AI-Powered Compliance Monitoring for Retail Staff

SOFTWARE QUALITY ASSURANCE PLAN

Group 4

Name	Position	Email	Phone
Vuong Khang Minh	Document Researcher, Document Organizer, AI Specialist	104179690@student.swin.edu.au	0967572933
Nguyen Dang Duc Anh	Software Specialist, Requirement Analysis	104182520@student.swin.edu.au	0947023256
Nguyen Ha Huy Hoang	Team leader/Project Manager	103487444@student.swin.edu.au	0358791781
Nguyen Dang Khanh Toan	Quality Assurance, Domain Expert, Software Specialist	103487389@student.swin.edu.au	0888752003
Nguyen Cuong Nhat	AI Specialist, Requirement Analysis	104178590@student.swin.edu.au	0974247189

COS40005 - Computing Technology Project A, Semester 3, May 2024

Review History

Version	Date	Author	Summary of Changes
1.00	17/06/2024	All authors	Created the initial draft, including Introduction, Purpose, and Reference Documents
1.01	20/06/2024	Vuong Khang Minh	Added detailed roles and responsibilities in Management section
1.02	25/06/2024	Nguyen Dang Duc Anh	Refined documentation in Documentation section, added SQAP and SRS details
1.03	28/06/2024	Nguyen Ha Huy Hoang	Updated Standards, Practices, Conventions, and Metrics; integrated team feedback
1.04	02/07/2024	Nguyen Dang Khanh Toan	Conducted quality assurance, revised Testing and Problem Reporting sections
1.05	05/07/2024	Nguyen Cuong Nhat	Enhanced AI-related sections in Tools and Methodologies, added Risk Management details
1.06	10/07/2024	Vuong Khang Minh	Finalized documentation for submission, incorporated final feedback and edits across all sections

Acronyms/Abbreviations

AI – Artificial Intelligence

SDLC – Software development life cycle

IEEE - Institute of Electrical and Electronics Engineers

SQAP - Software Quality Assurance Plan

SRS - Software Requirements Specification

CONTENTS

CHAPTER 1: INTRODUCTION

1.1. Author List/Roles

Name	Position semester 1	Position semester 2
Nguyen Ha Huy Hoang	Team leader/Project Manager	Team leader/Project Manager
Vuong Khang Minh	Document Researcher, Document Organizer, AI Specialist	AI Engineer, Document Manager
Nguyen Dang Khanh Toan	Quality Assurance, Domain Expert, Software Specialist	Software Engineer, Quality Assurance, Backend
Nguyen Dang Duc Anh	Software Specialist, Requirement Analysis	Software Engineer, Devops, Frontend, Solution Architect
Nguyen Cuong Nhat	AI Specialist, Requirement Analysis	AI Engineer, Solution Architect

1.2. Purpose

The purpose of this plan is to define the Software Quality Assurance plan for Employee Compliance Checker which consist of outlining the quality requirements and standards; SQA tasks and guideline on how to perform such tasks; providing document and describes the practices and conventions for conducting SQA tasks, and identifies the tools, techniques, and methodologies to support SQA activities and reporting.

CHAPTER 2: REFERENCE DOCUMENTS

1. Laporte, C.Y. and April, A. (2018). Software quality assurance. Hoboken, Nj: Ieee Press.
2. Institute of Electrical and Electronics Engineers (IEEE) Std 730-1998, IEEE Standard for Software Quality Assurance Plans
3. van Rossum, G., Warsaw, B. and Coghlan, N. (2001). PEP 8 – Style Guide for Python Code. [online] peps.python.org. Available at: https://peps.python.org/pep-0008/.
4. Atlassian (n.d.). Confluence Best Practices. [online] Atlassian. Available at: https://www.atlassian.com/software/confluence/resources/guides/get-started/best-practices#stay-organized [Accessed 16 Jun. 2024].
5. Atlassian Community. (2019). Seven best practices for agile in Jira (Medium.com). [online] Available at: https://community.atlassian.com/t5/Philadelphia-discussions/Seven-best-practices-for-agile-in-Jira-Medium-com/td-p/1074771 [Accessed 16 Jun. 2024].
6. GitHub Docs. (n.d.). Best practices for Projects. [online] Available at: https://docs.github.com/en/issues/planning-and-tracking-with-projects/learning-about-projects/best-practices-for-projects.
7. tutorialspoint (2019). SDLC Spiral Model. [online] www.tutorialspoint.com. Available at: https://www.tutorialspoint.com/sdlc/sdlc_spiral_model.htm.

CHAPTER 3: MANAGEMENT

This section aims to describe the organizational structure of this project, its workflow, roles and responsibilities.

3.1. Organization/Roles

Since the team is of small size, the organizational structure can be divided into 4 categories based on the focus of the current task.

- Documentation task: if the current task final output is an document
- Research Task: if the current task is to propose idea/solution for a problem
- Developing task: if the current task is to show running output
- Testing task: if the current task is to check the performance of developed/researched solution

3.1.1. Documentation Roles

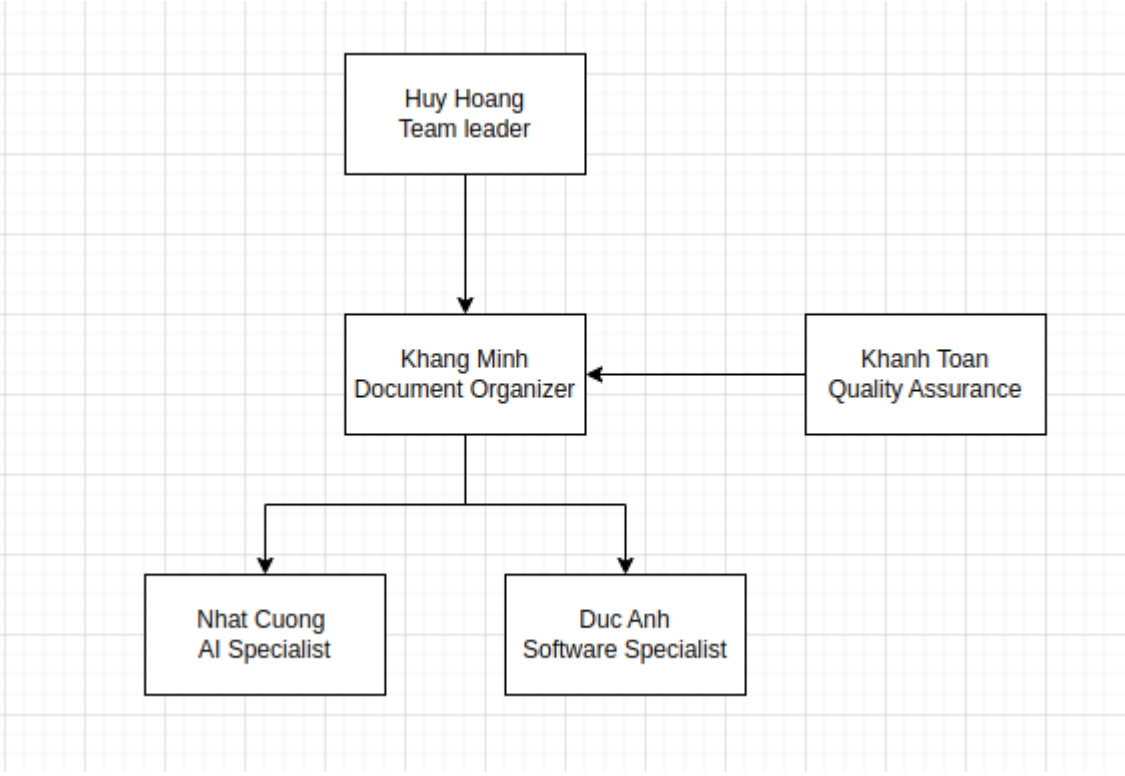

Figure 1. Team Document Task Structure

- **Document Organizer:** This person will be in charge of organizing the documentation written by everyone in the team and making sure that the doc is readable and has a

nice format. The person in this role will also need to prepare the documentation template and make it ready whenever new documentation is required.

- **Requirement Analysis:** This person will need to have a deep understanding about the requirement
- **AI Specialist:** The AI specialist will give the project an insight about how can it be solved with AI, what will be required to archive this and what should be avoid when working the propose AI solution
- **Software Specialist:** The Software Specialist will provide knowledge about the overall workflow and architecture of the project, how it can be deployed and how the user can communicate with the product developed.
- **Quality Assurance:** The document quality assurance will need to verify the content written inside the documentation to make sure that it is aligned with the team overall understanding of the project.

3.1.2. Research Task

- **Research Leader:** The research leader holds responsibility to analyze the problem and divide it into tasks for other team members to perform researching.
- **Devops:** The devops is responsible for preparing the environment for the research idea proposed by other researchers, and also setup the experiment for the team
- **AI researcher:** AI researcher's duty is to study about the state-of-the-art as well as the traditional solution for the problems/tasks identified by the researcher leader.
- **Policy & Requirement Researcher:** the person in charge of this position will need to clarify the legality risk of the project and study about the rightness of the proposed method, thus foreseeing if the method violates any privacy right.

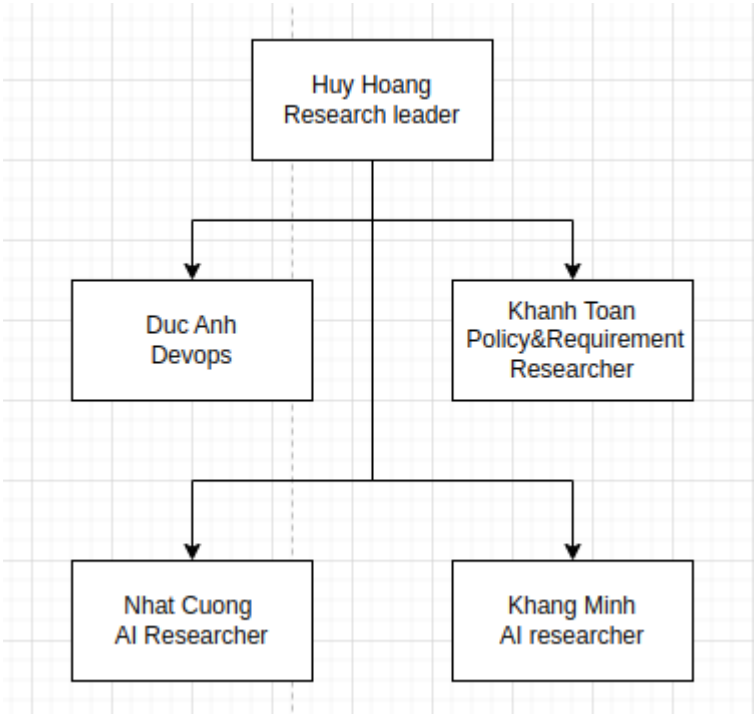

Figure 2. Team research task structure

3.1.3. Developing task

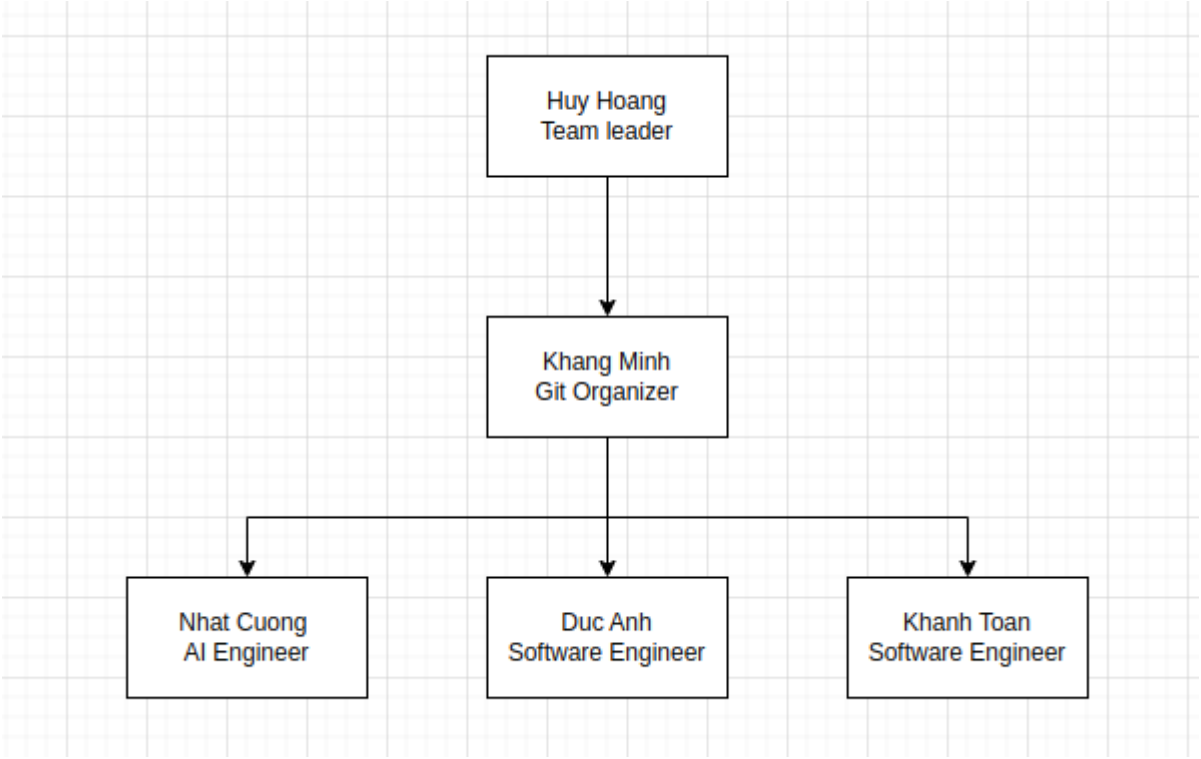

Figure 3. Team development task structure

Team leader: In developing task, the team leader's responsibility is to check working progress of the team and make appropriate adjustment to the solution to ensure the final outcome

Git Organizer: This person will control the pull request flow of the team, verify if the pushed code is appropriate and update the version of the github code. Moreover, this person will also control the version of the final solution.

AI engineer: AI engineer's main task is to develop AI algorithms for the project and align the AI model and workflow to the product workflow.

Software engineer: Software engineer's role is to develop an user friendly interface for the user to interact with the system and ensure that the user are having good user experience.

3.1.4. Testing role

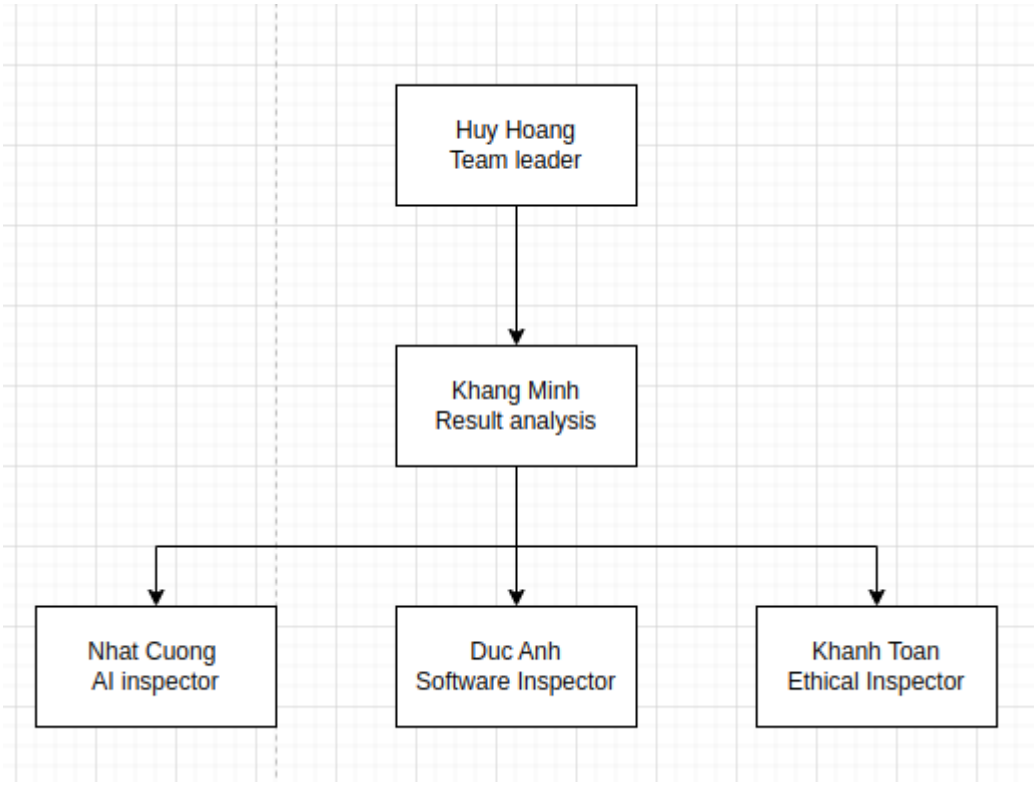

Figure 4. Team testing task structure

- **Team leader:** In the testing task, team leader will need to inspect the result of the test and make decision on the next step of the project

- **Result analysis:** The result analysis will need to analyze the recorded result from other team members to draw meaningful understanding about it.

- **AI Inspector:** This person will evaluate the performance and efficiency of the AI solution during the test, and report this back to the Result Analysis

- **Software Inspector:** This person will record the user experience and security analysis of the system during the test, then report this to the Result Analysis

- **Ethical Inspector:** This person will pay attention to the potential ethical risk of the product during the setup of the test.

3.1.5. Meeting role

Chair: This person will elaborate the main topic of the meeting and can be any team member who found a problem that required attention and mutual understanding between members.

Sargent at arm: Khang Minh will be the person who takes note of everything discussed in the meeting. If Khang Minh is holding the chair position, then Hoang will be the one who does it.

3.2. Tasks and Responsibilities

3.2.1. General team member Responsibilities

- **Task completion:** tasks assigned to team member are expected to be finished in time. If the task is not finished regarding to the due date, it will be the responsibility for the team member to report and clarify their problems in the task.

- **Discussion and Following the Decision:** the team member must follow the decision from team discussion. After team discussion, it will be the responsibilities for each individual to follow what have been discussed.

- **Effective communication:** When working on one's task that require the collaboration between members, these members will need to show a supportive attitude toward each other and when working online, one should be available when the others need.

- **Meeting Attendance:** joining team meeting to catch up each other progress and solving problem together is the most important responsibility of a team member. If one cannot show up during the meeting, then they have responsibility to check the meeting notes to catch up.

3.2.2. Roles activities

Team leader

- Organize the work of other team members
- Setup team meeting and ensure that everyone share the same idea
- Make decision
- Manage documentations
- Tracking worklog of other team members
- Resolving problems of other
- Communicate with client

AI engineer/Specialist/Researcher/Inspector:

- Provide insight about how the AI model work
- Research state-of-the-art techniques
- Implement the AI solution
- Validate the result of the AI model
- Support software team to integrate the AI model into the system

Software engineer/Specialist/Researcher/Inspector:

- Provide insight about how the software product will be created
- Research about the architecture of a software system
- Implement the software system with frontend and backend
- Test the functionality of the system

Devops:

- Ensure the developing environment is working
- Deploy the work of other to production

Ethical Inspector/ Policy researcher:

- Research about policies restriction of the project
- Ensure the work of other members is legal

Git organizer:

- Manage pull requests and resolve conflicts
- Manage the README

Result organizer:

- Gather the testing result
- Analyze the result to make decision for the next phase

Quality assurance:

- Ensure that everything is following the standard
- Perform corrective actions

CHAPTER 4: DOCUMENTATION

4.1. Software Documents

4.1.1. SQAP

The project SQAP of Compliance Camera will define quality assurance to maintain consistency and reliability of the final product. Moreover, it encompasses all project lifecycle phases, including requirements gathering, development and maintainance. This document will take into account code reviews, testing, documentation, and configuration management activities. The process will include the following criterias:

Objectives:

- Highlight the main quality assurance process
- Clarify the standard that the team will follow
- Specify how to document and product will be reviewed and audited
- Define the testing strategies to ensure the quality and standard is qualified
- Create a pipeline for problem reporting and corrective action
- Manage risks

4.1.2. SRS

The Software requirements specification will descript the product behaviours based on the client requirements and compliances reivew. The SRS will be based on the IEEE 830 standard and will be adjusted to suit with the project.

A general out of the document

1. **Introduction**
   a) Purpose:
      - Explain the purpose of the project
   b) Scope
      - Provide the scope of the project, what is inside the scope, what is outside
   c) Definitions, acronyms and abbreviations
      - Explain the shortcuts used in the documentation
   d) References
      - References to other documentation
   e) Overview
      - A general introduction to the project, how it will be progress and how it solves the customer problem
2. **Overall decription**
   a) Product perspective
      - Explain how to product will work with other tools integrated into the system
   b) Product functions
      - A list of function for the product
   c) User characteristics
      - User analysis to highlight the potential behaviour of customer
   d) Constraints
      - What the project should focus on, and cannot go beyond
   e) Assumption and dependencies
      - A list of assumption about the customer, the client and the product
3. **Specific requirements**

- This section contains detail analysis of the customer requirements => how can they interact with the system based on use case

4.1.3. Project Plan

1. Project Introduction: Introduce the project

2. Project objective: A brief information about the project objectives

3. Project Preparation: Information about team structure, project environment and skill requirements

4. Project Organization: How the project will be structured

5. Project Schedule: A detail timeline for the project

6. Budget: Information on how much the project cost and how much can the team afford

4.1.4. Module Plan

A Module plan is the plan to build a single module during the software development life cycle.

A module plan must contain:

- Module scope:
  + Define the part of the project requirements that the module will address
  + Highlight what is not inside the scope of the module
- Design solution for the module:
  + The architecture of the module must be included in the module plan
  + Parts of the module must be specified and listed in the architecture
- Work division:
  + Frontend Development: Toan
  + Backend Development: Duc Anh
  + AI engineer: Cuong Nhat
  + Quality Assurance: Khang Minh
- Timeline of artifacts and testing: A module usually develop
  + A module plan specifies the milestone fuctions and timeline to implement the module
  + Each module will be developed in 4 weeks following the workflow specify in project plan

4.1.5. Self-assessment reports

This document contains the following sections:

- Summary: a reflection on the contribution to the project, focus on the amount of work, mistake made and how the author has grown during the project

- Work complete: description of the tasks done during the project

- Mistakes made: highlight the mistakes, also including the lesson learned from the mistake and how to improve

- Knowledge gained: specify the skills and the knowledge gain during the project

- Evidence: the evidence of how an individual cóntribution to the project

4.1.6. Audit report

An audit report is a report based on the standard that the team is following. During the audit, if there is any fraud or something is not following the standard, it will be documented with recommeded corrective actions.

Areas of fraud:

- The audit report should include detail information about the areas of fraud, specifying the lines and document containing the fraud.
- If the fraud is lined within the code, the auditor will need highlight the compliance that the code has violated

Corrective actions:

- Hot fix for the fraud areas
- Sending reminders to the authors and ensure that they know what they are doing wrong
- Provide open audit sessions to identify and working with hard-to-fix frauds.

4.2. Management Documents

4.2.1. Meeting Agendas

The meetings agendas will cover the information about what will be discussed in the meeting. The content will require each team member to at least has some contribution to the documentation and request topic about the work to the agenda. Every topic in the agenda should be properly scheduled and team member should prepare slide and information about their topic. The information about the agendas will be uploaded in Confluence and must following the Confluence template.

4.2.2. Meeting Minutes

The meetings minutes will contain information about discussed topic during the meetings. This documentation must be written in the Confluence template. A meeting minutes should contains information about the Topics presented, the information about it, and the decision that the team has made based on the information and the tasks following with it.

CHAPTER 5. STANDARDS, PRACTICES, CONVENTIONS AND METRICS

5.1. Purpose

To ensure the project meets customer requirements, metrics and standardization will be required. This section will describe the standards for technical documentation and code structure, how can these standards be implemented with examples.

This section will include procedure and guideline that the research and development team must follow to ensure the output of the project and will be thoroughly reviewed during the development process. Moreover, it is worth mentioning that we will use Eslint as our code formatter during the development process.

These standards act as the foundation for verifying and validating the project's deliverables.

5.2. Standard

5.2.1. Coding Standard

- Files should be short, clear, and concise.
- If a file is too long, review the code logic and split it into smaller files, group them in a folder, and create a single file named [folder_name].[file_name] to interface with other modules (exceptions like index.ts or .js to avoid writing the full import file name).
- One class per file.
- Conventions:
    - Use snake_case for variables and module names.
    - Use UPPER_SNAKE_CASE for constants and static values outside all scopes (e.g., const PI = 3.14).
    - Use PascalCase for classes, traits, interfaces, types, enums, etc.
    - Use kebab-case for class IDs in CSS.
    - Prefix private variables in a class with _.
    - Variable names starting with "get" should always return a value.
    - Variable names starting with "set" should always set a value.
    - Code must have type hint and defensive programming style to prevent issues
    - Summary: Variable names should be self-explanatory about their purpose.
- Code Style:
    - Use 4 spaces for indentation.
    - Prefer composition over inheritance and interfaces over class inheritance.
    - Avoid excessive class inheritance (e.g., great-grandparent class -> grandparent class -> parent class -> current class); review the code logic in such cases.
    - No Exceptions (if absolutely necessary, add a comment warning).

5.2.2. Structure Standard

Frontend and backend is separated and follow this structure

- src: source code.
- doc: documentation.
- Inline short/temporary comments with code, but move lengthy comments to doc.
- Documentation should mirror the code structure (e.g., src/component/module1/main/Button.js has doc/component/module1/main/Button.js.md).
- README.md
- LICENSE.md (unsure about enterprise license writing).
- ARCHITECTURE.md: crucial file describing the overall frontend project structure.

5.2.3. Document standard

The document in this project must follow these standards:

Headings:

- Use headings to structure content (e.g., H1 for main titles, H2 for sections, H3 for subsections).
- Avoid using more than three levels of headings to maintain readability.

Text:

- Use clear and concise language.
- Font size: 15px
- Font: Arial
- Use bullet points or numbered lists for enumerations.
- The important keywords should be highlighted

Links and References:

- Use internal Confluence links to refer to other documents or sections.
- Provide external links where necessary and ensure they are functional.
- Include a references section at the end of documents for cited sources.

Images and Media:

- Include relevant images, diagrams, and other media to enhance understanding.
- Ensure all media is clear and appropriately labeled.
- Use Confluence’s image and file handling capabilities to manage media.

5.3. Practices

In order to maintain high quality standard throughout the duration of the project, the following practices are expected to be enforced

5.3.1. Communications

Client

- Communication with client will be done by the team leader
- Should a member unable to contribute, arrangements will be made to ensure no delay
- Client meeting will be held weekly with email being the primary method of communication

Supervisor

- Communication with supervisor will be done by the team leader
- The supervisor will attend one weekly meeting at the team's request
- The primary communication is email

Team

- Team meeting will be held at least twice a week, face-to-face is preferable after weekly supervisor meeting
- Email should be the main channel of communication. Member must check email frequently and must response as soon as possible

5.3.2. Meetings

- Weekly team meetings will be held with a fixed schedule along with the standard length of 30-60 minutes, full attendance is expected.
- Additional meetings beside regular scheduled meeting do not require full attendance
- Meeting with clients will be held weekly with a fixed schedule. This meeting requires the team to prepare adequately beforehand to ensure no problem will arise.
- All meetings, including additional meetings must have notes be taken and share among all members

5.3.3. Worklogs

- The team leader will monitor and remind other members to keeps up with the work logs as well as provide support if needed

CHAPTER 6: REVIEW AND AUDIT

6.1. Purpose

The review process is necessary to verify and validate the progress of the project and highlight the possible defect as well as supervise the implementation of standards. To do this, we will pay attention to requirements review, design review, code review, and AI review. Additionally, we will also perform code audit and document audit to ensure that everything is follow the standard.

6.2 Review/Audit list

6.2.1 Reviews

Reviews will occur throughout the project life cycle to identify potential defect within the work of everyone in the team.

Formal review process:

A formal review will be held after each sprint to validate the work of every team member. In this project, we will use peer review as the primary method during the formal review meeting. The process of the formal review will be descibed as following:

- Declare the review's objective: as listed above, we will focus on 4 sides of the project during the review including requirements, design, code and AI.
- Requirements review: each team member will present their work in the sprint and explain how their work is aligned with the requirements of the client.
- Design review: Toan and Duc Anh will be the main reviewer in this part, everyone will explain the structure of their code base while Toan and Duc Anh verify it.
- Code review: software members will review each other's code, same for AI.
- AI review: Hoang and Nhat will be the main reviewer in this part, both will verify the result and the understanding about the AI model in action.
- Implementation and decision: after the above discussion, the team will start to hot fix these problems. If the problem is too complicated, then it will become an issue and will be consider as a task with detail assignment to handle.
- Final: the team leader will review the process and decision made during the discussion and adjust the plan of the next sprint.

Informal review process:

Code

To ensure the project is running smoothly while the quality of code is maintained, flexibility in review activities is required. For this reason, regular review should be conducted imformally in 2 ways:

- Peer review: After everyday, software members should check the overall work of the team to identify any bug. Also, AI members should also review the implemented algorithms to verify if the implementation is correct.
- Client review: During the process, client should also have access to the product and give feedback back to the team for further improvement.

Meeting

In order to improve the quality of each meeting and optimize the time used for meeting, a review process for meeting documentation after each meeting is inevitable.

- This revision will focus on the time used for the meeting and how was the time used.
- This process will also record the impact of each session to the overall conclusion of the meeting to make final adjustment to the meeting plan.
- A review of the meeting agenda will also be reviewed to adjust the arrangement of discussion topics to enhance the efficiency of the meeting (e.g. explaination of ChatGPT should come before any other ChatGPT topic,…)

Management Document

To ensure that documents are readable and standardised, Khang Minh as document organizer will pay attention to the presentation and content of each document and ensure that they are in good form and shape when showing to the client. The review process can be done as following:

- The document organizer will scan through the document and highlight violated parts of the document.
- Form an issue to report the problem back to the author.

6.2.2. Audit

Coding Practices Audit

The code audit activities will be conducted by the software team who will scan through the code, find problem existing inside the code and fix it, ensure that every code block following the standard specified.

Git Audit

Github page of the will also need to be standardised. The documentation along with the code will be supported and need an audit before and after releasing.

CHAPTER 7: TESTING

To ensure the quality of the product before releasing it and handing it to the client, testing will be required. In this section, we will define the testing process, how can it be carried out.

Overall, there are 3 main tests that we will focus on. The first test is the unit test written by the software team. This test will be served as a test for code functionality and ensure that all main functions are running as expected. However, when it comes to dealing with customers behaviours and requirements fullfilness, the unit test alone will be not enough to evaluate the product. Therefore, we will run field tests to further understand the functionality of our product and improve it. We will also run usability/functionality test with random users from different user groups to evaluate their behaviour when using the system.

After the product passed the above 3 tests, it will be given to the client for the final feedback. The deliverables will include a feedback form, the product as well as the test record.

7.1. Requirements Testing

To ensure the product is usable and fit with the client demand, the requirements of the project will be thoroughly reviewed and tested. The client will also receive the product and testing from the team to verify if the product is suitable for them or not during the development lifecycle.

7.2. Use case generation

In order to have sufficient number of use cases that cover every aspect and possible fraud within the system, a use case generation plan must be specified. The process will include brainstorm the possible problem, gather user and system behaviours during prior tests, design a list of test case, verify these test case with client, and then bring these use cases into action.

7.3. Installation and User Documentation Generation

After all the tests are done, we will move to the installation process and hand all the documentations including user guideline and product’s design and maintainance documentation to the customer.

The installation process will include installing the cameras into the shop environment, evaluating the performance to ensure it is running for the last time.

After this process, the following documentation will be given to the customer:

- A user guideline on how to use the product with all features and use cases covered
- An architecture overview document of the product
- A Maintaince guideline of the product

CHAPTER 8: PROBLEM REPORTING AND CORRECTIVE ACTION

8.1. Personel

If there are problem occurs with personel, the person will need to report this to the team leader. The team leader will gather information about the problem and then define a proper corrective action:

- If the problem is related to the task (e.g. the task is hard,…) then counselling will be used to support that person to finish the task
- If the problem is related to the person in charge (e.g. busy,…) then reassignment will be used to ensure the progress of the task

8.2. Work

8.2.1 Project major timeline

Our project will be divided in 2 sub-projects including a web-based reporting system and an AI system for compliance checking.

The project will have 4 versions, each shall be a major milestone for development. The versions are:

- The prototype A version: this version will primarily focus on the working of the AI model and ensure that the model run with good accuracy for action recognition tasks.
- The prototype B version: this version will primarily focus on the working of the AI model for other tasks beside action recognition and all the models shall be working on a tested camera system and a training system for the cameras should also be done.
- The prototype C version: this final version will require the web-based reporting system running along with the AI model.
- The final: this version will be a well working version that will run perfectly on the shop environment.

With the above proposed milestone, we can also see that both AI and software sub-project will have 2 main phases.

To implement the spiral process model, when logging task and problem to jira, the following postfix should be used to give more information about where to problem come from:

- “dsg” for design related problems
- “dev” for development related problems
- “test” for testing related problems.

8.2.2. Stage-dependent tasks

Purpose

Stage-dependent tasks and issues must be clearly identified and documented to ensure they are properly managed within their respective sub-projects and phases.

Policy

All stage-dependent tasks must:

1. Indicate the sub-project to which they belong.
2. Specify the version of the sub-project.
3. Clarify the phase (e.g., design, development, testing).
4. Provide a detailed description of the task.

Format

Use the following format to document each stage-dependent task:

[Sub-project Identification] [Version] [Phase] - Task Description

Examples

- **[AI Training] v1.0 dsg**: This issue belongs to the design phase of AI Training, first release.
    - **Task**: Define training pipeline
    - **Description**: Create a training pipeline for the AI model

8.2.3. Crossed-states tasks

Purpose

Crossed-states tasks and issues must indicate the parent project's version to maintain consistency and traceability across different project states.

Policy

All crossed-states tasks must:

1. Indicate the parent project to which they belong.
2. Specify the version of the parent project.
3. Provide a detailed description of the task.

Format

Use the following format to document each crossed-states task:

[Parent Project] [Version] - Task Description

Examples

- **[Employee Compliance Checker] v1.0**: This documentation task falls under Employee Compliance Checker project version 1.0.
    - **Task**: specify the training pipeline
    - **Description**: design a training pipeline for human detection

8.2.4. Task creation

Purpose: To ensure that tasks is created in a correct way and be documented

Policy:

To create a task, the follow procedure must follow:

- The team leader must base on the discussion from other team members during meeting to form a new task
- The task will also need to be double check to prevent duplication
- Tasks can also be created based on the reports of bug on github issues

8.2.5. Task assignment

Purpose: To ensure that tasks is assigned to someone and make sure they have responsibility on it

Policy:

When a task is created a person will need to be assigned to it within 24 hours

The assignee should be an appropriate person, if the tasks are not well assigned, the team will have a meeting for rearrangement.

8.2.6. Task life

Purpose: To ensure that tasks is done and tracked

Policy:

- A task's life should be no longer than 2 weeks, if the task surpass this threshold, it will be investigate for appropriate solution

- After the task expected outcome is achieved, it will be marked as done

8.2.7. Issue Categories

Categories can be updated to adapt to the project's development, the following are most up to date:

• Administration

• Audit

• Client Liaison

• Coding - Prototype

• Documentation - AD

• Documentation - General

• Documentation - PP

• Documentation - SAR

• Documentation - SD

• Documentation - SQAP

• Documentation - SRS

• Lecture

• Meeting - Client

• Meeting - Other

• Meeting - Weekly

• Presentation Preparation

• Research - Coding

• Research - Documentation

• Review

• Confluence Management

• Requirements - Module X

• Design - Module X

• Coding - Module X

• Testing - Module X

CHAPTER 9: TOOLS AND METHODOLOGIES

9.1. Tools

9.1.1. Issue Tracking

We will manage issues using GitHub issues, allowing us to create, prioritize, and track the progress of each issue. Team members can assign issues, comment, and update their status, ensuring efficient and organized issue resolution.

9.1.2. Git

All team members are required to use Git for managing changes, commits, and updates. Additionally, we will set up GitHub Actions to automate unit testing.

9.1.3. Text Editor (Preferred: Visual Studio Code)

Any text editor can be used for this project. The project will be run and tested using CLI commands such as npm. Visual Studio Code is recommended for its extensive features.

9.1.4. Confluence and Jira

Confluence will be used for documentation and team collaboration, serving as a centralized hub for project information. Jira will track task progress, including tasks in progress, completed, or pending, and assign them to team members.

9.1.5. Microsoft Teams

If meetings via Microsoft Teams are needed, all team members must install Microsoft Teams and have access to a microphone and speakers.

9.2. Design Methodology

Spiral Lifecycle Model

- The Spiral Lifecycle Model’s core feature is its iterative approach, making it ideal for our project's modular structure.
- Each iteration and phase will be carefully planned and integrated into the overall project timeline to ensure systematic progress.
- Specific timelines for each module will be outlined in the module-specific plans.
- Each iteration will be treated as a distinct project phase, with a planning phase that includes key elements:
    - **Requirements**: Defining the specific requirements for the current iteration to align with the project's evolving needs.
    - **Design**: Developing a detailed plan for the iteration, including scheduling, resource allocation, and proactive risk management.

- **Development/Testing**: This crucial phase involves building (programming and prototyping) and thoroughly testing the software.

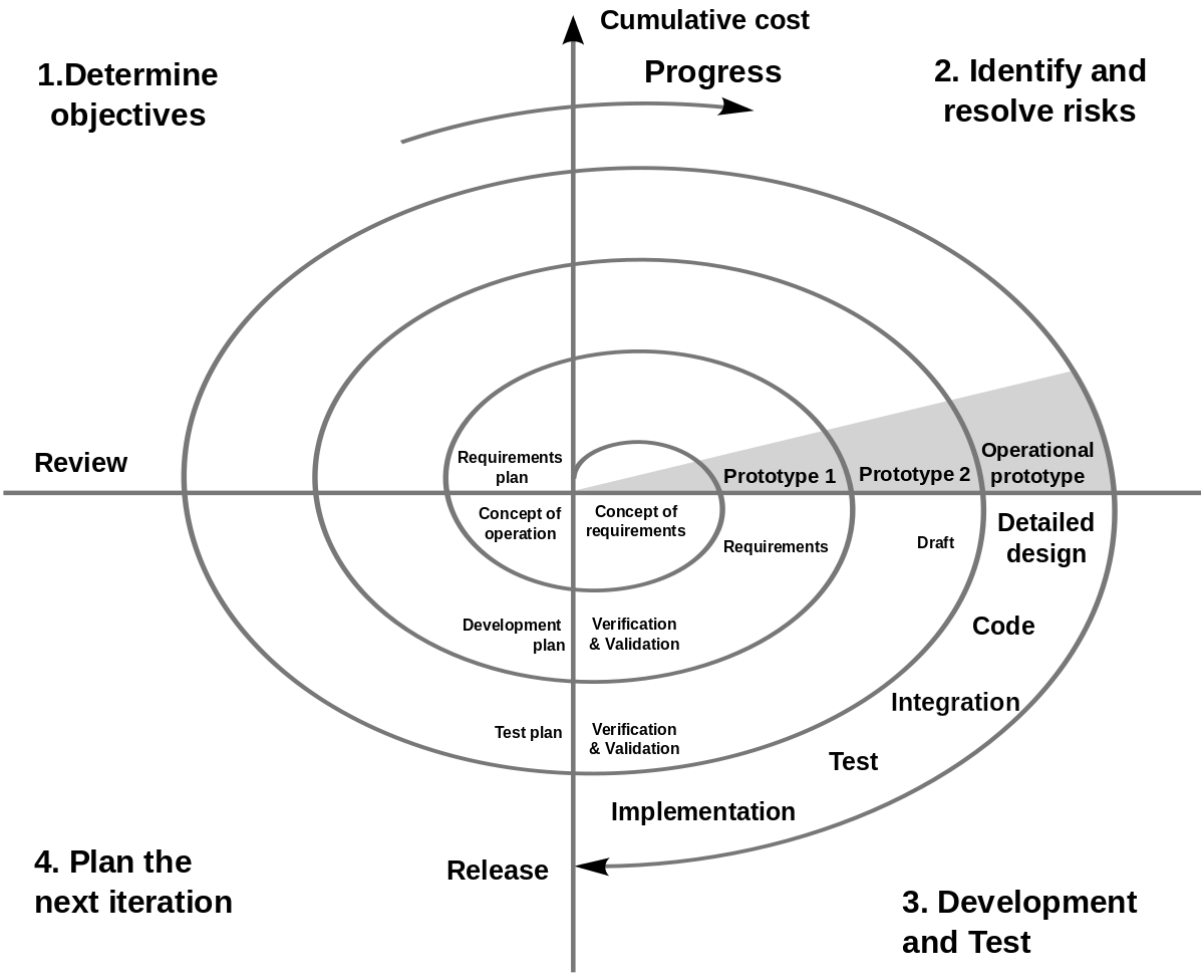

Figure 5. Spiral Methodology

CHAPTER 10: RECORDS COLLECTION, MAINTENANCE AND RETENTION

This section outlines the policies and procedures for collecting, maintaining, and retaining records related to the System Quality Assurance Plan (SQAP) for the project on using AI cameras to check staff compliance in the shop. Proper records management is essential to ensure the integrity, traceability, and accessibility of information throughout the project lifecycle.

10.1. Records Collection

10.1.1. Types of Records

- **Project Documentation:** Proposals, plans, reports, meeting minutes, and correspondence.
- **Data Records:** Raw video data, labeled datasets, and processed data outputs.
- **Development Records:** Code repositories, version control logs, and development notes.
- **Testing Records:** Test plans, test cases, test results, and defect logs.
- **Compliance Records:** Document compliance rules, monitoring logs, and compliance reports.
- **Ethical and Privacy Records:** Consent forms, privacy impact assessments, and ethical review documents.

10.1.2 Collection Process

- **Data Acquisition:** Raw video data is collected from CCTV cameras in the shop environment.
- **Data Annotation:** Labeled datasets are created through manual and semi-automated annotation processes.
- **Documentation:** All project-related documents are systematically created and updated by the project manager. All documents are stored in the Confluence site of the team.
- **Version Control:** Code and development records are maintained using Git which is a very common version control system.

10.2. Records Maintenance

10.2.1 Storage and Organization

- **Digital Records:** All digital records are stored in a centralized, secure cloud storage solution with appropriate access controls.
- **Physical Records:** Any physical records are stored in a locked cabinet with restricted access to authorized personnel only.
- **File Organization:** Records are organized into clearly labeled folders and subfolders based on their type, date, and project phase.

10.2.2 Access Control

- **Permissions:** Access to records is granted based on roles and responsibilities. Only authorized personnel have access to sensitive or confidential information.
- **Authentication:** Secure authentication mechanisms are implemented to ensure that only authorized users can access digital records.

10.2.3 Backup and Recovery

- **Regular Backups:** Digital records are backed up daily to prevent data loss.
- **Recovery Plan:** A disaster recovery plan is in place to restore records in the event of data loss or corruption.

10.3. Records Retention

10.3.1 Retention Periods

- **Project Documentation:** Retained for the duration of the project and for five years after project completion.
- **Data Records:** Retained for the duration of the project and for five years after project completion or as required by regulatory guidelines.
- **Development Records:** Retained for the duration of the project and for three years after project completion.
- **Testing Records:** Retained for the duration of the project and for three years after project completion.
- **Compliance Records:** Retained for the duration of the project and for five years after project completion.
- **Ethical and Privacy Records:** Retained for the duration of the project and for five years after project completion or as per ethical guidelines.

10.3.2 Disposal of Records

- **Digital Records:** Secure deletion methods are used to ensure that digital records are irretrievably deleted after the retention period.
- **Physical Records:** Physical records are shredded or incinerated to ensure they cannot be reconstructed.

10.4. Monitoring and Review

- **Regular Audits:** Regular audits are conducted to ensure compliance with the records collection, maintenance, and retention policies.
- **Continuous Improvement:** Policies are reviewed annually and updated as necessary to incorporate improvements and changes in regulatory requirements.

CHAPTER 11: RISK MANAGEMENT

11.1. Purpose

The purpose of this part is to identify, assess, and manage risks associated with the project about using CCTV cameras to check the compliance of staff in the shop. Effective risk management is crucial to ensure the project meets its objectives, remains on schedule, and stays within budget. This plan provides a structured approach to handling risks, minimizing their impact, and ensuring the project's success.

11.2. Risk categorization

Risks are categorized based on their source and impact areas:

- **Technical Risks:** Risks related to the technical aspects of the project.
- **Management Risks:** Risks related to project management and coordination.
- **Client-Related Risks:** Risks related to client interactions and requirements.
- **Operational Risks:** Risks related to the day-to-day operations and deployment.
- **Compliance Risks:** Risks related to adherence to laws, regulations, and ethical standards.

11.3. Risks with respect to the Work to Be Done

11.3.1. Technical Risks

- **Model Accuracy:** Risk of the AI model not achieving the required accuracy for action recognition and object detection.
    - **Mitigation:** Implement robust testing, use diverse training datasets, and continuously refine models.
- **Data Quality:** Risk of poor quality or insufficient data for training and testing.
    - **Mitigation:** Ensure high-quality data collection and augmentation techniques.
- **Integration Issues:** Risk of difficulties in integrating AI models with the existing CCTV system.
    - **Mitigation:** Plan for integration early in the project and conduct thorough testing.

11.3.2. Operational Risks

- **System Downtime:** Risk of the AI system experiencing downtime, affecting compliance monitoring.
    - **Mitigation:** Implement redundancy and failover mechanisms, and maintain regular system checks.
- **Deployment Challenges:** Risk of issues during the deployment phase in the real shop environment.
    - **Mitigation:** Conduct pilot tests and prepare a detailed deployment plan.

11.2. Risks with Respect to the Management

11.4.1. Project Delays

- **Schedule Slippage:** Risk of not meeting project deadlines due to unforeseen issues.
    - **Mitigation:** Develop a detailed project timeline with buffer periods and regularly monitor progress.
- **Resource Availability:** Risk of key team members being unavailable or leaving the project.
    - **Mitigation:** Ensure cross-training of team members and maintain a resource contingency plan.

11.4.2. Budget Overruns

- **Cost Overruns:** Risk of project costs exceeding the allocated budget.
    - **Mitigation:** Monitor budget regularly, control expenditures, and secure additional funding if necessary.

11.4.3. Communication Failures

- **Information Gaps:** Risk of poor communication leading to misunderstandings and errors.
    - **Mitigation:** Establish clear communication channels and regular update meetings.

11.3. Risks with Respect to the Client

11.5.1. Requirements Changes

- **Scope Creep:** Risk of clients requesting additional features or changes beyond the initial scope.
    - **Mitigation:** Clearly define project scope and manage changes through a formal change control process.

11.3.2. Client Satisfaction

- **Expectation Misalignment:** Risk of the final product not meeting client expectations.
    - **Mitigation:** Engage clients regularly through demos and feedback sessions to ensure alignment.

11.3.3 Data Privacy Concerns

- **Privacy Violations:** Risk of non-compliance with privacy regulations, potentially leading to legal issues.
    - **Mitigation:** Implement strict data privacy protocols and obtain necessary consents.

11.4. Additional Considerations

11.4.1. Risk Assessment and Monitoring

- **Risk Register:** Maintain a risk register to document identified risks, their impact, likelihood, and mitigation strategies.
- **Regular Reviews:** Conduct regular risk review meetings to reassess risks and update mitigation plans as necessary.

11.4.2. Risk Mitigation Strategies

- **Avoidance:** Eliminate the risk by changing project plans.
- **Mitigation:** Reduce the impact or likelihood of the risk.
- **Acceptance:** Acknowledge the risk and plan for its potential impact.
- **Transfer:** Shift the risk to a third party (e.g., through insurance).

11.4.3. Contingency Planning

- Develop contingency plans for high-impact risks to ensure quick and effective responses if risks materialize.