**Vietnam General Confederation of Labor**

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**FINAL REPORT**

**SOFTWARE ENGINEERING**

*Instructor*: **Mr. PHẠM THÁI KỲ TRUNG**

*Student*: **PHAN THÀNH ĐẠT - 521H0218**

**VĂN CÔNG NGUYÊN PHONG - 521H0287**

*Year* **: 2022-2023**

**HO CHI MINH CITY, 2023**

Vietnam General Confederation of Labor

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**FINAL REPORT**

**SOFTWARE ENGINEERING**

*Instructor*: **Mr. PHẠM THÁI KỲ TRUNG**

*Student*: **Phan Thành Đạt - 521H0218**

**Văn Công Nguyên Phong - 521H0287**

*Year***:2022-2023**

**HO CHI MINH CITY, 2023**

ACKNOWLEDGEMENT

We want to give to Mr. Pham Thai Ky Trung sincere thanks, who enthusiastically taught and worked very hard to give us to have enough knowledge and skills to complete this report. And we want to give thanks for Tôn Đức Thắng University, when this place give me an opportunity to follow up with large exercises so that we can consolidate the knowledge learned through the lectures.

*Ho Chi Minh city, 7th May, 2023*

*Author*

*(Sign and write full name)*

*Phan Thành Đạt*

*Văn Công Nguyên Phong*

**THIS PROJECT WAS COMPLETED AT**

**TON DUC THANG UNIVERSIY**

I fully declare that this is my own project and is guided by Mr. Pham Thai Ky Trung; The research contents and results in this topic are honest and have not been published in any form before. The data in the tables for analysis, comments and evaluation are collected by the author himself from different sources, clearly stated in the reference section.

Besides that, the project also uses a number of comments, assessments as well as data from other authors, other agencies and organizations, with citations and source annotations.

**Should any frauds were found, I will take full responsibility for the content of my report.** Ton Duc Thang University is not related to copyright and copyright violations caused by me during the implementation process (if any).

*Ho Chi Minh city, 7th May, 2023*

*Author*

*(Sign and write full name)*

*Phan Thành Đạt*

*Văn Công Nguyên Phong*

CONFIRMATION AND ASSESSMENT SECTION

**Instructor confirmation section**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Ho Chi Minh May, 2023*

*(Sign and write full name)*

**Evaluation section for grading instructor**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Ho Chi Minh May 2023*

*(Sign and write full name)*

SUMMARY

Develop a software for A distributor of mobile phones needs software that can manage warehouse receipts, order processing, delivery notes, and generate reports on stock, sales, and revenue.

INDEX

[CHAPTER 1 – INTRODUCTION 7](#_Toc18241)

[1.1. Purpose and Scope 7](#_Toc22711)

[1.2. Product Overview (including capabilities, scenarios for using the product, etc.) 7](#_Toc183)

[1.3. Structure of the Document 8](#_Toc193)

[1.4 Terms, Acronyms, and Abbreviations 10](#_Toc12529)

[CHAPTER 2 – PROJECT MANAGEMENT PLAN 12](#_Toc1576)

[2.1. Project Organization 12](#_Toc12577)

[2.2. Lifecycle Model Used 12](#_Toc23692)

[2.3. Risk Analysis 12](#_Toc4274)

[2.4. Hardware and Software Resource Requirements 15](#_Toc24250)

[2.5. Deliverables and Schedule 16](#_Toc19497)

[2.6. Monitoring, Reporting, and Controlling Mechanisms 16](#_Toc30843)

[2.7. Professional Standards 17](#_Toc25618)

[2.8. Impact of the project on individuals and my team 18](#_Toc9300)

[CHAPTER 3 – REQUIREMENT SPECIFICATIONS 19](#_Toc7271)

[3.1 Stakeholders for the system 19](#_Toc980)

[3.2 Use case model 19](#_Toc12302)

[3.2.1. Graphic use case model 19](#_Toc20370)

[3.2.2. Textual Description for each use case 20](#_Toc24926)

[3.3 Functional requirements 23](#_Toc1057)

[3.4. Non-functional requirements 24](#_Toc22485)

[CHAPTER 4 – ARCHITECTURE 26](#_Toc7027)

[4.1. Architectural style(s) used 26](#_Toc31835)

[4.2. Architectural model 27](#_Toc24198)

[4.3. Technology, software, and hardware used 28](#_Toc20686)

[4.4. Rationale for your architectural style and model 29](#_Toc31058)

[CHAPTER 5 – DESIGN 31](#_Toc25760)

[5.1. Database design 31](#_Toc2897)

[5.2. Static model – class diagrams 31](#_Toc22491)

[5.3. Dynamic model – sequence diagrams 32](#_Toc280)

[5.4. Rationale for your detailed design model 35](#_Toc9922)

[5.5. Traceability from requirements to detailed design model 37](#_Toc10195)

[CHAPTER 6 – TEST PLAN 38](#_Toc3040)

[6.1. Requirements/specifications-based system level test cases 38](#_Toc12719)

[6.2. Traceability of test cases to use cases 38](#_Toc29520)

[6.3. Techniques used for test generation 39](#_Toc23151)

[6.4. Assessment of the goodness of your test suite. 39](#_Toc4479)

[CHAPTER 7 – DEMO 41](#_Toc5562)

[7.1. Database 41](#_Toc181)

[7.2. Source code 43](#_Toc15600)

[7.3. Testing 44](#_Toc9254)

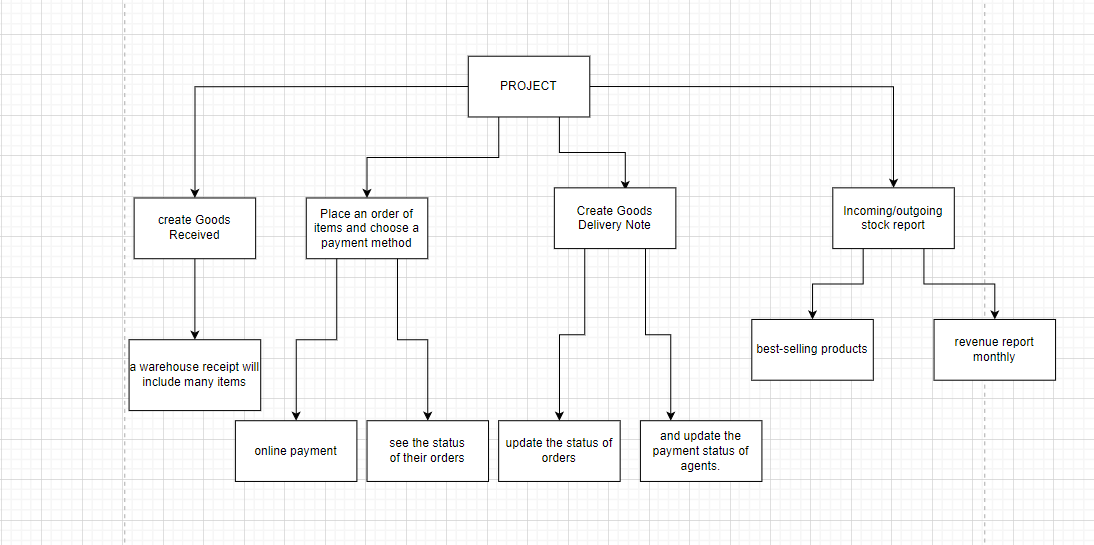
[CHAPTER 8 – ACKNOWLEDGMENT AND RUBRICS ASSESSMENT. 53](#_Toc8323)

CHAPTER 1 – INTRODUCTION

* 1. **Purpose and Scope**

The purpose of creating a software for management software for phone distributors is to automate and streamline the various processes involved in the distribution of phones. The software will help to manage inventory, sales, customer data, and other activities related to the distribution of phones.

The scope of the software includes features such as:



* 1. **Product Overview (including capabilities, scenarios for using the product, etc.)**

The capabilities of software depends on various factors such as its design, functionality, user interface, user experience, and user feedback. If the software is well-designed, easy to use, and meets the needs of the phone distributor, then its usability may be high.

Have many possible scenarios for using the produce:

- Phone distributors can use the software to manage their inventory, including tracking the stock levels of various phone models, managing orders from suppliers, and monitoring sales and stock movement.

- It can be used to create and manage marketing campaigns, track customer interactions and preferences, and generate reports on sales and marketing performance.

- The software can be used to manage the order fulfillment process, including tracking orders, managing returns, and ensuring timely delivery of products to customers.

* 1. **Structure of the Document**

1. Introduction

1.1. Purpose and Scope

1.2. Product Overview (including capabilities, scenarios for using the product, etc.)

1.3. Structure of the Document

1.4. Terms, Acronyms, and Abbreviations

2. Project Management Plan

2.1. Project Organization

2.2. Lifecycle Model Used

2.3. Risk Analysis

2.4. Hardware and Software Resource Requirements

2.5. Deliverables and Schedule

2.6. Monitoring, Reporting, and Controlling Mechanisms

2.7. Professional Standards

2.8. Evidence all the artifacts have been placed under configuration management

2.9. Impact of the project on individuals and organizations

3. Requirement Specifications

3.1. Stakeholders for the system

3.2. Use case model

3.2.1. Graphical use case model

3.2.2. Textual Description for each use case

3.3. Functional requirements

3.4. Non-functional requirements

4. Architecture

4.1. Architectural style(s) used

4.2. Architectural model

4.3. Technology, software, and hardware used

4.4. Rationale for your architectural style and model

5. Design

5.1. Database design

5.2. Static model – class diagrams

5.3. Dynamic model – sequence diagrams

5.4. Rationale for your detailed design model

5.5. Traceability from requirements to detailed design model

6. Test Plan

6.1. Requirements/specifications-based system level test cases

6.2. Traceability of test cases to use cases

6.3. Techniques used for test generation

6.4. Assessment of the goodness of your testsuite

7. Demo

7.1. Database

7.2. Source code

7.3. Testing

**1.4 Terms, Acronyms, and Abbreviations**

The terms, acronyms, and acronyms used in the phone distributor's software development process may vary depending on the specific project and technology stack.

Below we will list a few words that can be used in the project.

**API:** Application Programming Interface

**SDK:** Software Development Kit

**UI:** User Interface

**UX:** User Experience

**QA:** Quality Assurance

**MVP:** Minimum Viable Product

**CRUD:** Create, Read, Update, Delete

**ORM:** Object-Relational Mapping

**SQL:** Structured Query Language

**NoSQL:** Not only SQL

**REST:** Representational State Transfer

**JSON:** JavaScript Object Notation

**XML:** Extensible Markup Language

**HTML:** Hypertext Markup Language

**CSS:** Cascading Style Sheets

**JVM:** Java Virtual Machine

**IDE**: Integrated Development Environment

**VCS:** Version Control System

**CI/CD:** Continuous Integration/Continuous Deployment

**AWS:** Amazon Web Services

**DBMS:** Database Management System

**DNS:** Domain Name System

**SSL:** Secure Sockets Layer

**TLS:** Transport Layer Security

**HTTP:** Hypertext Transfer Protocol

**HTTPS:** Hypertext Transfer Protocol Secure

**IP:** Internet Protocol

**TCP:** Transmission Control Protocol

**UDP:** User Datagram Protocol

**OSI:** Open Systems Interconnection.

CHAPTER 2 – PROJECT MANAGEMENT PLAN

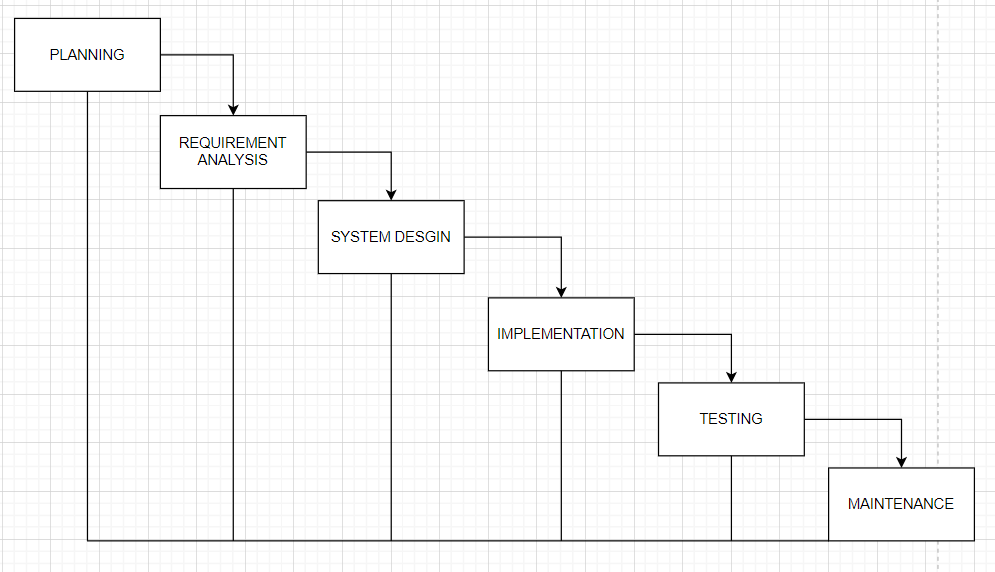
**2.1. Project Organization**

But the project only allows 2 people, so we will divide the work into 2 groups: Project Manager and Software Developers. Dat will be in charge of the Project Manager team and Software Developer will be in charge of Phong.

The reason that we divide like that, is because we realize what our ability is good at, Phong is good at coding while Dat is good at analysis and project management.

But there won't be only one person in charge of an area we will split and rotate to complete

**2.2. Lifecycle Model Used**

**2.3. Risk Analysis**

Possible project risks for software development at a phone distributor can vary depending on the specifics of the project.

Technology Risks: Technology risks can arise due to changes in technology, compatibility issues, or unanticipated technical difficulties. This can lead to project delays, increased costs, and unsatisfactory performance. The likelihood of technology risks depends on the complexity of the technology involved, the experience of the development team, and the nature of the project.

Risk Reduction Strategies: Conduct thorough research to ensure that the technology is suitable for the project, engage experienced development team members who have expertise in the technology being used, and conduct testing and prototyping to identify and address technical challenges early on.

Resource Risks: Resource risks can arise due to a lack of resources, inadequate staffing, or underestimating project requirements. This can lead to delays, poor quality, and increased costs. The likelihood of resource risks depends on the size and complexity of the project, the availability of resources, and the level of planning and estimation done at the beginning of the project.

Risk Reduction Strategies: Conduct thorough planning and estimation to identify resource requirements, engage skilled and experienced team members, and track progress against the project plan to identify and address resource constraints early on.

Scope Risks: Scope risks can arise due to changes in project requirements, lack of clarity on project objectives, or inadequate communication with stakeholders. This can lead to delays, increased costs, and unsatisfactory project outcomes. The likelihood of scope risks depends on the level of stakeholder involvement, the complexity of the project requirements, and the level of planning and estimation done at the beginning of the project.

Risk Reduction Strategies: Engage stakeholders throughout the project to ensure that project requirements are clear and understood, conduct regular project reviews to assess progress against the project plan and address any scope creep, and establish a clear change management process to manage changes to the project scope.

Financial Risks: Financial risks can arise due to budget constraints, unexpected costs, or inaccurate project cost estimates. This can lead to project delays, reduced quality, or project termination. The likelihood of financial risks depends on the accuracy of cost estimates, the complexity of the project, and the level of financial oversight and control.

Risk Reduction Strategies: Conduct thorough financial planning and estimation, track and monitor project expenses closely, engage experienced project managers and financial advisors, and establish clear financial controls and reporting mechanisms.

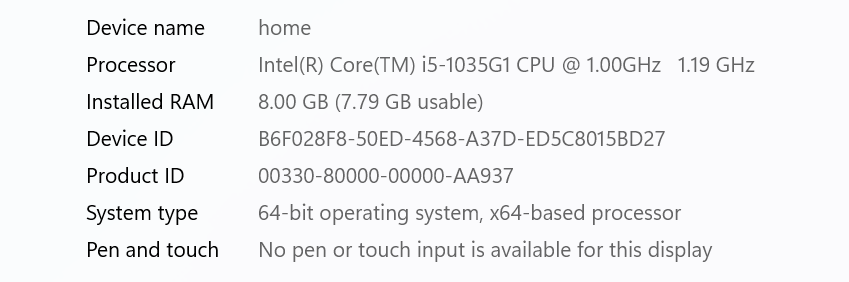
Legal Risks: Legal risks can arise due to intellectual property issues, licensing violations, or regulatory compliance issues. This can lead to legal disputes, project delays, and financial penalties. The likelihood of legal risks depends on the nature of the project, the regulatory environment, and the level of legal oversight and compliance.

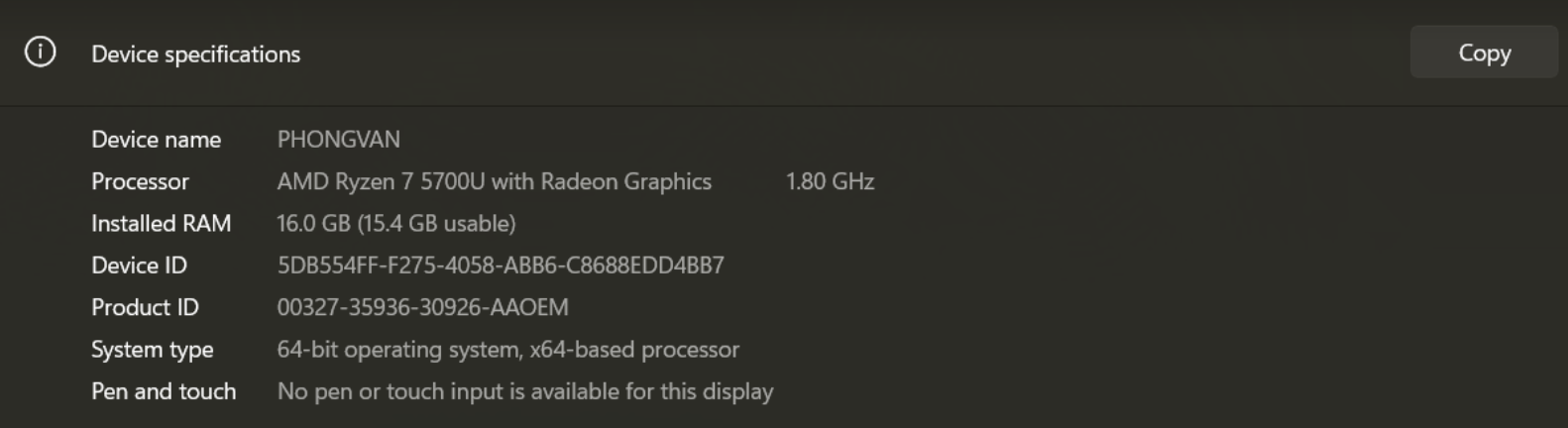
Risk Reduction Strategies: Conduct thorough legal due diligence to identify and address any legal risks early on, engage experienced legal counsel, establish clear compliance policies and procedures, and monitor legal developments that may impact the project.

**2.4. Hardware and Software Resource Requirements**

In order to effectively run Visual Studio, it is recommended to have a fast processor with a speed of at least 1.8 GHz, preferably quad-core or faster. You should have at least 2 GB of RAM, but 8 GB of RAM is highly recommended (at least 2.5 GB if running on a virtual machine). The amount of hard disk space required will vary depending on which features you install; typical installations require 20-50 GB of free space, though a minimum of 800MB and up to 210 GB may be necessary. To optimize performance, it is suggested that you install both Windows and Visual Studio on a solid state drive (SSD). Your video card should support a display resolution of at least 720p (1280 by 720), although Visual Studio is best used with a resolution of WXGA (1366 by 768) or higher.

**Hardware:**





**Software:**

- Microsoft SQL Server Management Studio 18

- Visual Studio 2020

- Visual Studio Code

**2.5. Deliverables and Schedule**

- The date of the project discussion will determine the delivery timetable. If the scheduled appointment time is surpassed, the customer will not receive the requested product. We'll assume total accountability.

- We will set a date for a demo report, outlining the project's completion plan, before delivery. Gather feedback, then edit it to meet software users' needs.

- by the delivery date anticipated. All tasks will be finished. The program that is delivered will fulfill the client's requests.

- End date: May 7, 2023

**2.6. Monitoring, Reporting, and Controlling Mechanisms**

Because this is a small project in one discipline, we only focus on the management reports that should be produced:

Project Status Report: This report provides an overview of the project's progress, including milestones achieved, project risks, and issues encountered. It should be produced on a regular basis, such as weekly or bi-weekly, to ensure all stakeholders are informed about the project's progress.

Risk and Issue Report: This report identifies and tracks risks and issues that could impact the project's success. It should be produced on a regular basis, such as weekly or bi-weekly, to ensure that any potential problems are identified early and addressed promptly.

The project monitoring and control mechanisms that can be used to manage the project about a phone distributor to agent include:

Project Schedule: A project schedule should be created and updated regularly to ensure that the project is progressing as planned. It should include milestones, deadlines, and critical path activities.

Performance Metrics: Performance metrics should be established to measure progress and identify any areas where improvements can be made. Examples of performance metrics include cost performance, schedule performance, and quality performance.

These reports and mechanisms should be do when it is started and do it throughout the project lifecycle to closure, to ensure that the project is delivered on time, and give the best software to the satisfaction of customer

**2.7. Professional Standards**

- Experience in using Windows Form Applications

- Experience in using Visual Studio

- Experience in using Microsoft SQL Server Management Studio 18

- Have SQL database knowledge

**2.8. Impact of the project on individuals and my team**

We also get certain influences from doing this project, both positive and negative.

The first positive is that we have developed ourselves, during the course of the project we were able to gain new skills, knowledge, and experience, which can enhance their personal and professional growth.

Second, we learn how to work in a team, and strengthen that when Working together towards a common goal can strengthen the bonds between team members and promote better collaboration and communication.

Besides, we also had some negative effects when doing this project.

Project stress and burnout can be challenging and demanding, leading to stress and burnout for my members.

Dissatisfied and lose motivation because of Failure to achieve project goals. Because of students, we have poor time balance. In some cases, working on a project that can span many hours can be productive, but our time management goes wrong, resulting in a negative impact on mental health and well-being, individual's physicality

CHAPTER 3 – REQUIREMENT SPECIFICATIONS

**3.1 Stakeholders for the system**

- Stakeholder for the system:

+ Software developers

+ Phone manufacturers

+ Agents

+ Customers

+ Distributors

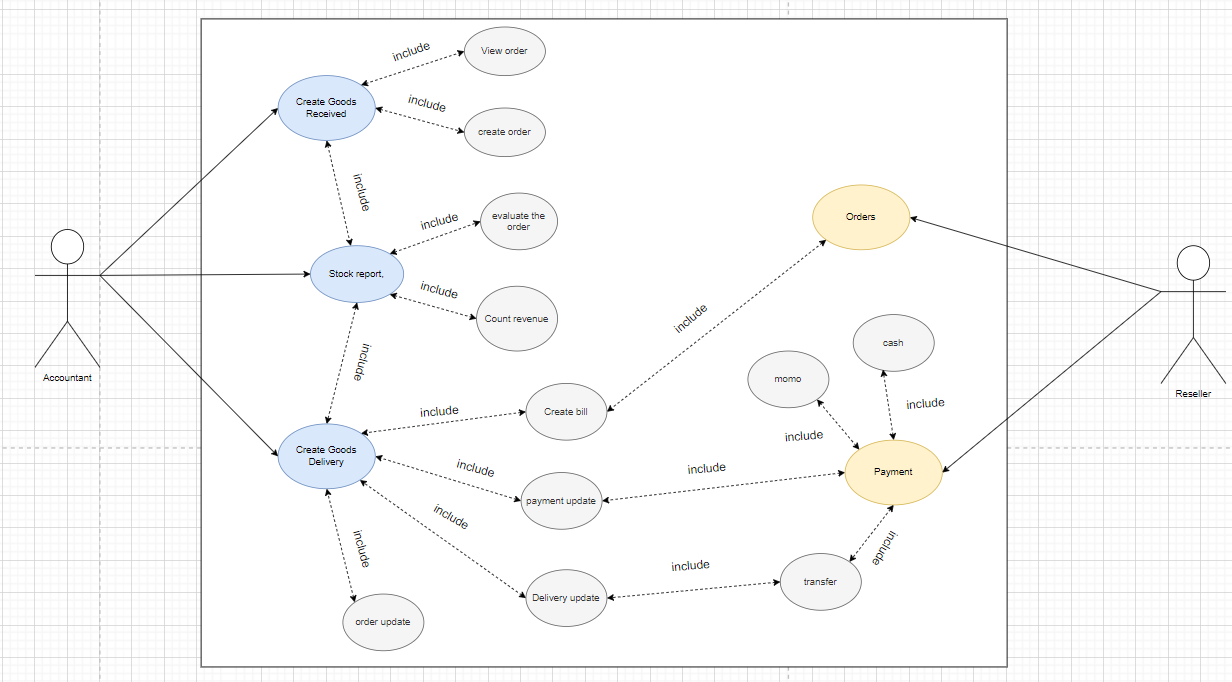
+ Payment processors

+ Legal and regulatory bodies

The system aims to achieve maximum overall satisfaction of all stakeholders by balancing their interests, even though it may not be possible to satisfy each individual party completely.

**3.2 Use case model**

**3.2.1. Graphic use case model**



**3.2.2. Textual Description for each use case**

**Use case for create Goods Received.**

Usecase name: Create Goods Received

Context: Directly at the store’s warehouse

Trigger: Select “Manage goods import” -> Select “Create purchase order”

Description: Usecase is used to creat receipt information when there is a new batch of goods and save it in the database.

Actor: Accounting

Related: Login

Stakeholders: Manage

Pre - Condition: Login with access right according to the position of accountant

Post - Condition: Update information in the database

Basic Flow:

+ Agent: Log in to the system. Select “Manage goods import” -> Select “Receipt of goods” The accouting enter the goods receipt information and click “ Save”.

+ System: Display the form enter the information of the receiptDatabase update system.

Result: Goods receipt created successfull

Exception: None

**Use case for Delivery bill**

Usecase name: Delivery bill

Context: Directly at the store’s warehouse

Trigger: Select “Manage goods export” -> Select “Create delivery note”

Description: Usecase is used to create delivery slip information when there is a shipment and save it in the database

Actor: Accounting

Related: Login

Stakeholders: Manage

Pre - Condition: Login with access right according to the position of accountant

Post - Condition: Update information in the database

Basic Flow:

+ Agent: Log in to the system. Select “Manage goods export” -> Select “Delivery note” The accouting enter the goods export information and click “ Save”

+ System: Display the form enter the information of the receip Database ,update system. Update order status and payment

Result: Goods receipt created successfull

Exception: None

**Use case for Stock Report**

Usecase name: Stock Report

Context: Manage operations directly on the system

Trigger: Select “ Report” -> Select the Report -> select “View Report”

Description: Usecase used to view Stock Report

Actor: Accounting

Related: Login

Pre - Condition:

-Login with access right according to the position of accountant

-There is at least one Stock Report vote in the list

Post - Condition: View information about Report

Basic Flow:

Agent:

- Login to the system -> Select “Report” -> Select the statistics sheet you want to see then select “View Report”

System:

- Check login permission -> Display “Report” interface -> Display information of the selected Report.

**Use case for Order**

Usecase name: Order Information

Context: Agency order at the company’s website

Trigger: Select “Order” -> Select “Create Order”

Description: Usecase is used for ordering

Actor: Agency/Reseller

Related: Login

Stakeholders: Company

Pre - Condition:

-Login to the system with agency access

-Create orderPost - Condition: View information about Report

Post – Condition: Update information in the database

Basic Flow:

Agent:

- Log in to the system

- Select “Orders” -> Select “Create Order”

- Enter order information

- Click “Save” -> Order

System:

- The system updates the database

- Display order on the company’s order management system

Result: Order successful

**Use case for Payment**

Usecase name: Payment

Context: Agency payment at the company’s website

Trigger: Select “Payment” -> Choose payment method

Description: Usecase is used for payment

Actor: Agency

Related: Login

Stakeholders : Accounting

Pre - Condition:

-Login to the system with agency access

-Agency order has been successfully placed

Post - Condition:

Update information in the database

Basic Flow:

- Agent:

- Log in to the system.

- Select “Payment” -> Select payment method

- Agency choose payment method and “pay”

- System:

- Show form paid

- Database update system.

Result: Payment successfull

**3.3 Functional requirements**

Agent registration: Agents need to register with their details such as name, contact number, email address and other required information.

Login: Registered agents should be able to log in to the system using their unique credentials.

Order Management: Agents should be able to place orders for phones, view details of orders, track order status and delivery updates.

Payment Gateway Integration: The software should integrate with payment gateways to allow agents to make payments online through various channels such as internet banking, credit/debit cards, e-wallets, etc.

Inventory management: The software should help phone distributors manage inventory levels, stock movement, and restocking of products.

Sales Reports: Generate sales reports based on different parameters such as date range, product type, agent, etc.

Customer support: Provide customer support to agents via chat, email or phone, to resolve their queries and issues.

Notification alerts: Send notifications to agents regarding new product launches, discounts, promotions, etc.

**3.4. Non-functional requirements**

Performance: The software should be designed to handle a large number of concurrent users and transactions, while ensuring fast performance and response time.

Security: The system should provide adequate security mechanisms to protect sensitive data such as customer information, payment details, and personal information of agents.

Scalability: The system should be scalable to cater for future growth of the business, in terms of increased number of agents or products sold.

Reliability: The system should be reliable and available at all times, with minimum downtime and disruption to the business operations.

Accessibility: The software should be accessible to agents from different locations, using devices such as smartphones, tablets or desktop computers, with an easy-to-use interface.

Usability: The software should be intuitive and user-friendly, with clear navigation, and minimal training required for agents to use it effectively.

Compatibility: The software should be compatible with a wide range of devices, operating systems, and browsers, to ensure maximum accessibility for agents.

Maintainability: The system should be easy to maintain and upgrade, with minimal impact on the business operations.

Compliance: The software should comply with relevant industry standards, regulations and legal requirements related to data privacy, security, and payment processing.

CHAPTER 4 – ARCHITECTURE

Product catalog: The software should display a catalog of available phone products including product description, price and availability.

**4.1. Architectural style(s) used**

For a distributor selling mobile phones products, an architectural style that could be suitable is the Microservices Architecture.

The system of a mobile phones distributor typically involves multiple business processes such as inventory management, order processing, payment processing, and customer service. Each of these processes could be managed by a separate microservice, allowing each service to be independently developed, deployed, and scaled as needed.

In a Microservices Architecture, the system is broken down into smaller, loosely coupled services that communicate with each other using lightweight protocols such as REST or messaging. This allows for better scalability and fault tolerance since each service can be scaled independently without affecting the others. Additionally, it allows for more flexibility in technology choices since each service can be developed and deployed using different technologies.

For example, a distributor's system could be composed of microservices such as an inventory management service, an order processing service, a payment processing service, and a customer service portal. These microservices would be responsible for managing their specific business process and would communicate with each other through APIs.

Overall, the Microservices Architecture provides a flexible, scalable, and efficient way to manage complex systems such as those used by mobile phone distributors.

**4.2. Architectural model**

The architectural model for a distributor selling mobile phones products using packages stereotyped as subsystems <<subsystem>> and no classes:

Inventory Management Subsystem: This subsystem would be responsible for managing the inventory of mobile phone products. It would include packages such as Product Catalog, Warehouse Management, and Supply Chain Management.

Order Processing Subsystem: This subsystem would be responsible for managing customer orders. It would include packages such as Order Placement, Order Tracking, and Order Fulfillment.

Payment Processing Subsystem: This subsystem would be responsible for processing payments for customer orders. It would include packages such as Payment Authorization, Payment Capture, and Payment Refunds.

Customer Service Subsystem: This subsystem would provide a user interface for customers to interact with the distributor's system. It would include packages such as Account Management, Order History, and Customer Support.

All these subsystems would communicate with each other through well-defined interfaces, which would allow them to exchange data and coordinate their actions. For example, the Order Processing Subsystem would communicate with the Inventory Management Subsystem to ensure that products are available before accepting an order. Similarly, the Payment Processing Subsystem would communicate with the Order Processing Subsystem to confirm that an order has been paid for before initiating the shipping process.

Overall, this architectural model provides a clear separation of concerns between different subsystems, allowing for better maintainability and scalability of the system. By using packages stereotyped as subsystems <<subsystem>>, it allows for a high-level view of the system's architecture without getting into the details of individual classes.

**4.3. Technology, software, and hardware used**

**Technology**

- SQL Server

- C#

- Git

- ASP.NET

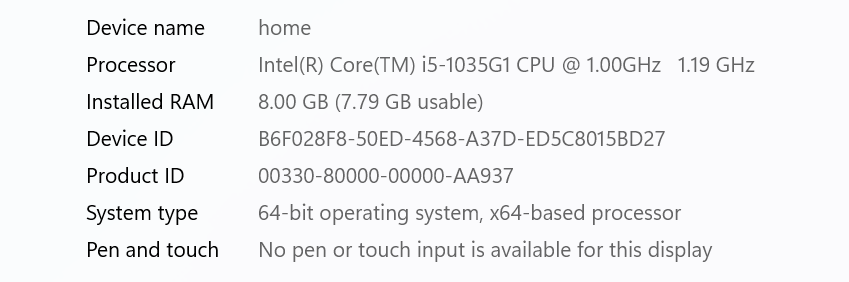
**Software:**

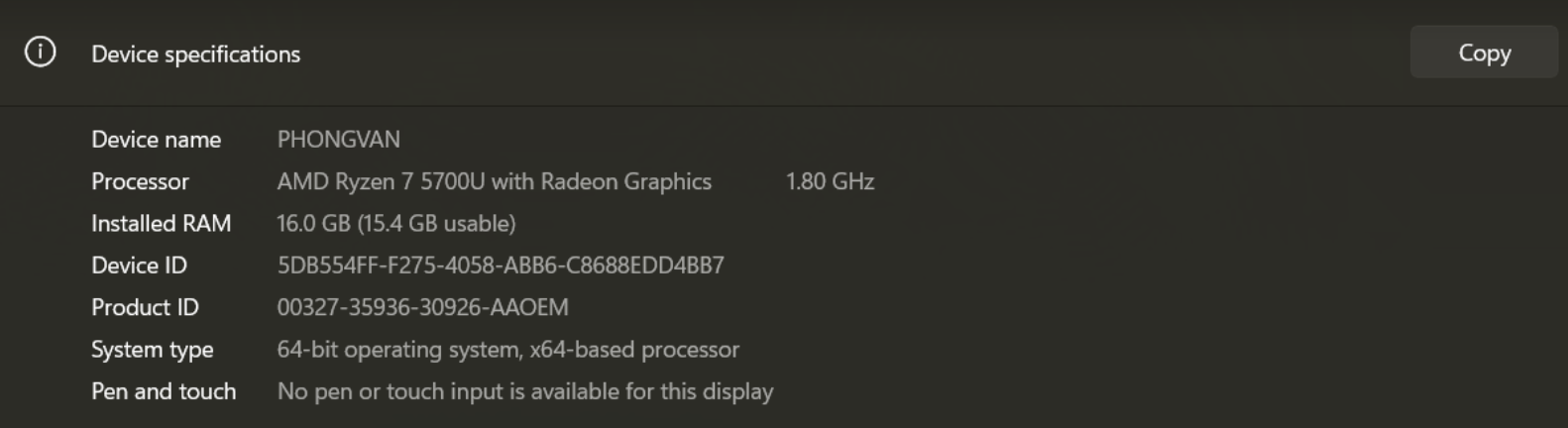
-Microsoft SQL Server Management Studio 18

-Visual Studio 2019

-Visual Studio Code

**Hardware:**





**4.4. Rationale for your architectural style and model**

The Microservices Architectural style is a suitable choice for the Distributor selling Mobile Phones products due to several reasons:

Scalability: A mobile phone distributor's system can be complex and involve multiple services that need to scale independently based on demand. The Microservices Architecture allows each service to be independently developed, deployed, and scaled as needed, making it easier to manage the scalability of the system.

Modularity: The Microservices Architecture provides clear boundaries between different services, making it easier to update and maintain the system. Each service can be independently developed, tested, and deployed, which can result in faster release cycles and fewer regressions.

Resilience: In a Microservices Architecture, if one service fails, it does not necessarily bring down the entire system. This is because each service is designed to be independent and have its own set of resources, allowing it to continue functioning even if other services fail.

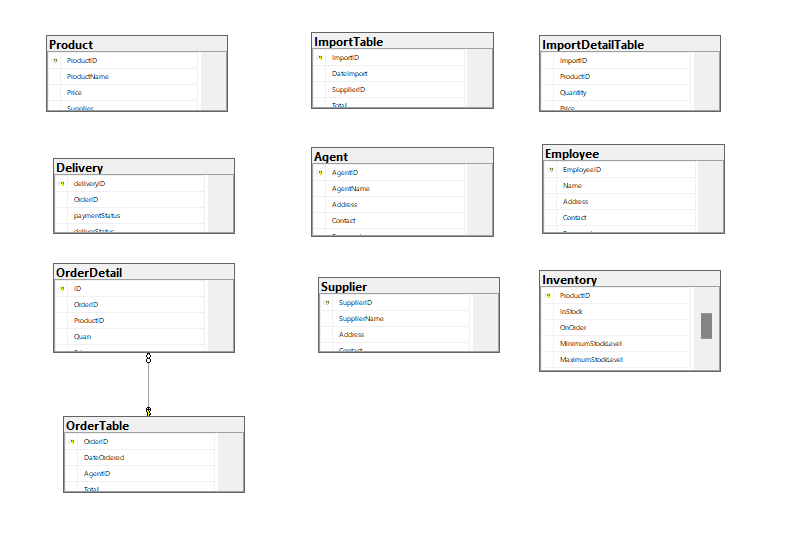
Technology diversity: In a Microservices Architecture, each service can be developed using a different technology stack, allowing developers to choose the most appropriate tool for the job.

The model of the Distributor selling Mobile Phones products using the Microservices Architecture would involve breaking down the system into multiple independent services such as Inventory Management, Order Processing, Payment Processing, and Customer Service Portal. Each of these services would be responsible for handling a specific aspect of the system and would communicate with each other through APIs. This would allow the system to be scalable, modular, and resilient to failure.

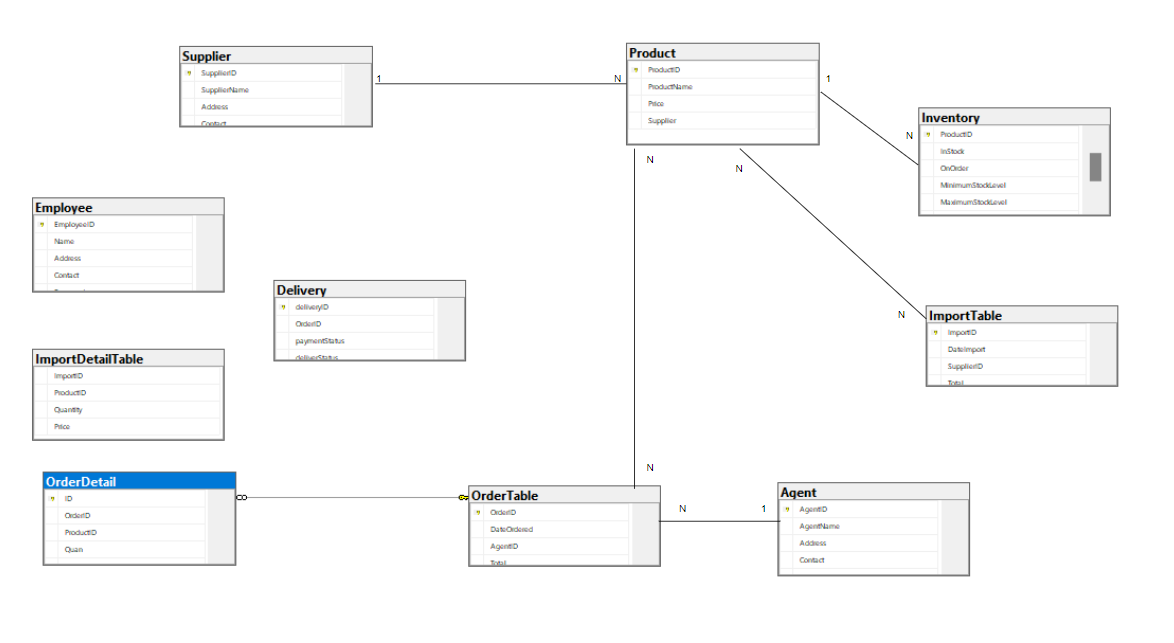
For example, the Inventory Management service could handle tasks such as updating stock levels and receiving new shipments. The Order Processing service could handle tasks such as accepting orders and coordinating shipping. The Payment Processing service could handle tasks such as payment authorization and capturing. The Customer Service Portal could provide an interface for customers to manage their accounts and access support.

CHAPTER 5 – DESIGN

**5.1. Database design**

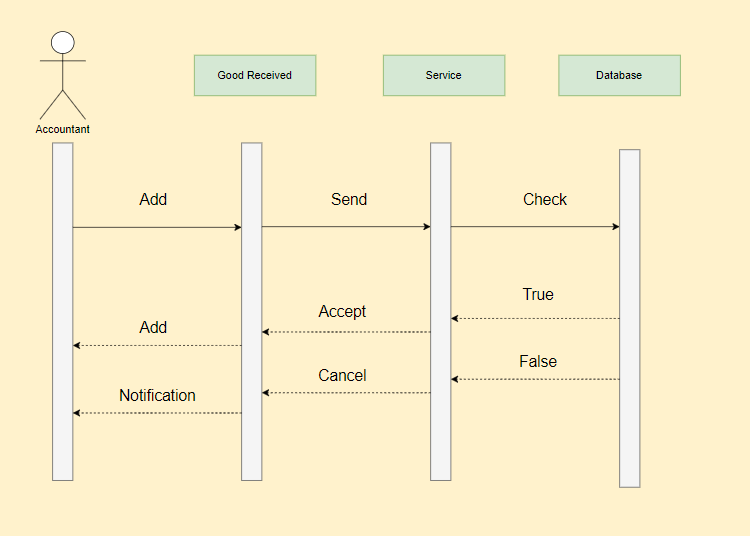


**5.2. Static model – class diagrams**

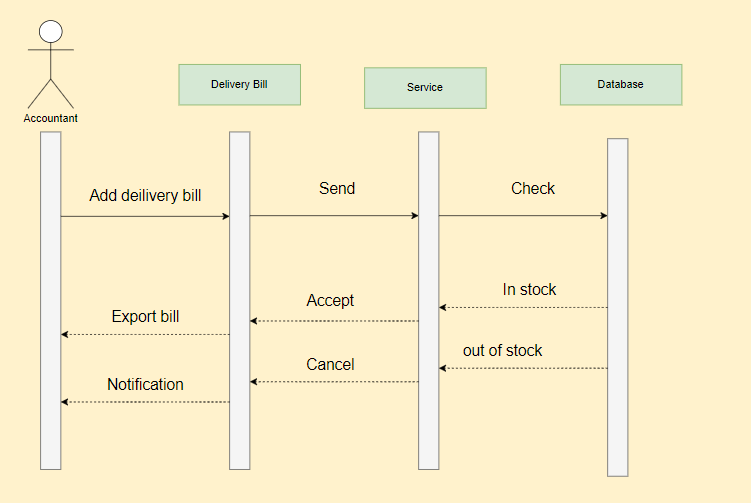


**5.3. Dynamic model – sequence diagrams**

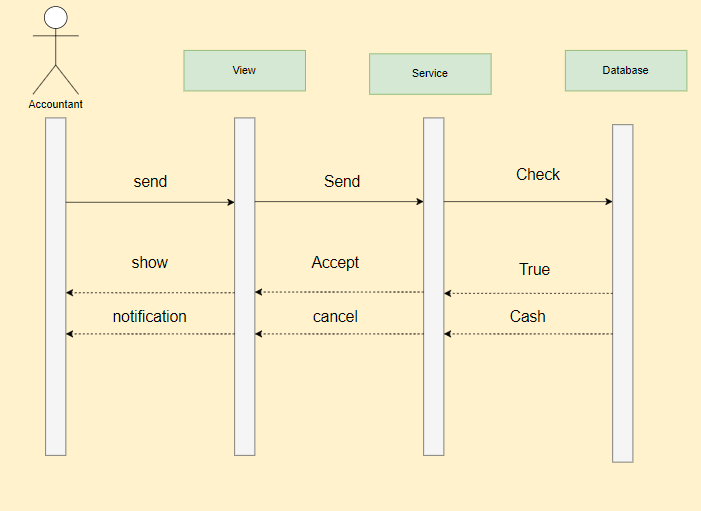
sequence diagrams of create goods receive:



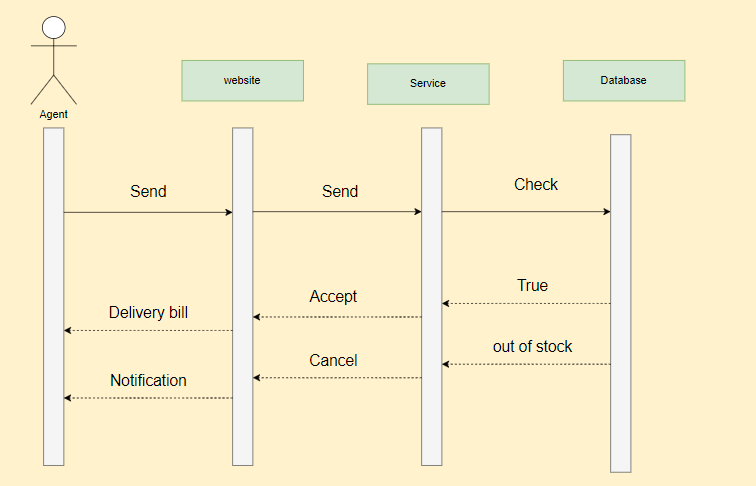
sequence diagrams of delivery:



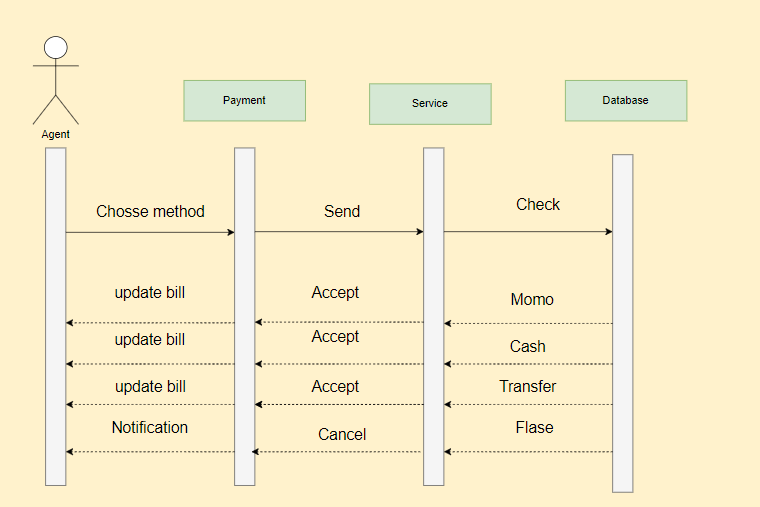
sequence diagrams of view report:



sequence diagrams of Order:



sequence diagrams of payment:



**5.4. Rationale for your detailed design model**

The detailed design model for a distributor selling mobile phones products to authorized resellers/agents would consist of several entities and their attributes. Here is the rationale for each entity in the model:

Supplier: This entity represents the Supplier who sells mobile phone products to authorized resellers/agents. The attributes of this entity may includeSupplierID (primary key), Name, Address, Contact details, and other relevant information.

Agent: This entity represents the authorized resellers/agents who buy mobile phone products from the distributor. The attributes of this entity may include AgentID (primary key), Name, Address, Contact details, and other relevant information.

Product: This entity represents the mobile phone products that the distributor sells to authorized resellers/agents. The attributes of this entity may include ProductID (primary key), ProductName, Price, Manufacturer, Model, and other relevant product information.

Inventory: This entity represents the inventory of mobile phone products that the distributor has available for sale. The attributes of this entity may include ProductID (foreign key from the Product table), InStock Quantity, OnOrder Quantity, MinimumStockLevel, MaximumStockLevel, ImportPrice, and other relevant information.

Order: This entity represents the orders that authorized resellers/agents place with the distributor for mobile phone products. The attributes of this entity may include OrderID (primary key), AgentID (foreign key from the Reseller/Agent table), DateOrdered, and Total Order Value.

OrderDetail: This entity represents the details of each order placed by an authorized reseller/agent. The attributes of this entity may include ID (primary key), OrderID (foreign key from the Order table), ProductID (foreign key from the Product table), Quantity Ordered, and Total Cost.

Delivery: This entity represents the shipments of mobile phone products made by the distributor to authorized resellers/agents. The attributes of this entity may include DeliveryID (primary key), OrderID (foreign key from the Order table), DateShipped, and Tracking Information.

By creating these entities and defining their relationships, we can create a database schema that enables us to track inventory levels, manage orders and shipments, and generate reports for the distributor's mobile phone sales to authorized resellers/agents.

**5.5. Traceability from requirements to detailed design model**

To establish a clear and organized system for managing their business, the company has implemented several requirements. Accountants must be able to generate Goods Received reports when goods are imported and Agents should be able to place orders and select payment methods from several options. Additionally, Accountants should be able to create Goods Delivery Notes, update order and payment statuses, and print delivery slips. Finally, they require a monthly report on incoming/outgoing stock and revenue. To achieve these objectives, the company has designed a database consisting of several tables such as accountant, item, receipt, detailReceipt, customer, orders, detailOrder, and deliveryBill.

CHAPTER 6 – TEST PLAN

**6.1. Requirements/specifications-based system level test cases**

- Add, edit, delete employee

- Add, edit, delete agent

- Add, edit, delete supplier

- Add, edit, delete product

- Add, edit, delete inventory

- Add, edit, delete orders and order details

- Add, edit, delete Importable

- Add, edit, delete delivery

- Add, edit, delete ImportDetailTable

**6.2. Traceability of test cases to use cases**

Use case for Goods Received:

- Add, edit, delete product

- Add, edit, delete employee

- Add, edit, delete importTable and ImportdetailsTable

Use case for Order:

- Add, edit, delete product

- Add, edit, delete agent

- Add, edit, delete orders and order details

Use case for Payment:

- Add, edit, delete product

- Add, edit, delete agent

- Add, edit, delete orders and order details

Use case for Delivery bill:

- Add, edit, delete product

- Add, edit, delete employeee

- Add, edit, delete agent

- Add, edit, delete orders and order details

- Add, edit, delete delivery

Use case for Stock Report:

- Add, edit, delete product

- Add, edit, delete importTable and ImportdetailsTable

- Add, edit, delete orders and order details

- Add, edit, delete delivery bill

**6.3. Techniques used for test generation**

For test generation of a distributor selling mobile phones products, techniques such as black-box testing and white-box testing can be used.

Black-box testing involves testing the system without any knowledge of its internal workings, focusing on the inputs and outputs. An example would be testing the functionality of the mobile phone by checking if the camera app opens and takes photos as expected.

On the other hand, white-box testing involves testing the system with knowledge of its internal workings, focusing on the code structure and logic. An example would be testing the mobile phone's operating system by checking if the code for the camera app is efficient and error-free.

**6.4. Assessment of the goodness of your test suite.**

Automated testing is an effective way of testing large portions of code and complex functions. Testers don't need to have knowledge of the actual code, and there is a clear distinction between the user's perspective and the developer's perspective. This type of testing can be applied at higher levels like system and acceptance tests, without requiring a deep understanding of the function's internal structure or how it may be accessed.

CHAPTER 7 – DEMO

**7.1. Database**

-- Create Employee table

CREATE TABLE Employee (

    EmployeeID VARCHAR(50) PRIMARY KEY,

    Name NVARCHAR(100) NOT NULL,

    Address NVARCHAR(200) NOT NULL,

    Contact VARCHAR(20) NOT NULL,

    Password VARCHAR(50) NOT NULL

);

-- Create Supplier table

CREATE TABLE Supplier (

    SupplierID VARCHAR(50) PRIMARY KEY,

    SupplierName VARCHAR(100) NOT NULL,

    Address NVARCHAR(200) NOT NULL,

    Contact VARCHAR(20) NOT NULL

);

-- Create Product table

CREATE TABLE Product (

    ProductID VARCHAR(50) PRIMARY KEY,

    ProductName VARCHAR(100) NOT NULL,

    Price INT NOT NULL,

    Supplier VARCHAR(50) NOT NULL,

);

-- Create Inventory table

CREATE TABLE Inventory (

    ProductID VARCHAR(50) PRIMARY KEY,

    InStock INT NOT NULL,

    OnOrder INT NOT NULL,

    MinimumStockLevel INT NOT NULL,

    MaximumStockLevel INT NOT NULL,

    ImportPrice int

);

CREATE TABLE Agent (

  AgentID VARCHAR(50) PRIMARY KEY,

  AgentName VARCHAR(50) NOT NULL,

  Address VARCHAR(100) NOT NULL,

  Contact VARCHAR(50) NOT NULL,

  Password VARCHAR(50) NOT NULL

);

CREATE TABLE OrderTable (

    OrderID varchar(50) PRIMARY KEY,

    DateOrdered datetime NOT NULL,

    AgentID varchar(50) NOT NULL,

    Total decimal(10,2) NULL

);

CREATE TABLE OrderDetail (

    ID int IDENTITY(1,1) PRIMARY KEY,

    OrderID varchar(50) NOT NULL REFERENCES OrderTable(OrderID),

    ProductID varchar(50) NOT NULL,

    Quan int NOT NULL,

    Price decimal(10, 2) NOT NULL

);

CREATE TABLE ImportTable (

    ImportID varchar(20) NOT NULL,

    DateImport datetime NOT NULL,

    SupplierID varchar(20) NOT NULL,

    Total decimal(10,2) NOT NULL,

    PRIMARY KEY (ImportID)

);

-- Create ImportDetailTable

CREATE TABLE ImportDetailTable (

    ImportID varchar(20) NOT NULL,

    ProductID varchar(20) NOT NULL,

    Quantity int NOT NULL,

    Price decimal(10,2) NOT NULL,

);

CREATE TABLE Delivery (

    deliveryID INT IDENTITY(1,1) PRIMARY KEY,

    AgentID varchar(50) NOT NULL,

    OrderID varchar(50) NOT NULL,

    paymentStatus VARCHAR(20),

    deliverStatus VARCHAR(20)

);

GO

-- tự động tạo đơn delivery

CREATE TRIGGER InsertDelivery

ON OrderTable

AFTER INSERT

AS

BEGIN

    INSERT INTO Delivery (AgentID, OrderID, paymentStatus, deliverStatus)

    SELECT i.AgentID, i.OrderID, 'unpaid', 'preparing'

    FROM inserted i

END

GO

-- cập nhật lại số lượng trong kho

CREATE TRIGGER UpdateInventory

ON OrderDetail

AFTER INSERT

AS

BEGIN

    UPDATE Inventory

    SET InStock = InStock - i.Quan,

        OnOrder = OnOrder + i.Quan

    FROM Inventory

    INNER JOIN inserted i

        ON Inventory.ProductID = i.ProductID

END

GO

INSERT INTO Product(ProductID, ProductName, Price, Supplier)

VALUES

  ('P1', 'iPhone 13 Pro Max', 2299, 'Apple'),

  ('P2', 'Samsung Galaxy S21 Ultra', 1199, 'Samsung'),

  ('P3', 'Google Pixel 6 Pro', 1099, 'Google'),

  ('P4', 'OnePlus 10 Pro', 999, 'OnePlus'),

  ('P5', 'Xiaomi Mi 12', 899, 'Xiaomi'),

  ('P6', 'Oppo Find X3 Pro', 1299, 'Oppo'),

  ('P7', 'Sony Xperia 1 III', 1399, 'Sony'),

  ('P8', 'Asus ROG Phone 5', 999, 'Asus');

GO

INSERT INTO Supplier(SupplierID, SupplierName, Address, Contact)

VALUES

  ('S1', 'Apple', 'Cupertino, CA', '1-800-275-2273'),

  ('S2', 'Samsung', 'Seoul, South Korea', '82-2-2053-3000'),

  ('S3', 'Google', 'Mountain View, CA', '1-650-253-0000'),

  ('S4', 'OnePlus', 'Shenzhen, China', '+86 755-2533-8888'),

  ('S5', 'Xiaomi', 'Beijing, China', '+86 10-6060-6666'),

  ('S6', 'Oppo', 'Dongguan, China', '+86 769-8505-8888'),

  ('S7', 'Sony', 'Tokyo, Japan', '+81 3-6748-2111'),

  ('S8', 'Asus', 'Taipei, Taiwan', '+886 2-2894-3447');

GO

INSERT INTO Inventory(ProductID, InStock, OnOrder, MinimumStockLevel, MaximumStockLevel, ImportPrice)

VALUES

  ('P1', 100, 50, 20, 200, 1500),

  ('P2', 150, 20, 30, 300, 1000),

  ('P3', 75, 10, 15, 150, 900),

  ('P4', 200, 30, 25, 250, 800),

  ('P5', 120, 15, 20, 180, 700),

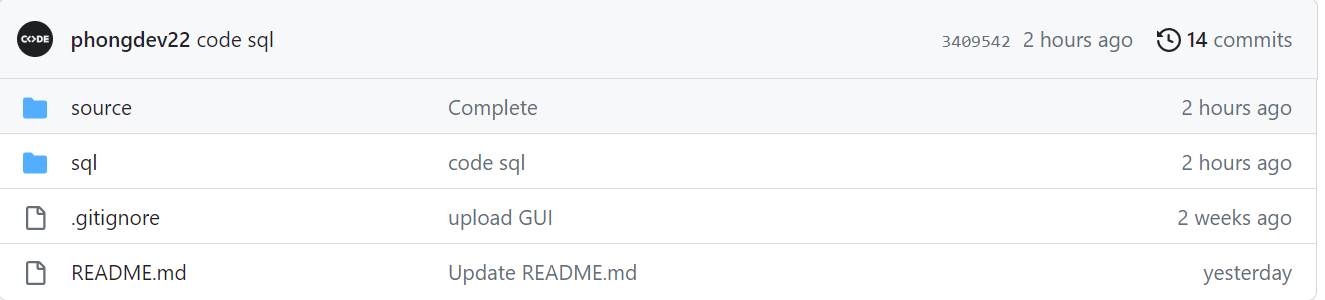
  ('P6', 80, 25, 10, 100, 1100),

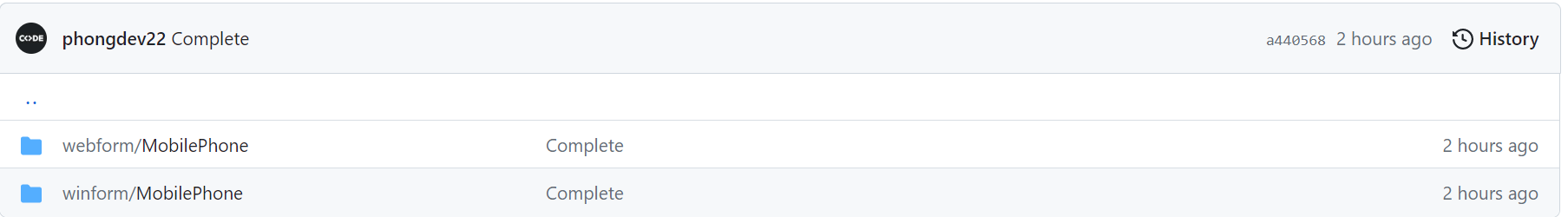
  ('P7', 90, 5, 10, 120, 1200),

  ('P8', 50, 10, 5, 75, 950);

**7.2. Source code**

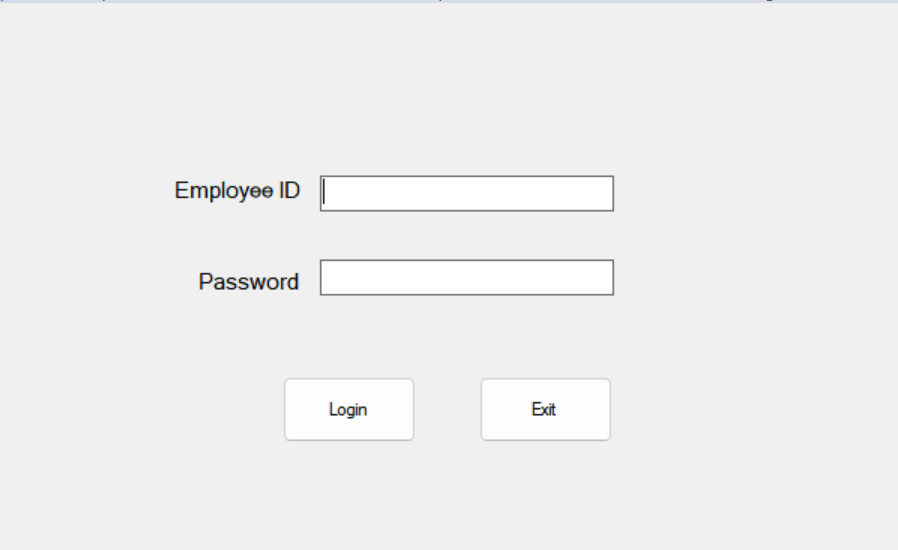
**Github:** <https://github.com/phongdev22/final_se>



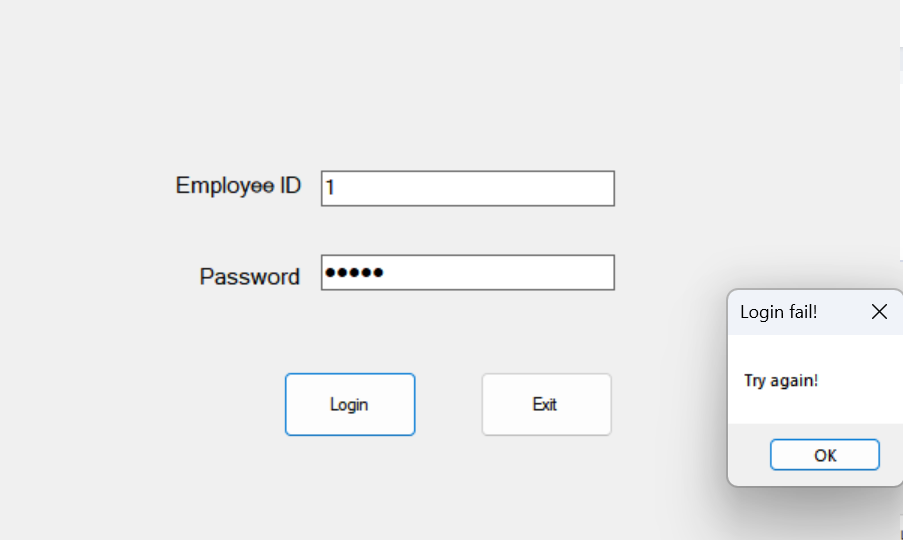


**7.3. Testing**

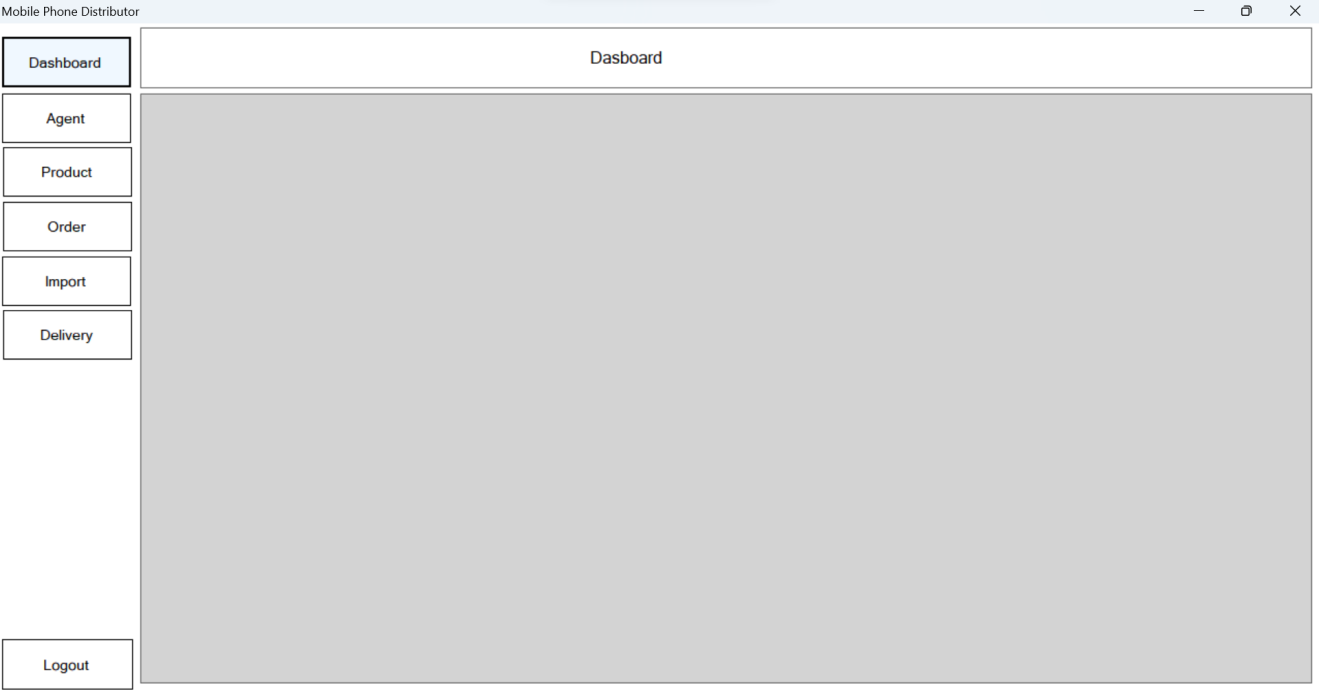
**Winform:**

**Login** 

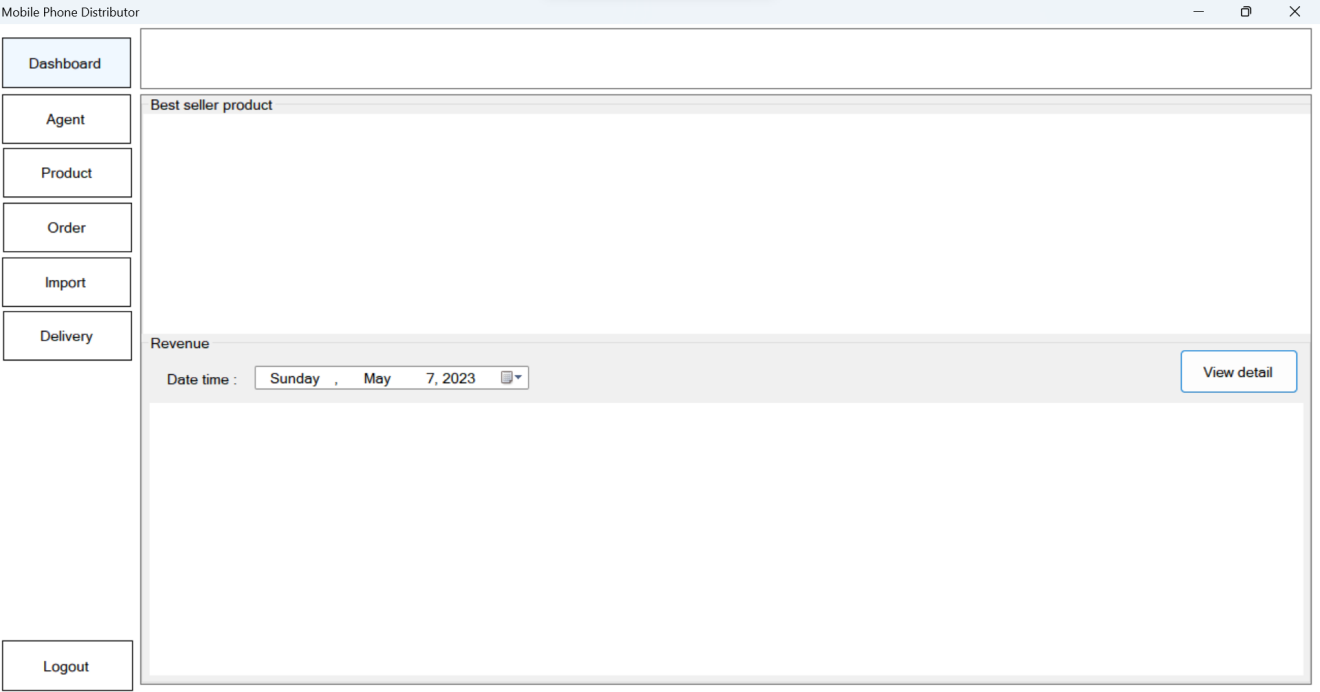
**Login Fail:**



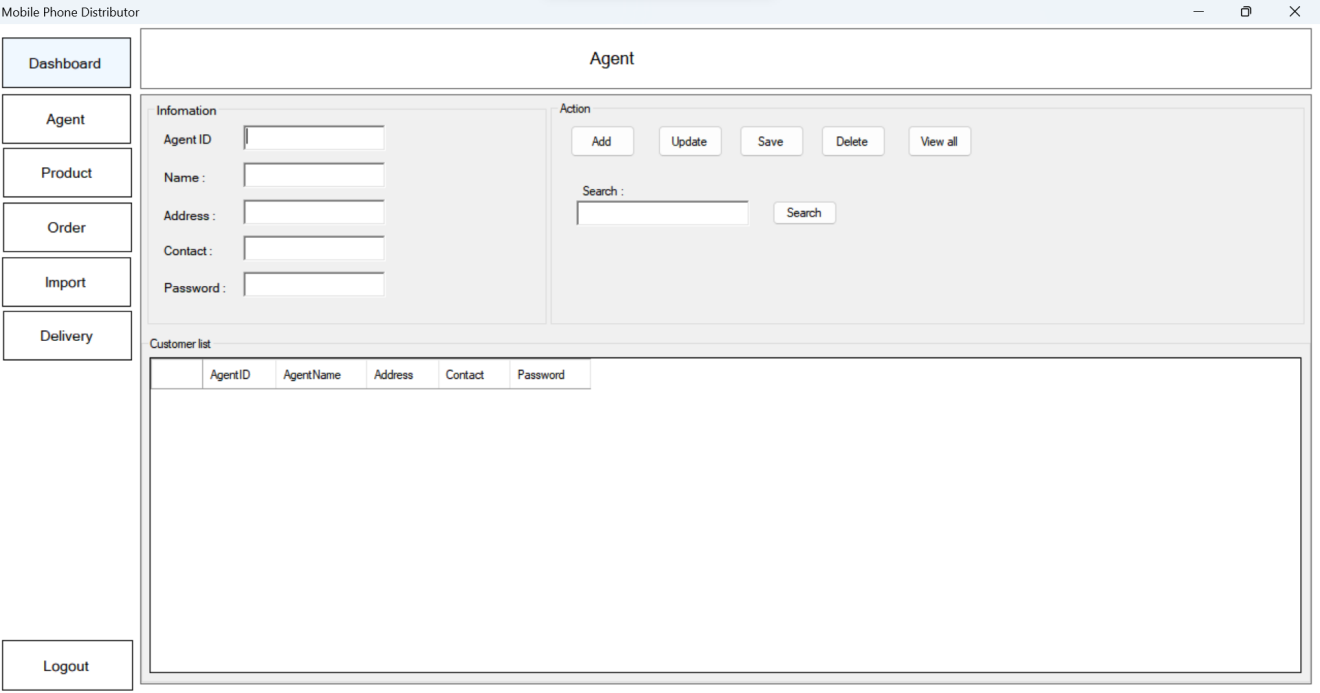
Login successful:



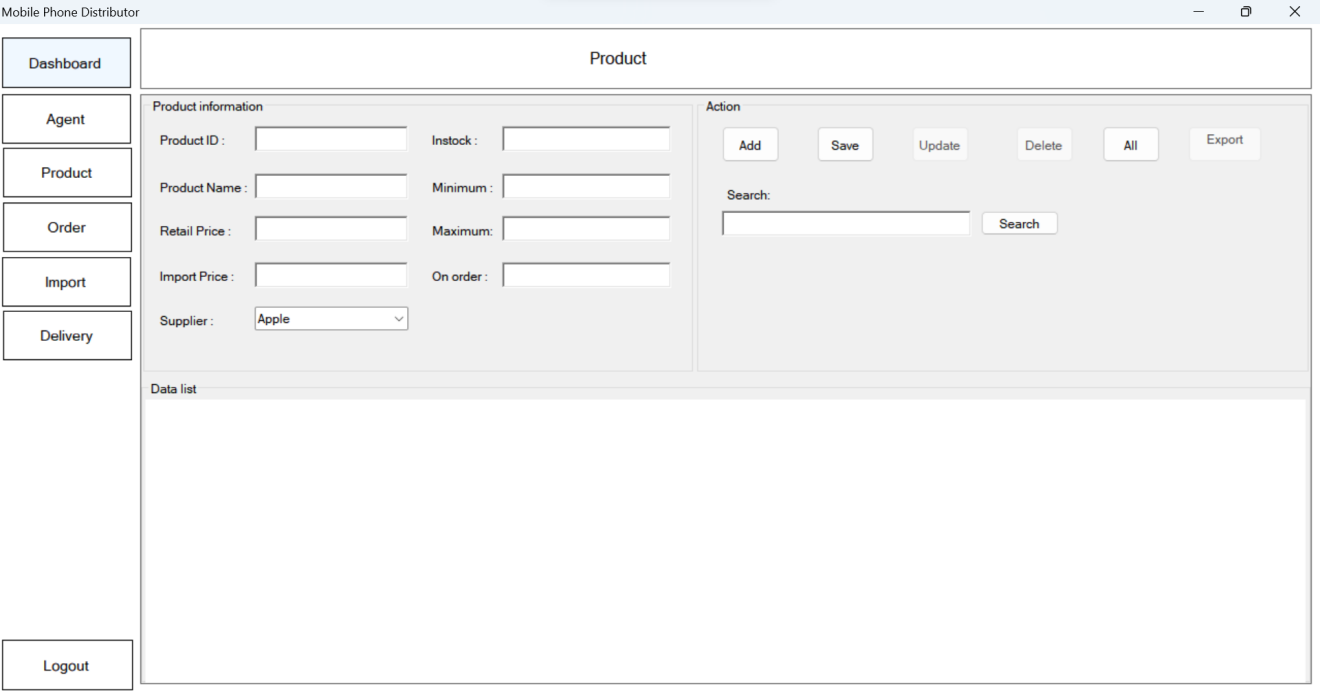
**Dasboard interface:**

****

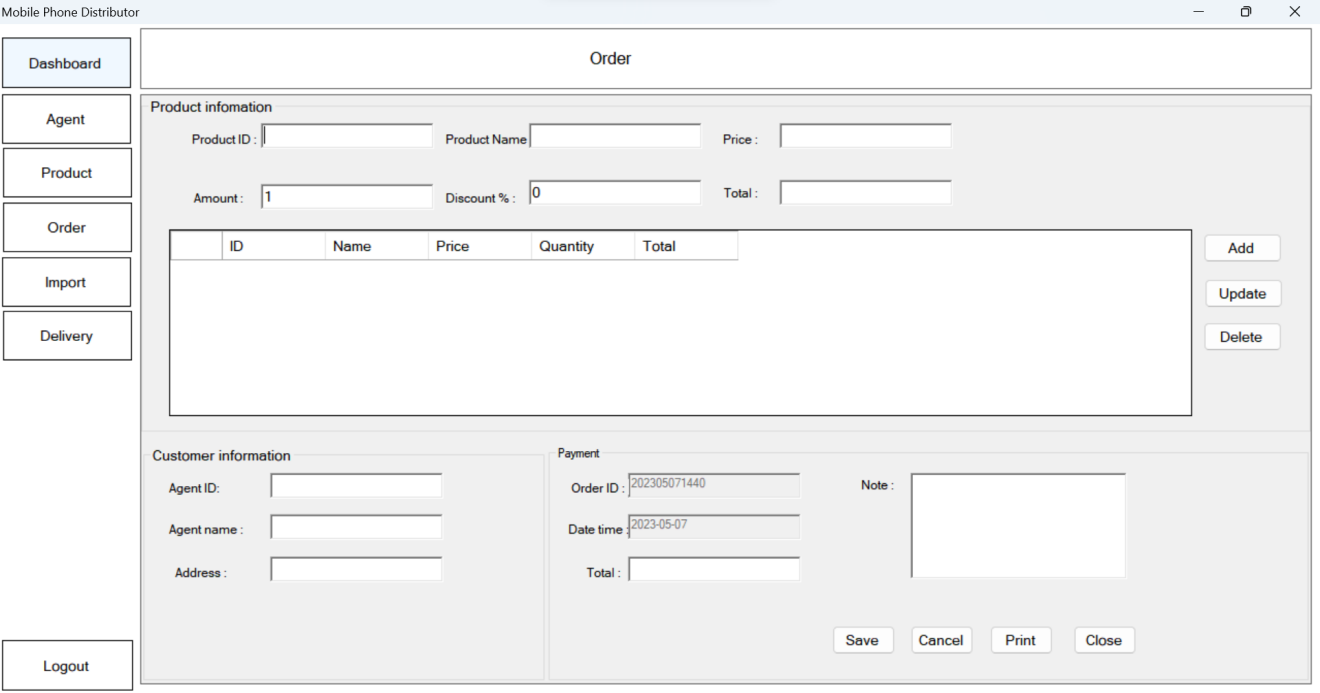
**Agent Interface:**



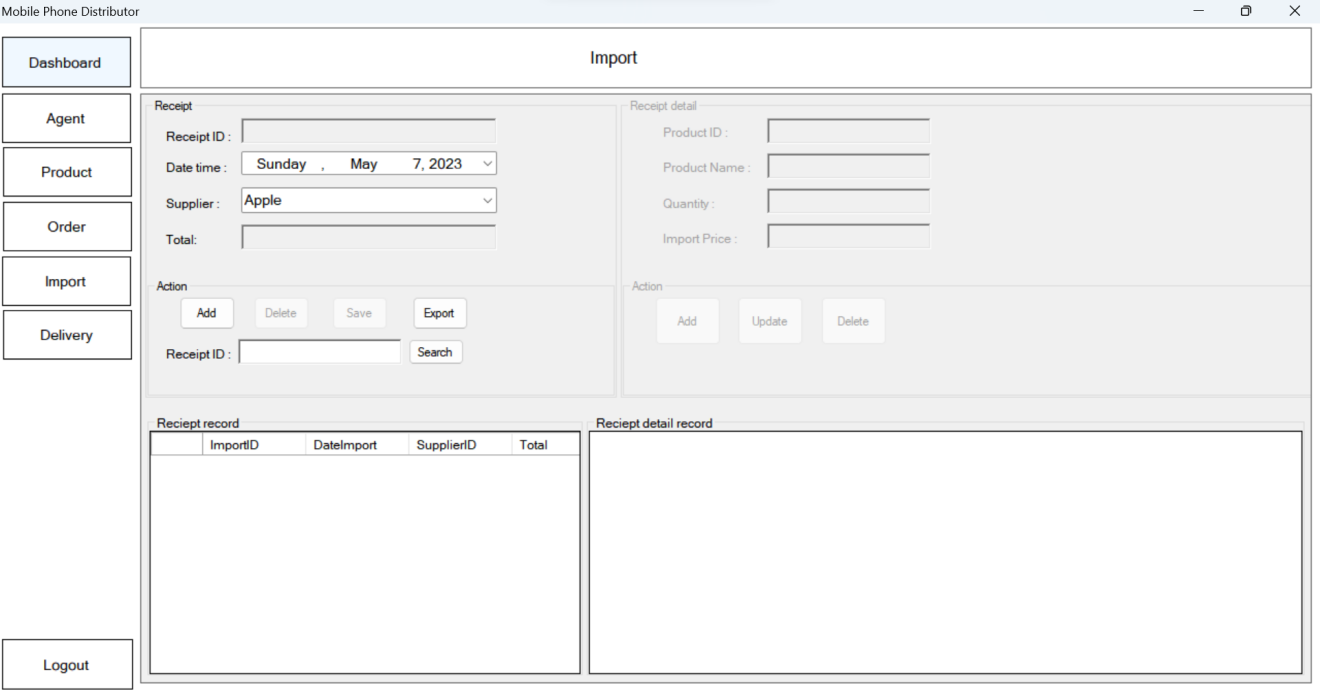
**Product Interface:**



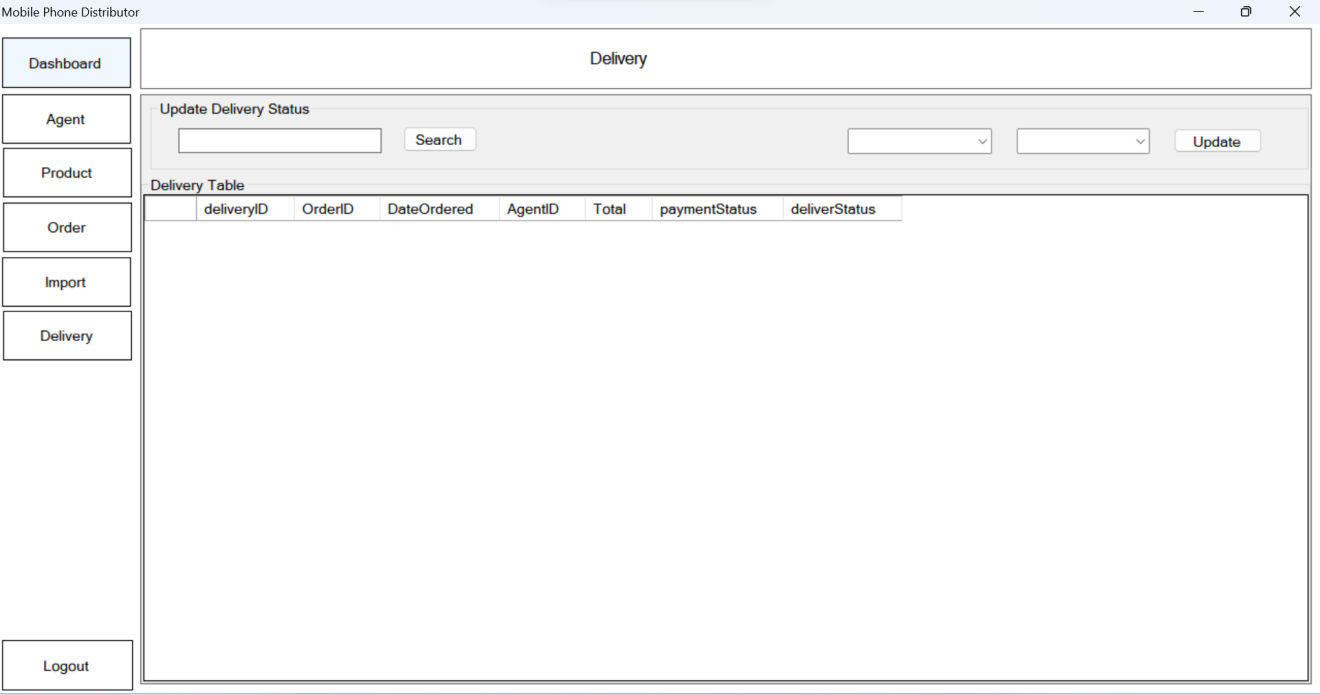
**Order Interface:**



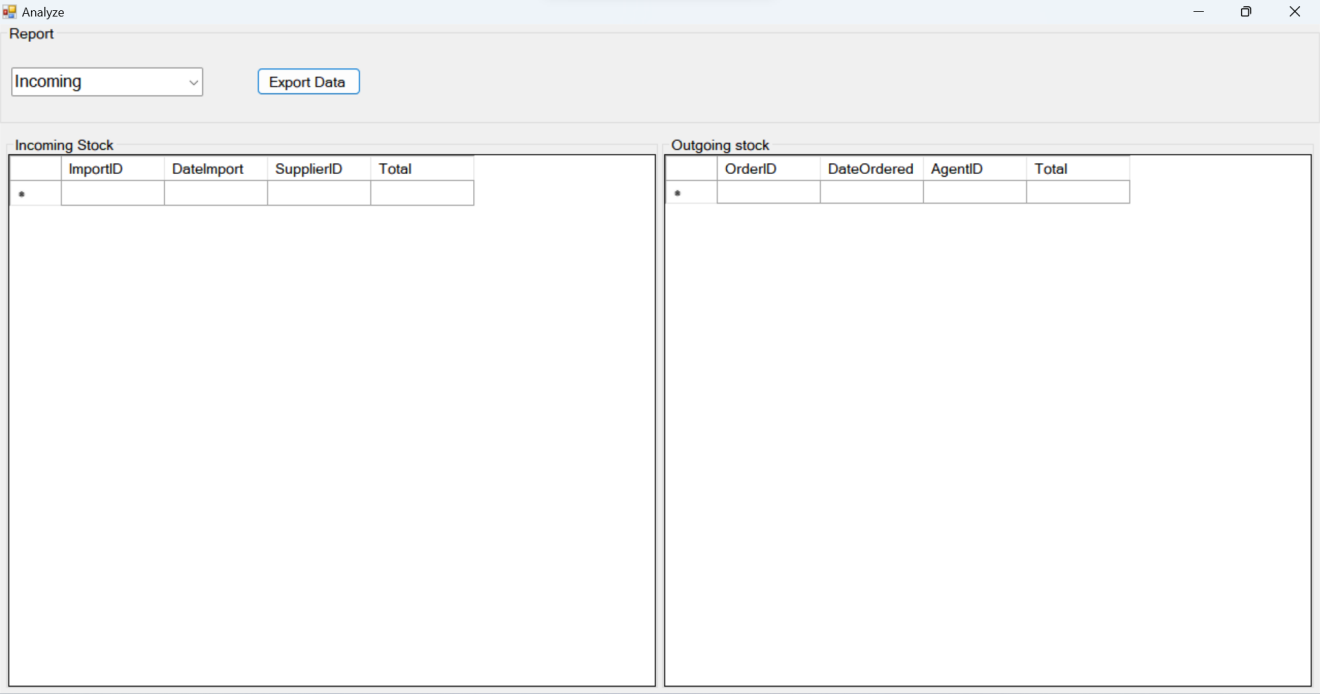
**Import Interface:**



**Delivery Interface:**

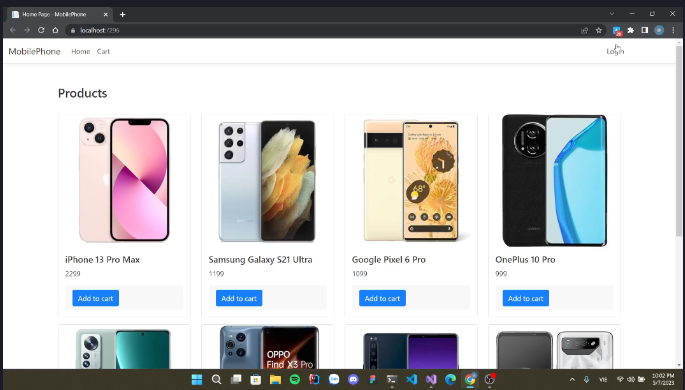


**View Detail Interface:**

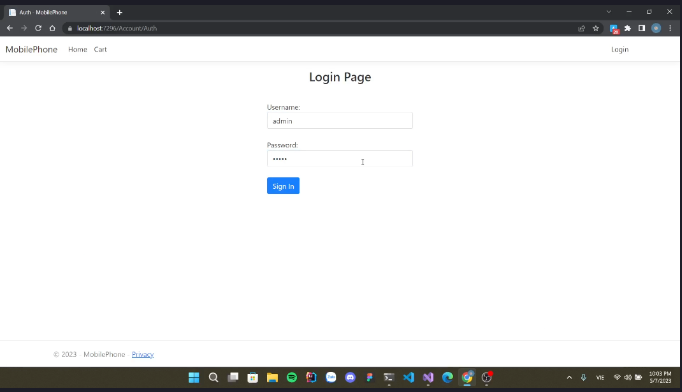


**Web form:**

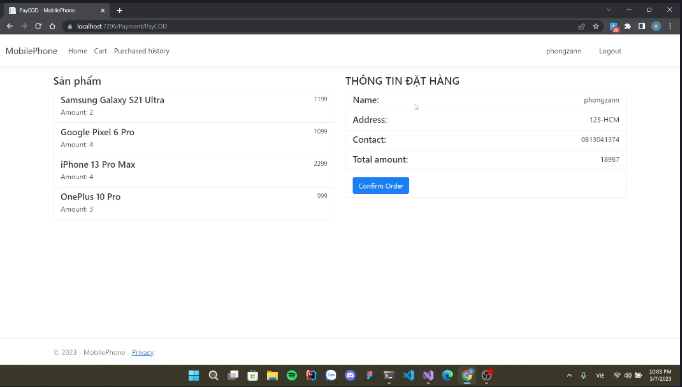
**Page interface:**



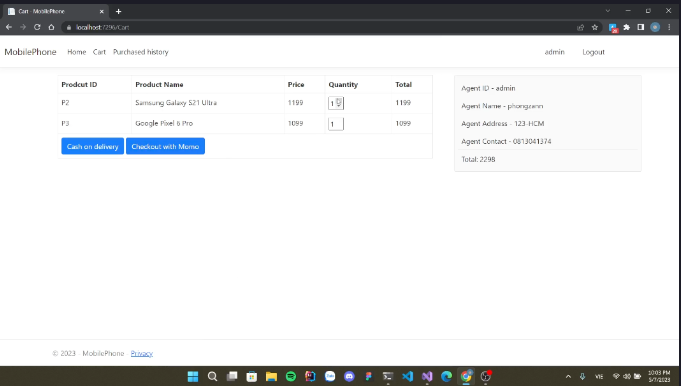
Login Interface:



Product ordering interface:



payment interface:



CHAPTER 8 – RUBRICS ASSESSMENT.

|  |  |  |
| --- | --- | --- |
| **Item** | Rubrics | self assessment |
| **1/ Project Management Plan** | **1.0** | **1.0** |
| **2/ Requirements Specification** | **1.0** | **0.75** |
| Use case diagrams | 0.5 | 0.5 |
| Use case specification | 0.5 | 0.25 |
| **3/ Architecture design** | **1.0** | **1.0** |
| **4/ Design** | **2.0** | **1.75** |
| Mockup/Prototype | 0.5 | 0.25 |
| Class diagrams | 0.5 | 0.5 |
| Sequence diagrams | 0.5 | 0.5 |
| Database design | 0.5 | 0.5 |
| **5/ Build apps with at least 4 main functions** | **2.0** | **1.5** |
| **6/ Coding Convention** | **0.5** | **0.25** |
| **7/ Testing, Test case, Unit test** | **1.0** | **0.5** |
| **8/ SVN/GIT** | **0.5** | **0.25** |
| **9/ Hình thức báo cáo, demo** | **1.0** | **1.0** |
| **10/ High Distinction Task or Extra Point** | **1** | **0** |
| **Total** | **10+1** | **8** |