - Shortage of research articles related to the system - Articles often have a large academic volume and are difficult to apply in practice - Difficult to access internal research works |
| Design | - Old version UI is hard to reuse to extend the functionality | - Old version UI is hard to reuse to extend the functionality - Completely change the UI of the system |
| Automatic processing method | - Lack of supporting technologies and tools | - Lack of supporting technologies and tools - Data processing results are not quite as expected |
| Coding | - Difficult to implement - Use different languages for each module | - Difficult to implement - Use different languages for each module - Dirty code |
| System integration | - The difficulty of creating a fully automated pipeline to run different modules - It is very difficult to integrate quickly because the system is built on many different programming languages and technologies | - The difficulty of creating a fully automated pipeline to run different modules - It is very difficult to integrate quickly because the system is built on many different programming languages and technologies |
| Deliver on time | - Large workload leads to late deadline - Difficulty in implementation causing slow progress | - Large workload leads to late deadline - Difficulty in implementation causing slow progress |
| Other | - Lack of money to maintain the cloud server and organize weekly meetings | - Lack of money to maintain the cloud server and organize weekly meetings |
| Total | 18 | 24 |
|  |  |  |

1. **Strategy for Meeting Quality Objectives**

## Table 9: Strategy for Meeting Quality Objectives

|  |  |
| --- | --- |
| **Strategy** | **Expected Benefits** |
| Use standards and design patterns to implement, prevent unnecessary defects during system installation | 5-10% reduction in defect injection rate and about 2% improvement in productivity |
| Public project to perform public user acceptance test, ensure test samples are verified with a large number of users to evaluate system performance and find potential defects | Find out most defects in UI/UX and data processing, take corrective measures as soon as possible, through which can improve the system quality in the best way |
| Write as many test cases as possible, make sure to cover all test cases | Ensure that the testing process is continuous without being forced at the end of the project, increasing the quality of each module of the system. |
| Use rolling wave planning to execute test cases continuously during development | Approximately 5% reduction in defect injection rate and 1% improvement in overall productivity |
| Use proven technologies, absolutely do not use tools that are no longer supported | Ensure the continuity and scalability of the system, avoiding the case of sudden system death when technologies stop supporting |

* + 1. **Quality Control**

## Table 10: Quality Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Review Item** | **Type of Review** | **Reviewer** | **When** |
| System Requirement Specification | One-person review | Scrum Master | Before the project kick-off meeting |
| Scope, Objective, and Goal | One-person review | Scrum Master | End of requirements 90% |
| System architecture specification | Group review | Scrum Master, Developer | After finishing System Requirement Specification |
| UI/UX Design | Group review | Scrum Master, Developer | After finishing the System architecture specification |
| Project Plan, Scrum Schedule | Group review | Scrum Master, Developer, Tester | After finishing the System architecture specification |
| Resource allocation | One-person review | Scrum Master | After finishing Project Plan |
| Design document, object model | Group review | Scrum Master, Developer, Tester | End of 90% design |
| Database design | Group review | Scrum Master, Developer | After finishing object model design |
| Testing plan | Group review | Scrum Master, Tester | End of Sprint 1 |
| Code | Group review | Scrum Master, Developer | After each module built |
|  |  |  |  |

* + 1. **Measurements Program**

## Table 11: Measurements

|  |  |  |  |
| --- | --- | --- | --- |
| **Data to be collected** | **Purpose** | **Responsible** | **When** |
| Size: No. of KLOC// FP | Group review or One-person review | Project Manager / Scrum Master | At the end of stages |
| Effort: No. person-day |  | Team members | Daily |
| Quality: No. defects detected |  | Reviewer, Tester | Right after the review/test |
| Schedule |  | Project Manager / Scrum Master | Weekly and at the end of stages |

## Unit Testing Strategy

In this project, most of the functionality is highly specific, so we focus on using manual testing.

**Tool**: Manual Test: Google Sheet

There are principles while doing unit tests:

* For each class, there should be a test class that tests all the public methods.
* Tests cover at least positive tests and negative tests.
* Dependencies to other classes should be substituted by mock objects.
* Each test case covers exactly one functionality to achieve a quick bug fixing.
* For each abstract class, an abstract test class will be implemented. This abstract test class tests the implementation parts of the abstract class and outlines the correct use of the abstract class and the test classes to implement for the concrete classes.

**Requirements:**

* At least 70% of lines of code get coverage.
* Unit tests must be done once before the tester executes test cases in each sprint.
* Unit tests must be performed consistently, ensuring that there is no fluctuation in the number of unit tests per module.

## Integration Testing Strategy

Testers must design a test plan for integration testing with both Sub-component integration testing and Component integration testing because a component is typically composed of a set of sub-components and these sub-components consist of classes and packages.

* Sub-component integration testing: The integration tests between sub-components ensure compatibility of these software units across development cycles. An example of a sub-component integration test in the Design Environment is the validation of outputs from the Modelling Library against the expected inputs of the Modelling Tool.
* Component integration testing: These tests will examine the compatibility of components’ APIs according to a predefined set of rules. Furthermore, they examine the basic functionalities of a group of components as a sub-system. An example of such tests is between Design and Algorithm Optimization environments, where the outputs of the Modelling Tool will be tested against expected inputs of the Optimization Tool

## System Testing Strategy

For system testing, we use the following technique:

* Usability Testing: Test focus on user-friendly, easy-to-interact interface, flexibility in handling controls, and ability of the system to meet project objectives.
* Load Testing: Check whether the performance of the system is balanced and stable, whether the criterion is met in quality attribute or not.
* Functional Testing: provide missing functionality, functions list that testers think are likely to affect the product.

# Estimation

## Size

## Table 12: Estimation Size Project

|  |  |
| --- | --- |
| **Modules** | **LOC** |
| UI | 1300 |
| Database Design | 550 |
| Function | 1500 |
| Restfull API | 600 |
| Total | 3950 |

* 1. **Effort**

## Table 13: Estimation Effort Project

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Total budgeted Effort Usage (pd)** | **Total % budgeted Effort Usage (%)** |  | | |  | | |  | | |  | | |
| **Activity/** | **Sprint 1** | | | **Sprint 2** | | | **Sprint 3** | | | **Sprint 4** | | |
| **Process** |  |  |  | |  |  | |  |  | |  |
|  | **No.** | **%** | **No.** | | **%** | **No.** | | **%** | **No.** | | **%** |
| Requirement | 250 | 11.03 | 70 | 28 | 60 | | 24 | 60 | | 24 | 60 | | 24 |
| Design | 300 | 13.24 | 90 | 30 | 70 | | 23.3 | 70 | | 23.3 | 70 | | 23.3 |
| Coding | 725 | 31.99 | 185 | 25.5 | 185 | | 25.5 | 177.5 | | 24.48 | 177.5 | | 24.48 |
| Unit Testing | 150 | 6.62 | 40 | 26.7 | 40 | | 26.7 | 35 | | 23.3 | 35 | | 23.3 |
| Testing | 100 | 4.41 | 25 | 25 | 25 | | 25 | 25 | | 25 | 25 | | 25 |
| Deployment | 100 | 4.41 | 0 | 0 | 0 | | 0 | 25 | | 25 | 75 | | 75 |
| Support for Acceptance Test | 110 | 4.85 | 21 | 19.09 | 21 | | 19.09 | 34 | | 30.91 | 34 | | 30.91 |
| Project Planning | 200 | 8.83 | 70 | 35 | 70 | | 35 | 30 | | 15 | 30 | | 15 |
| Project monitoring | 175 | 7.72 | 43.75 | 25 | 43.75 | | 25 | 43.75 | | 25 | 43.75 | | 25 |
| Quality Assurance | 100 | 4.41 | 25 | 25 | 25 | | 25 | 25 | | 25 | 25 | | 25 |
| Training | 30 | 1.32 | 30 | 100 | 0 | | 0 | 0 | | 0 | 0 | | 0 |
| Total | **2266** | **100** | **599.75** | **26.47** | **539.75** | | **23.82** | **525.25** | | **23.18** | **575.25** | | **25.39** |

## Schedule

* + 1. **Project Milestone & Deliverables**

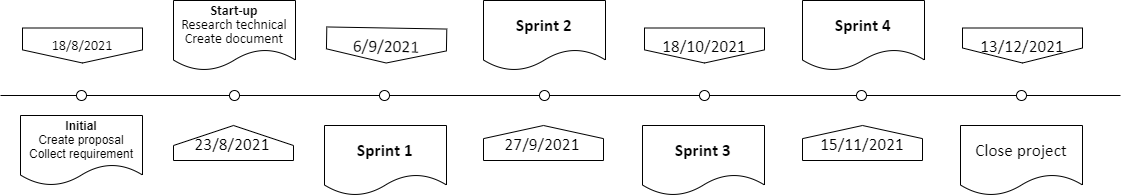


Figure 5: Project Timeline

**DELIVERABLES**

## Table 14: Deliverables Project

|  |  |  |
| --- | --- | --- |
| **No** | **Activities** | **Deliverables** |
| 0 | Requirements Description | Requirements Description 1.0 |
| 1 | Project Proposal | Project Proposal Document 1.0 |
| 2 | Project Plan | Project Plan Document 1.0 |
| 3 | Product Backlog | Product Backlog and User story Document 1.0 |
| 4 | Architecture Document | Architecture Document 1.0 |
| 5 | Database Design | Database Design Document 1.0 |
| 6 | Interface Design | Interface Design Document 1.0 |
| 7 | Test Plan | Test Plan Document 1.0 |
| 8 | Test Case | Test Case Document 1.0 |
| 9 | Sprint Backlog & Burndown Chart | Sprint Backlog & Burndown Chart Document 1.0 |

* + 1. **Work Breakdown Structure**

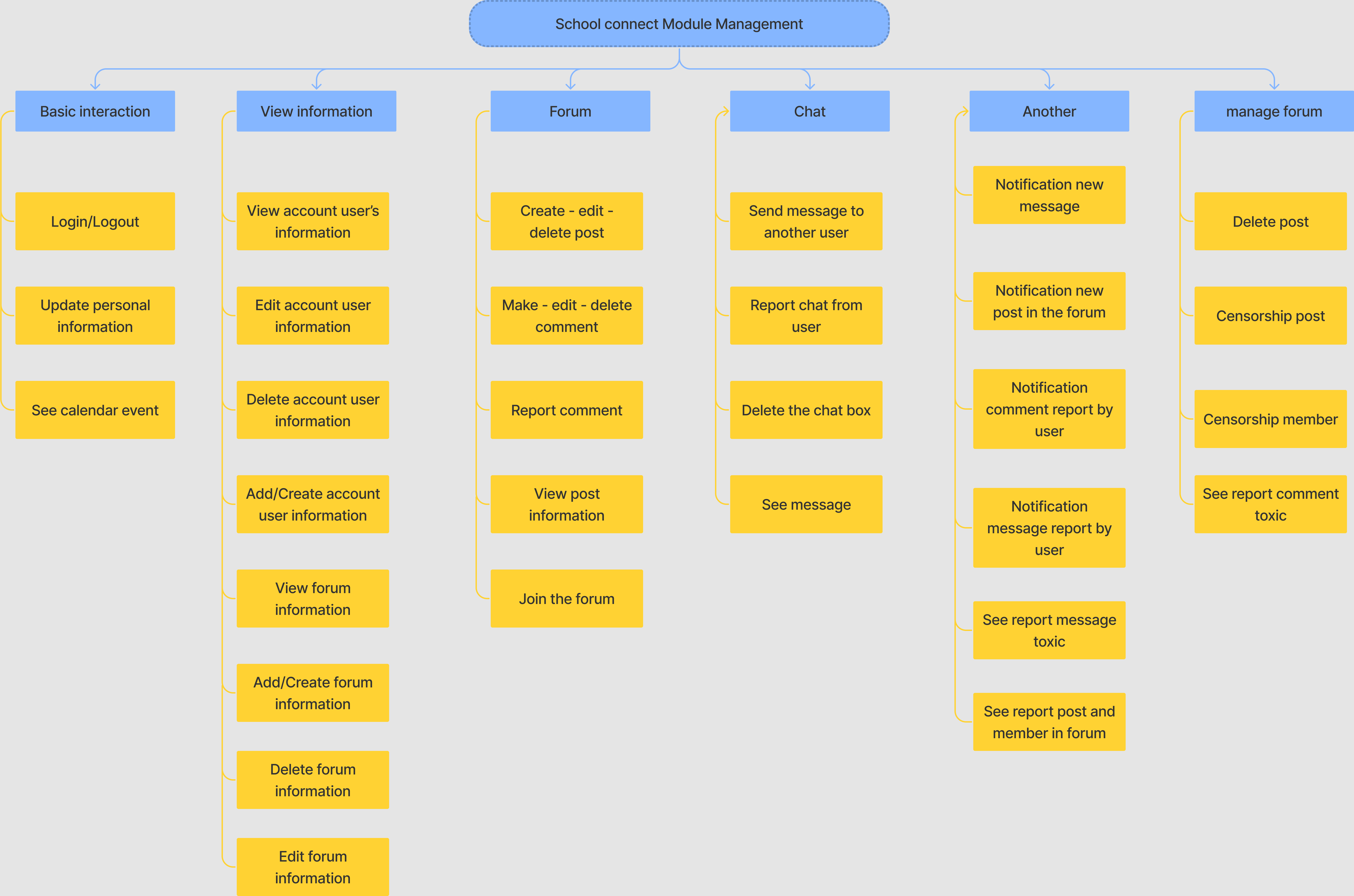


Figure 6: Work Breakdown Structure

* + 1. **Detailed Schedule**

## Table 15: Schedule Project

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Task Name** | **Duration (Days)** | **Start** | **Finish** | **Assign to** |
| **1.** | **Initial** | **19** | **18/8/2021** | **05/9/2021** |  |
| 1.1 | Project’s Kick-off Meeting | 5 | 18/8/2021 | 22/8/2021 | Team |
| 1.2 | Proposal | 4 | 19/08/2021 | 20/08/2021 | Phu |
| 1.3 | Research technical | 10 | 20/08/2021 | 30/08/2021 | Team |
| 1.4 | Project plan | 4 | 31/08/2021 | 01/09/2021 | Phu |
| 1.5 | Product backlog | 4 | 02/09/2021 | 03/09/2021 | Hieu |
| 1.6 | User story | 4 | 04/09/2021 | 05/09/2021 | Hoai |
| **2** | **Development** | **98** | **06/9/2021** | **12/12/2021** |  |
| 2.1 | Sprint 1 | 28 | 06/9/2021 | 03/10/2021 |  |
| 2.1.1 | Sprint 1 backlog |  |  |  | Hieu |
| 2.1.2 | Test plan |  |  |  | Thang |
| 2.1.3 | Design architecture |  |  |  | Phu |
| 2.1.4 | Design database sprint 1 |  |  |  | Phu |
| 2.1.5 | Design UI page login |  |  |  | Hoai, Hieu |
| 2.1.6 | Code function login |  |  |  | Phu |
| 2.1.7 | Test case for login |  |  |  | Thang |
| 2.1.8 | Design UI homepage |  |  |  | Hieu, Hoai |
| 2.1.9 | Design UI page Create/edit post |  |  |  | Hieu |
| 2.1.10 | Test case for create/edit post |  |  |  | Thang |
| 2.1.11 | Code create/edit post |  |  |  | Phu |
| 2.1.12 | Design UI page to add/modify account user information |  |  |  | Hoai |
| 2.1.13 | Test case for add/modify account user information |  |  |  | Thang |
| 2.1.14 | Code add/modify account user information |  |  |  | Phu |
| 2.2.15 | Test report sprint 1 |  |  |  | Thang |
| 2.2 | Sprint 2 | 28 | 4/10/2021 | 31/10/2021 |  |
| 2.2.1 | Sprint 2 backlog |  |  |  | Hieu |
| 2.2.2 | Design database sprint 2 |  |  |  | Phu |
| 2.2.3 | Design UI page view post information |  |  |  | Hieu |
| 2.2.4 | Design UI page view account information |  |  |  | Hoai |
| 2.2.5 | Code view post information |  |  |  | Phu |
| 2.2.6 | Code view account information |  |  |  | Phu |
| 2.2.7 | Design UI page view and create/modify forum information |  |  |  | Hoai |
| 2.2.8 | Design UI page censorship post and member |  |  |  | Hieu |
| 2.2.9 | Test case censorship post and member |  |  |  | Thang |
| 2.2.10 | Code view and create/modify forum information |  |  |  | Phu |
| 2.2.11 | Code censorship post and member |  |  |  | Phu |
| 2.2.12 | Design UI comment |  |  |  | Hieu |
| 2.2.13 | Design UI page see report post and member in forum |  |  |  | Hoai |
| 2.2.14 | Test case see report post and member in forum |  |  |  | Thang |
| 2.2.15 | Test case comment |  |  |  | Thang |
| 2.2.16 | Code comment |  |  |  | Phu |
| 2.2.17 | Code see report post and member in forum |  |  |  | Phu |
| 2.2.18 | Test report sprint 2 |  |  |  | Thang |
| 2.3 | Sprint 3 | 21 | 01/11/2021 | 21/11/2021 |  |
| 2.3.1 | Sprint 3 backlog |  |  |  | Hieu |
| 2.3.2 | Design database sprint 3 |  |  |  | Phu |
| 2..3.3 | Design UI sent message |  |  |  | Hoai |
| 2.3.4 | Design UI delete post, member, comment, chat box |  |  |  | Hieu |
| 2.3.5 | Test case sent message |  |  |  | Thang |
| 2.3.6 | Test case notify new post |  |  |  | Thang |
| 2.3.7 | Code sent message |  |  |  | Phu |
| 2.3.8 | Code notify new post |  |  |  | Phu |
| 2.3.9 | Code delete post and member |  |  |  | Phu |
| 2.3.10 | Code delete comment |  |  |  | Phu |
| 2.3.11 | Code delete chat box |  |  |  | Phu |
| 2.3.12 | Design UI notify new message and new post |  |  |  | Hoai |
| 2.3.13 | Design UI report message and comment |  |  |  | Hieu |
| 2.3.14 | Test case notify new message |  |  |  | Thang |
| 2.3.15 | Test case report message and comment |  |  |  | Thang |
| 2.3.16 | Code notify new message and new post |  |  |  | Phu |
| 2.3.17 | Code report message and comment |  |  |  | Phu |
| 2.3.18 | Test report sprint 3 |  |  |  | Thang |
| 2.4 | Sprint 4 | 21 | 22/11/2021 | 12/12/2021 |  |
| 2.4.1 | Sprint 4 backlog |  |  |  | Hieu |
| 2.4.2 | Design database sprint 4 |  |  |  | Phu |
| 2.4.3 | Design UI see report message, comment |  |  |  | Hoai |
| 2.4.4 | Design UI update personal |  |  |  | Hieu |
| 2.4.5 | Test case see report message and comment |  |  |  | Thang |
| 2.4.6 | Test case update personal |  |  |  | Thang |
| 2.4.7 | Code see report message and comment |  |  |  | Phu |
| 2.4.8 | Code see update personal |  |  |  | Phu |
| 2.4.9 | Design UI see calendar |  |  |  | Hieu |
| 2.4.10 | Design UI notify message and comment report by user |  |  |  | Hoai |
| 2.4.11 | Test case see calendar |  |  |  | Thang |
| 2.4.12 | Test case notify message and comment report by user |  |  |  | thang |
| 2.4.13 | Code see calendar |  |  |  | Phu |
| 2.4.14 | Code notify message and comment report by user |  |  |  | Phu |
| 2.4.15 | Test report sprint 4 |  |  |  | Thang |
| **3** | **Close project** | 2 | 13/12/2021 | 15/12/2021 |  |
|  | **Duration** | 119 |  |  |  |

## Resource

* 1. **Human Resources**

## Table 16: Human Resources

|  |  |  |
| --- | --- | --- |
| **Position Titles** | **Quantity** | **Member** |
| Backend Developer | 2 | Thanh Phu |
| Frontend Developer | 1 | Hieu, Hoai |
| Data Engineer | 2 | Thanh Phu |
| Graphics Designer | 1 | Hieu, Hoai |
| Business Analyst | 1 | Hieu, Hoai, Phu |
| Tester | 1 | Thang |
| Project Manager | 1 | Thanh Phu |
| Total Human-Resources | 9 |  |

1. **Equipment**

## Table 17: Equipment Resources

|  |  |
| --- | --- |
| **Name of equipment** | **Quantity** |
| Laptop | 4 |
| Monitor | 4 |
| Tablet | 1 |

## Infrastructure

## Table 18: Infrastructure

|  |  |  |  |
| --- | --- | --- | --- |
| **Work/Product** | **Purpose** | **Expected Availability by** | **Note** |
| **Development Environment** | | | |
| Win 10 | Operating Systems | Initiation stage |  |
| MySQL, Firebase | DBMS | Initiation stage |  |
| JavaScript | Development language for Web interface and API server | Initiation stage |  |
| Java | Development language for data processing, handling data sources and handling WebSocket. | Initiation stage |  |
| **Hardware & Software** | | | |
| Tomcat server | 2vCPUs, 1GB memory, 5GB storage, version 9.30 |  |  |
| Adobe Xd | UI Design & Architecture design |  |  |
| **Other Tools** | | | |
| GitHub, Bitbucket | Source version control | Initiation stage |  |
| Excel Office | Effort logging | Initiation stage |  |
| Slack | Discussion | Initiation stage |  |
| Jira | Task tracking | Initiation stage |  |
| Word Office | Managing documents | Initiation stage |  |
| Google met | Meeting online | Initiation stage |  |

## Training Plan

## Table 19: Training Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Training Area** | **Participants** | **When, Duration** | **Waiver Criteria** |
| Technical | | | |
| VueJs framework | Hoai, Hiieu | 7 days | If already trained |
| Spring MVC | Phu | 7 days | If already trained |
| WebSocket | Phu | 7days | If already trained |
| Restfull API | Hoai, Hieu, Phu | 5 days | If already trained |
| Business domain | | | |
| Environment |  | 7 days |  |
| Process | | | |
| Quality system | All members | 3 hrs. | If already trained |
| Configuration management | All members | 2 hrs. | If already trained for CC. For others, on-the- job training |
| Group review | All members | 4 hrs. | If already trained |
| Defect prevention | All members | 4.5 hrs. | Mandatory |

**Additional training program**

The way of working on a Scrum project.

* Release Planning Meeting
* Sprint Planning Meeting
* Daily Scrum Meeting
* Sprint Review Meeting
* Retrospective
* Scrum of Scrums

## Finance

## Table 20: Finance

|  |  |
| --- | --- |
| **Cost Description** | **Detail** |
| Salary | Duration: 17 weeks (119 days) Man-hour: 2hours/1day Salary: $2/1 hour Persons: 4 members Overtime cost per hour: 1.5$/hour The salary of 1 person: $567 Total: $2266 |
| Laptop | Laptop for each member: $800 Total: $3200 |
| Monitor | LG Monitor for each member: $100 Total: $400 |
| Maintenance | Cost per month: $400 |
| Bugs fixing | Cost per bug: $20 |
| Total | Salary + Laptop + Monitor  Total: $5904 |

# Project Organization

## Organization Structure

## . Table 21: Organization Structure

|  |  |  |
| --- | --- | --- |
| **Role** | **Responsibility** | **Name** |
| ProductOwner | Understand the user and customers with their needs. Collaborate with the development team. Manage the stakeholders. Describe the user experience and product features. Provides detailed user stories. | Nguyen Trung Hieu,  Dang Nguyen Bao Hoai |
| ScrumMaster | Communicate the value of Scrum Teach the organization on Scrum to maximize business value Facilitate Sprint Planning, Daily Scrums, Sprint Reviews, and Retrospective Meetings Create the Task Board and Sprint Burndown Chart at the start of every Sprint Attend all Scrum meetings Preserve the integrity and spirit of the Scrum framework Maintain the focus of the Team Make the Team aware of impediments and facilitate efforts to resolve them Serve as a coach and mentor to members of the Team Respectfully hold the Team, Product Owner, and Stakeholders accountable for their commitments Continually work with the Team and business to find and implement improvements | Nguyen Thanh Phu |
| Secretary | Record the content of group meetings and activities of the member | Nguyen Trung Hieu |
| Reviewer | Analysis of the functions and requirements of the product. Review documents related to the project | Nguyen Thanh Phu |
| TeamLeader | Back-end Dev: Server, WebSocket, Database, Restfull API | Nguyen Thanh Phu |
| Teammember | Frontend Developer, UI/UX Designer | Dang Nguyen Bao Hoai |
| Teammember | Frontend Developer, UI/UX Designer | Nguyen Trung Hieu |
| Teammember | Tester | Nguyen Van Thang |

## Project Team

## Table 22: Project Team

|  |  |  |  |
| --- | --- | --- | --- |
| **Full Name** | **Phone** | **Email** | **Position** |
| Nguyen Minh Nhat | +84 905 125 143 | nhatnm2010@gmail.com | Mentor |
| Nguyen Thanh Phu | +84 772 492 301 | thangphu104@gmail.com | Scrum master, Dev Team |
| Nguyen Trung Hieu | +84 975 299 149 | hnguyentrung20@gmail.com | Product Owner, Dev Team |
| Dang Nguyen Bao Hoai | +84 773 305 395 | dangbhoai@gmail.com | Product Owner, Dev Team |
| Nguyen Van Thang | +84 906 421 132 | thangnguyendien@gmail.com | Dev Team |

1. **Communication & Reporting**
2. **Reporting Methodology**

## Table 23: Reporting Methodology

|  |  |  |  |
| --- | --- | --- | --- |
| **Audience/ Attendees** | **Topic/ Deliverable** | **Frequency** | **Method** |
| - Product Owner - Scrum Master - Team Members | Project Progress Review | Weekly | Email, google met |
| - Product Owner - Scrum Master - Team Members | Explicit Requirement | When needed | Email, google met |
| - Mentor - Scrum master - Team members | Milestone review | End of each Milestone | google met |
| - Scrum master - Team members | Daily tasks | Each day | Slack, Discord, Jira |

* 1. **Communication Methodology**

## Table 24: Communication Methodology

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type of Communication** | **Communication Schedule** | **Communication way** | **Who Initiates** | **Recipient** |
| Status Report ( Daily meeting) | Daily at 10AM | Slack | Scrum Master | Scrum Team |
| Schedule and Effort Tracking | Daily | Google sheet,  Jira | Scrum Master | Scrum Team |
| Work Review | Daily at 10AM | Trello | Scrum Master | Scrum Team |
| Work Report | Every Friday | Slack | Scrum Master | Scrum Team |
| Project Review, ask problems | Every Friday (flexible) | Google met | Scrum Master | Mentor, Scrum Team |
| Ask & Review problems | Anytime | Slack, gg met | Scrum's Member | Scrum Team,  Mentor, |

1. **Configuration Management**

Table 25: Configuration Management

|  |  |  |
| --- | --- | --- |
| **No** | **Tool** | **Content** |
| 1 | Excel office | Track member activities. At the end of each day, team members will post on time log and Scrum Master will check. |
| 2 | Word office | Track the changing of documents & manage versions of document |
| 3 | Bitbucket | Repositories for source code version management. |
| 4 | Weekly Meeting | Hold a meeting every week to assign tasks to each member. If there are some emergencies but we cannot sit together then we can use Discord to discuss online. |
| 5 | Document | All meetings must be documented and pictured. |
| 7 | Slack | Store document resources and designed components, Daily Scrum |
| 8 | Discord | Discuss online, stream and share problems |
| 9 | GitHub bitbucket | Repositories for open-source code of the project |

# Security Aspects

To deal with security risks, SConA is precisely process-configured at every stage of the project.

* Concept and planning

The purpose of this stage is to define the application concept and evaluate its viability.

This includes developing a project plan, writing project requirements, and allocating human resources.

* Prepare a list of security requirements for the project.
* Training sessions provide essential security knowledge ranging from basic threat awareness to in-depth information on secure development.
* Architecture and design

The purpose of this stage is to design a product that meets the requirements. This includes modeling the application structure and its usage scenarios, as well as choosing third-party components that can speed up development. The result of this stage is a design document.

* Secure design: The design document and subsequent updates are validated in light of the security requirements. Early design reviews assist in identifying features exposed to security risks before they are implemented.
* Implementation
* This is the stage at which an application is actually created. This includes writing the application code, debugging it, and producing stable builds suitable for testing.
* Guides and checklists remind programmers of typical mistakes to be avoided, such as storing unencrypted passwords. Enforcing secure coding principles eliminates many trivial vulnerabilities and frees up time for other important tasks.
* Manual code reviews are still a must for building secure applications. Timely reviews help developers to flag and fix potential issues before they shift attention to other tasks.
* Testing and bug fixing

The purpose of this stage is to discover and correct application errors. This includes running automatic and manual tests, identifying issues, and fixing them. We use the rolling wave testing method to perform continuous testing after each phase, which ensures that security issues will be detected and resolved as soon as possible.

* Release and maintenance

At this stage SConA goes live, with many instances running in a variety of environments. Before going live, we set up security for the cloud server, install SSL for hosting and related procedures to manage the cloud environment as closely as possible.

1. **References**

Table 26: References

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Reference item** | **Issued Date** | **Source** | **Note** |
| 1 | [SConA]Proposal: | 01-Sep-2021 | https://bitbucket.org/PhuNguyenThanh/capstone\_1/src/master/docs/1%20PROPOSAL/C1SE.44-Proposal-SConA.docx |  |
| 2 | What is Scrum: | 02-Sep-2021 | https://www.scrum.org/resources/what-is-scrum |  |
| 3 | Design: | 02-Sep-2021 | https://www.adobe.com/products/xd.html |  |
| 4 | Security Aspect: | 02-Sep-2021 | https://www.ptsecurity.com/ww-en/analytics/knowledge-base/how-to-approach-securesoftware-development/ |  |
| 5 | Integrating Quality Management System into Software Development Processes: | 02-Sep-2021 | https://assist-software.net/blog/integrating-quality-management-system-software-devel opment-processes |  |