

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

GRADUATION THESIS

E-commerce platform: A module for suppliers

DUONG THI HUE

hue.dt176772@sis.hust.edu.vn

Major: Information Technology

Supervisor: MSc. Nguyen Hong Phuong _____

Signature

Department: Computer Science

School: Information and Communications Technology

HANOI, 07/2022

ACKNOWLEDGMENTS

I would like to express my special thanks of gratitude toward my supervisor, Master Nguyen Hong Phuong, who has provided able guidance and support in completing my graduation research.

ABSTRACT

In the 4.0 revolution, the considerable development of technology makes so many utilities for humans in every aspect, specifically in shopping demand. However, because of the distance and time, nowadays, people take more time online shopping, the online e-commerce appears. Especially in covid 19 disease, online shopping has emerged as a "lifesaver," helping consumers secure their lives and jobs and helping manufacturers and distributors of goods develop production and business. Currently, Vietnam has nearly 45 million people participating in online shopping. Online shopping is convenient and saves time. Consumers can freely choose the necessary items and necessities for daily life and work without spending too much time everywhere that has the internet.

However, online shopping also has many problems. The seller delivers poor quality, counterfeit goods. Many sellers only post pictures and prices, but the buyers wait forever without delivery, even the buyers finish trading. The risk of personal information being exposed when online shopping.

There are some solutions for customers to prevent risks when online shopping. Firstly, customers chose the prestige e-commerce store. Secondly, the government should take action to avoid goods of prestige from online shopping.

I think about creating an e-commerce site to help customers buy authentic goods with reasonable shipping fees and the shortest time shipment. To do that, it needs to make a subsystem for suppliers to manage their products and distributors. This graduation research is about a module for suppliers in my e-commerce system. I named it Soda e-commerce. This subsystem has a user-friendly experience and a user-friendly user interface to help suppliers bring their products to Soda e-commerce.

TABLE OF CONTENTS

LIST OF ABBRIVIATIONS	v
CHAPTER 1. INTRODUCTION.....	1
1.1 Motivation	1
1.2 Objectives and scope of the graduation thesis	1
1.3 Tentative solution	1
1.4 Thesis organization	2
CHAPTER 2. REQUIREMENT SURVEY AND ANALYSIS.....	3
2.1 Status survey	3
2.2 Functional Overview	4
2.2.1 General use case diagram.....	4
2.2.2 Use case request to be supplier	5
2.2.3 Use case request to be distributor.....	5
2.2.4 Use case approve/reject the request.....	5
2.2.5 Use case manage product.....	5
2.2.6 Use case manage order	6
2.2.7 Use case manage warehouse.....	6
2.2.8 Use case view reports	7
2.2.9 Use case update store name.....	7
2.2.10 Use case reset password.....	7
2.2.11 Use case forgot password	7
2.2.12 Use case login.....	7
2.2.13 Use case log out.....	7
2.2.14 Received order	7

2.2.15 Business process	7
2.3 Functional description.....	9
2.3.1 Description of Use case request to be supplier	9
2.3.2 Description use case login	9
2.3.3 Description use case manage product.....	11
2.3.4 Description use case accept/deny an order	13
2.4 Non-functional requirement.....	14
CHAPTER 3. TECHNOLOGY	15
3.1 Technology	15
3.1.1 HTML	15
3.1.2 JavaScript.....	15
3.1.3 CSS	15
3.1.4 Blazor.....	15
3.1.5 Abp framework.....	16
3.1.6 MongoDB	16
3.1.7 Elastic search.....	16
3.1.8 RabbitMq.....	16
3.1.9 Redis	17
3.1.10 S3.....	17
CHAPTER 4. EXPERIMENT AND EVALUATION.....	19
4.1 Architecture design.....	19
4.1.1 Software architecture selection	19
4.1.2 Overall design.....	20
4.1.3 Detailed package design	23
4.2 Detailed design.....	25
4.2.1 User interface design.....	25

4.2.2 Layer design	27
4.2.3 Database design	29
4.3 Application Building.....	32
4.3.1 Libraries and Tools.....	32
4.3.2 Achievement.....	33
4.3.3 Illustration of main functions	33
4.4 Testing.....	43
4.5 Deployment	44
CHAPTER 5. CONCLUSION AND FUTURE WORK	45
5.1 Conclusion.....	45
5.2 Future work.....	47
REFERENCE	48

LIST OF FIGURES

Figure 2.1	General use case diagram	4
Figure 2.2	Use case manage product	5
Figure 2.3	Use case manage order	6
Figure 2.4	Use case manage warehouse	6
Figure 2.5	Activity diagram process an order	8
Figure 3.1	Messaging in rabbitMQ	17
Figure 4.1	UML package diagram of subsystem	20
Figure 4.2	UML package diagram of 1 micro service	22
Figure 4.3	Detail package design use case register to be supplier	24
Figure 4.4	success message	26
Figure 4.5	warning message	26
Figure 4.6	reset password page	26
Figure 4.7	Sequence diagram of use case register to be supplier	27
Figure 4.8	Sequence diagram of use case manage warehouse	28
Figure 4.9	Sequence diagram of use case update product	28
Figure 4.10	ER diagram of authentication service	29
Figure 4.11	ER diagram of sale service	30
Figure 4.12	ER diagram of warehouse service	31
Figure 4.13	ER diagram of partner ship service	32
Figure 4.14	Login page	34
Figure 4.15	Register page	35
Figure 4.16	Create product page	37
Figure 4.17	Create warehouse page	39
Figure 4.18	List order page	40
Figure 4.19	Detail of unconfirmed order page	41
Figure 4.20	Chose warehouse page	42
Figure 4.21	Report growth page	43

LIST OF TABLES

Bảng 2.1	Main flow of use case register to be supplier	9
Bảng 2.2	Alternative flow of use case register to be supplier	9
Bảng 2.3	Main flow of use case login	10
Bảng 2.4	Alternative flow of use case login	10
Bảng 2.5	Main flow of use case create product	11
Bảng 2.6	Alternative flow of use case create product	11
Bảng 2.7	Main flow of use case delete product	12
Bảng 2.8	Alternative flow of use case delete product	12
Bảng 2.9	Main flow of use case search product	13
Bảng 2.10	Main flow of use case accept/deny an order	13
Bảng 2.11	Alternative flow of use case create product	14
Bảng 4.1	List of libraries and tools	33
Bảng 4.2	Screen specification of login page	34
Bảng 4.3	Define attribute field of login page	34
Bảng 4.4	Screen specification of register page	35
Bảng 4.5	Define attribute field of register page	36
Bảng 4.6	Screen specification of create product page	37
Bảng 4.7	Define attribute field of create product page	38
Bảng 4.8	Screen specification of create warehouse page	39
Bảng 4.9	Screen specification of list order page	40
Bảng 4.10	Define attribute field of list order page	40
Bảng 4.11	Screen specification of unconfirmed order detail page	41
Bảng 4.12	Screen specification of chose warehouse in order detail page	42
Bảng 4.13	Screen specification of report growth page	43
Bảng 4.14	Testing result	44

Acronym	English name
UI	User interface
API	Application Programming Interface
ABP	AspNet Boilerplate
E-commerce	Electronic commerce
AWS	Amazon web service
S3	Simple cloud storage
DDD	Domain driven design
UML	Unified Modeling Language

CHAPTER 1. INTRODUCTION

1.1 Motivation

Currently, in the market, according to the trading model, a manufacturer will produce products, and distributors will import and sell to dealers. From there, retailers will bring products to consumers. The management of distributors and agents in practice is quite difficult. Among distributors, dealers have price differences, and the appearance of counterfeit goods does not guarantee product quality. In addition, price is increase two or three times because intermediaries make prices, "pickpocket" consumers. Shoppers have few choices and suggestions and can't tell the difference between fakes. Solving these problems helps the market become stable, suppliers can manage distributors, and buyers are satisfied with the quality and amount of money spent.

1.2 Objectives and scope of the graduation thesis

There are a lot of e-commerce platforms on the market, especially in Vietnam. However, price differences and counterfeit goods have not been taken care of. Most e-commerce platforms aren't interested in distributors and dealers yet. On the side of customers, users are not concerned about the quality of products when shopping online. Distributors and agents do not have appropriate policies.

Therefore, I develop e-commerce that focuses on suppliers, distributors but still brings authentic products to the customer. My E-commerce creates an easy and convenient exchange of purchases. This e-commerce solves management issues of price increment when trading among distributors, maximum cut of intermediaries, bringing quality products to consumers. In addition, the suppliers can manage the distributors bought their goods. This research focused on the module for the supplier in that e-commerce. The main functions of suppliers are managing their products, stock inventory, orders, and reports in this project.

1.3 Tentative solution

It is necessary to create an e-commercial to help suppliers manage distributors and bring products to consumers. I generate a provider's management software to help suppliers can manage their resources. Firstly, I thought about using microservice architecture. After research, I chose ASP.NET Boilerplate (ABP), an open-source and well-documented application framework. This framework can help by offering a microservice-compatible, strict module architecture where your module is split into multiple layers/projects and developed independently. The system needs

an API gateway as an intermediary between the client and the back-end system microservices. This gateway helps hide the structure of the microservices system from the outside and quickly tracks and manages traffic. For the user interface, I use blazorise, a framework for building interactive client-side web UI. NET. For storage data, I decide to use MongoDB. The research provides the solution for managing suppliers in the e-commerce system.

1.4 Thesis organization

The other parts of graduation research are organized as follows. Chapter 2 present the main functional and non-functional requirement of the project. It mentions a website for the supplier in an e-commerce system in a distributed architecture. From the survey, I pointed out the critical business in this system and the detailed specifications of each use case. This chapter also describes essential business processes in the activity diagram.

In chapter 3, I introduce about technology applied in this project to perform the required function in chapter 2.

Architecture design, detail design, application building, testing, and deployment are in chapter 4. Architecture design describes the software architecture with the supplement or improvement and the overall design project with a UML diagram. In the detail design section, I provide a specific description of user interface design, layer design, and database design. The library and tools used in this project are mentioned in the application building section. In this section, the achievement and illustration of the main functions are also explicitly described. Testing is an indispensable part of this chapter. I design test cases for the essential tasks in this section and analyze testing results. Finally, the deployment section demonstrates the model and how to deploy the project in practice.

Finally, Chapter 5 is the conclusion of this project. This chapter is about the process of working on this project, the difficulties of mine, and how to face and solve each difficulty, the upgrade. The experiment achievement and expensive lessons of mine are mentioned in this chapter.

CHAPTER 2. REQUIREMENT SURVEY AND ANALYSIS

2.1 Status survey

A small survey asking 20 people about their experience when shopping online showed that most of them had at least once bought a fake. In addition, some people have to wait half a month to receive their goods, and the shipping fee also affects their decision to order. If the shipping fee is small, they will complete the order quickly. Otherwise, they will consider the importance of the product and may not place the order.

The supplier wants to manage its product and the distributors in the market. Currently, on the market of e-commerce systems in Vietnam (such as Shopee, Lazada, Tiki, etc.), most of them are platforms for retailers and end consumers, so retailers will have to find their source of goods. That leads to unstable sources of goods, unknown origin, non-optimal import prices, and complicated transportation methods, for example, having a close source but having to import the same item from afar. Therefore, it is necessary to have an e-commerce platform between the manufacturer (where there is a stable, safe, and transparent source of goods) and the supplier and distributors.

Recently, similar platforms have appeared on the Vietnamese market. For example, Telio is a platform that imports groceries or drugs from a distributor to a retail store. Or Alibaba is a place of direct exchange from factories or trading companies to retailers or end users. As explained above, the value that these e-commerce platforms bring is the ability to provide a stable source of goods, helping small and medium-sized commercial units not have a broken supply chain, and at the same time can source selection. With reasonable import prices and optimal shipping methods.

This research will make an e-commerce platform similar to the above, called Soda. This project describes an e-commerce platform between Suppliers and Distributors and Customers. Customers buy goods from distributors, and the system will split orders into small orders by supplier. The closest distributors has enough goods to qualify for small order and accepts that the order will be prepared and given to delivery. Distributors have to buy items with a minimum quantity corresponding. Suppliers will prepare these orders and give them to deliver. This research is a module for a supplier in the above e-commerce system. It includes the following features:

- Manage resources for suppliers (including Inventory, Products, Orders, etc.).
- Business reports (by filters).
- Manage ordered distributor information.
- Distributors can select and order from multiple suppliers in one order.
- Suppliers can be flexible in choosing warehouses for picking and receiving goods.

2.2 Functional Overview

In the subsystem of suppliers, suppliers can manage their warehouse, product, order, membership functions, view reports, and export reports.

2.2.1 General use case diagram

The figure 2.1 show the general use cases, and actors join in subsystem for supplier.

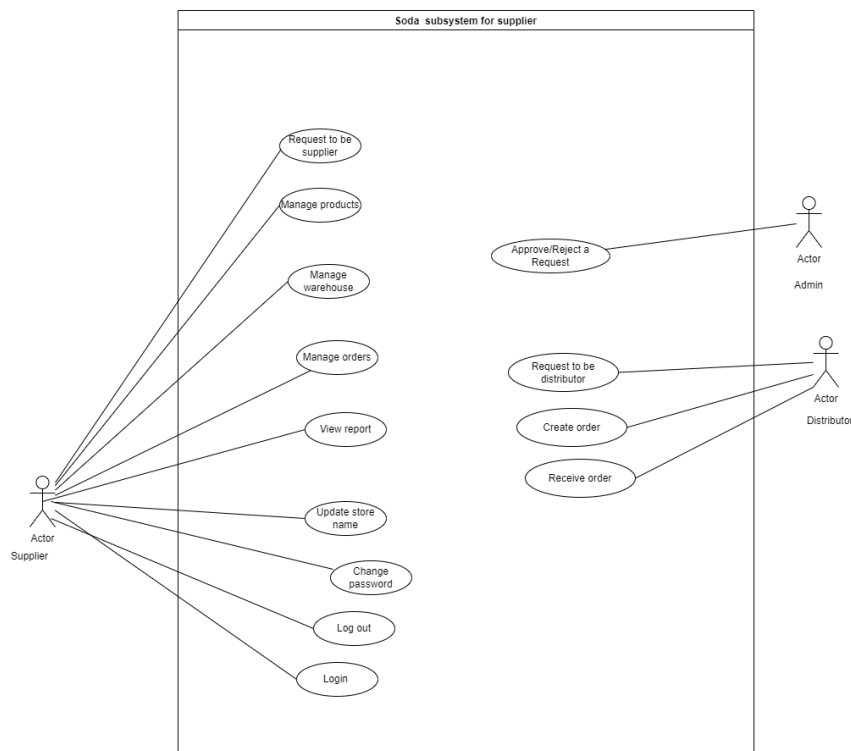


Figure 2.1: General use case diagram

This software has three actors: administrator, distributor, and supplier. The administrator can approve or reject a request of a supplier or distributor. The distributor can create/cancel an order. The system will separate the order into small orders based on suppliers. Each small order will be given to the corresponding supplier on their website. In this project, for simplicity the shipping fee is zero. The suppliers are the company responsible for their products about the quality and origin. To come

to a supplier in the system, they have to request to be suppliers when suppliers can manage their resources and membership functions.

2.2.2 Use case request to be supplier

The suppliers must request to be suppliers by filling in all the required information. After the request sends successfully, the System will send an email with the link to follow the recommendation.

When the admin rejects the request, System will send an email with the reason for rejection to the supplier. They can update the request with the link in the email. If the admin accepts the request, the System emails the supplier containing the account to login into the System.

2.2.3 Use case request to be distributor

Similar to a use case request to the supplier, the distributor who wants to join the system must create a request. And wait for the admin to approve it.

2.2.4 Use case approve/reject the request

Admin can approve or reject the request of the supplier. After approval, the system will send an email with account information to a supplier so that the supplier can log in to the system or if the reject system will send an email with a reason and link to update the request. Admin also has other actions with the supplier, such as blocking, editing information, and changing the password, but these use cases are not in this module. Admin can approve/reject the distributor request similar to the supplier's request.

2.2.5 Use case manage product

Figure 2.2 below is use case diagram of use case supplier manage their product.

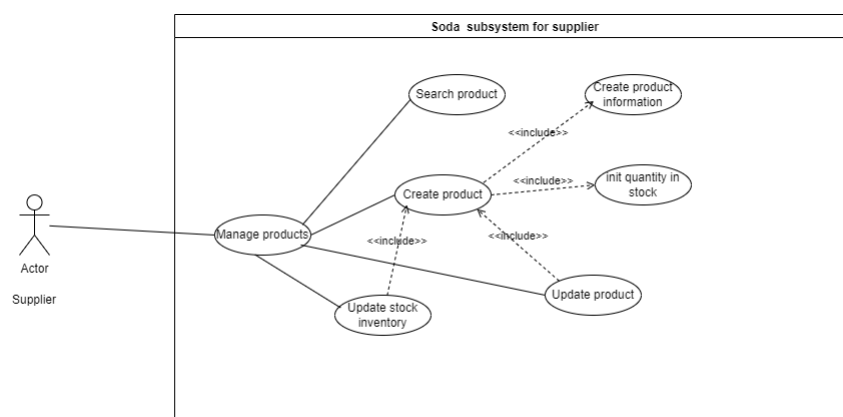


Figure 2.2: Use case manage product

The supplier can create a product, update information, and delete the product.

When editing product information, they can adjust the stock inventory of that product in each warehouse.

2.2.6 Use case manage order

Use case diagram of use case supplier manage order is show in figure 2.3

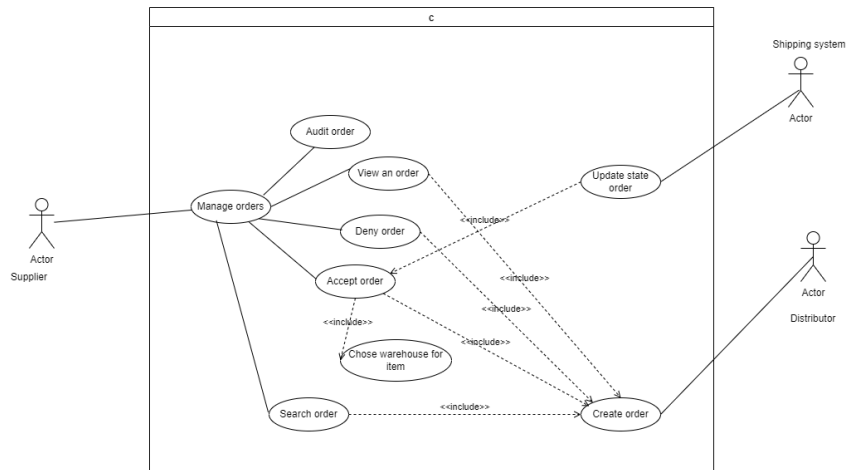


Figure 2.3: Use case manage order

The supplier can view the distributor's order, and they can accept or deny an order. If they take, they have to pick a stock with an amount for each item in the order. After preparing the order, they can change the order's status to shipping. They can also search order by the order number or address of the receiver.

2.2.7 Use case manage warehouse

The supplier has so many warehouses located in different locations. The figure 2.4 below describe use case diagram of use case supplier manage warehouse:

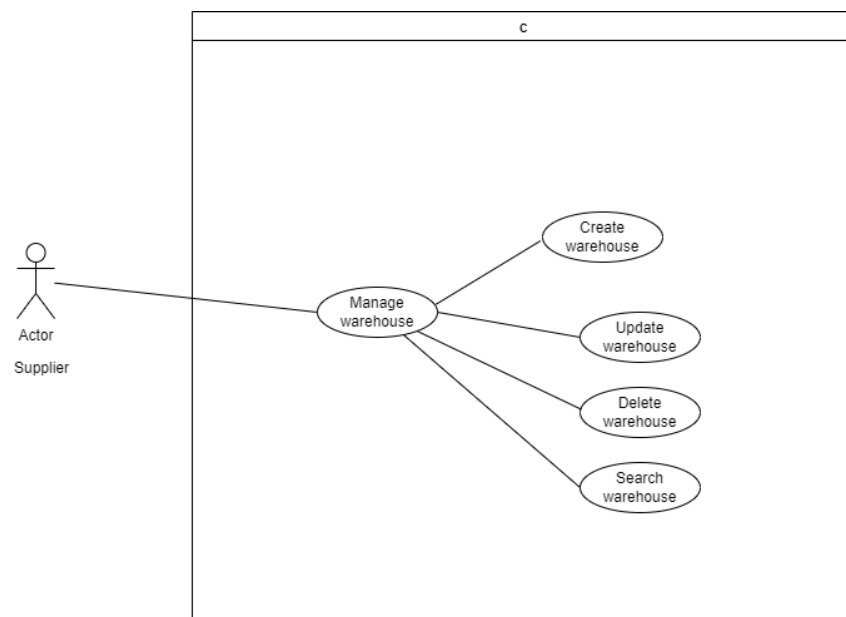


Figure 2.4: Use case manage warehouse

The module provides some actions with a warehouse: creating, updating, deleting, viewing a piece of information, and searching the warehouse by name.

2.2.8 Use case view reports

The supplier can view the report of the product and order. The supplier can export a file report in excel with the defined template in this use case.

2.2.9 Use case update store name

The supplier has a store name and website URL updated on the website. This information will appear in the view of the distributor, retailer, and customer.

2.2.10 Use case reset password

The supplier can reset their password. After resetting successfully, they can log in with the new password.

2.2.11 Use case forgot password

When supplier forgets their password, they can use this function. After they submit the email, the system will send the link to reset the password in an email. Then supplier can change to login with a new password.

2.2.12 Use case login

To access the system, supplier has to login with correct tax-code and password.

2.2.13 Use case log out

The supplier also can log out of the system.

2.2.14 Received order

Distributor received order from deliver.

2.2.15 Business process

The processing an order from beginning until delivered is describe as Figure 2.5:

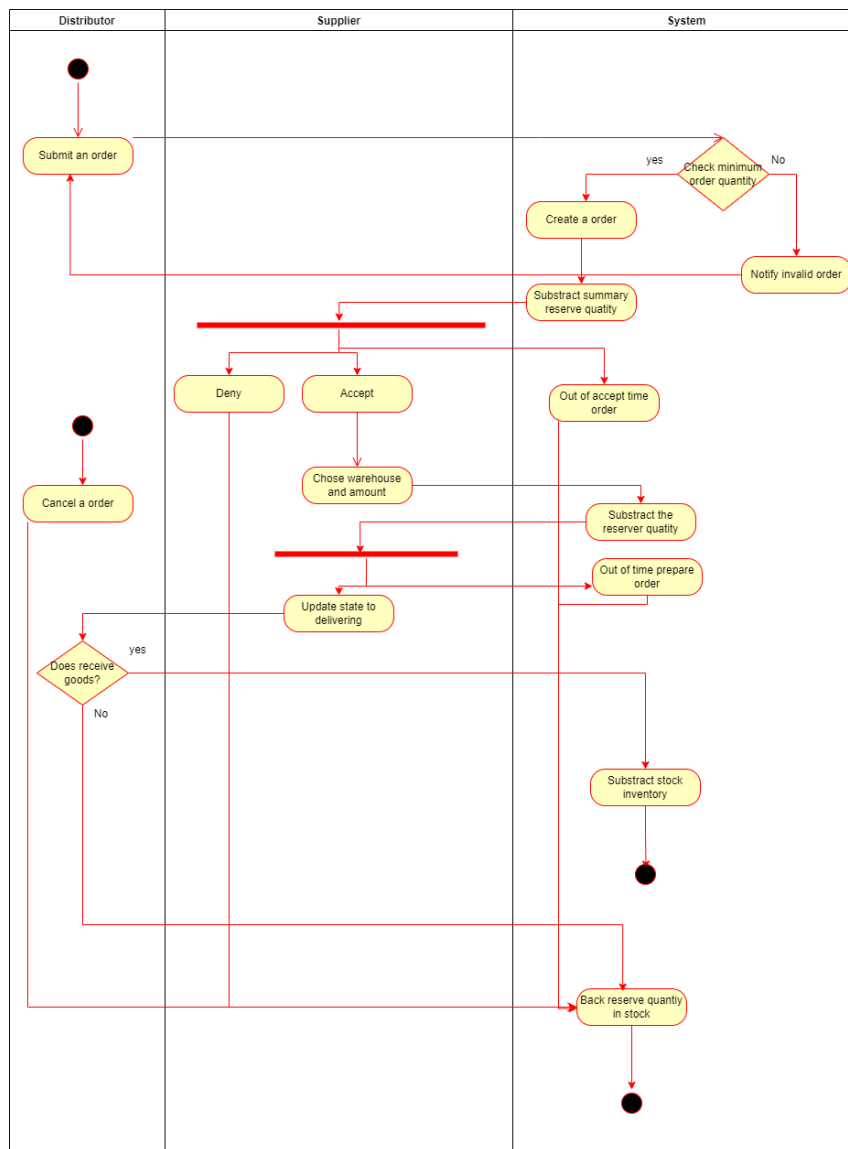


Figure 2.5: Activity diagram process an order

As you can see, the distributor creates a valid order if the number of items is more significant than a minimum order quantity which is a product attribute. Whenever distributors make an order, the supplier's reserve quantity will be subtract. The reserved amount is to guarantee enough stock inventory for other orders. The supplier can take the items from many warehouses in enough quantity. There is a time limitation in accepting orders and preparing orders. If out of this time, the order will automatically be canceled. The distributor can cancel the order before it is delivered or if they do not receive a charge from the shipper. When an order is delivered, the stock inventory will update.

2.3 Functional description

2.3.1 Description of Use case request to be supplier

Brief Description: As a supplier, I want to join Soda e-commerce. I have to register to be supplier in website of supplier.

Actors:Supplier, Administrator, System

Priority:Require

Trigger(s):When user select "Đăng ký trở thành NCC" in login page

Precondition(s):User access to login page

Main flows is shown in table2.1:

#	Actor	Action
1	User	Submit the form
2	System	Send email include the flowing link
3	Admin	Accept request
4	System	Send email include account information

Table 2.1: Main flow of use case register to be supplier

Alternative Flows is shown in table 2.2:

#	Actor	Action
2a	Admin	Reject the request
	System	Send email include the flowing link and reason
	Supplier	Update the request and return to 2

Table 2.2: Alternative flow of use case register to be supplier

2.3.2 Description use case login

Brief Description: As a supplier, I want to login into system.

Actors:Supplier

Priority:Require

Precondition(s):User already have account

Main flows is described in table 2.3:

#	Actor	Action
1	User	Submit the login form
2	System	Check the username and password
3	System	Redirect into dashboard

Table 2.3: Main flow of use case login

Alternative Flows is described in table2.4:

#	Actor	Action
1a	User	Submit forgot password
	System	Redirect to forgot password page
	User	Submit phone number or email
	System	Check exist phone number or email
	System	Send email with link to reset password
	User	Submit the new password
	System	Redirect to successfully reset password and return to 1
1c	User	Click on remember password
	System	Save the token to storage and return dashboard

Table 2.4: Alternative flow of use case login

2.3.3 Description use case manage product

a, Use case create product

Brief Description: Supplier want to create their product

Actors:Supplier

Priority:Optional

Trigger(s):When user select "Thêm mới sản phẩm" in menu

Precondition(s):User login successfully into system as a supplier

Main flows is in table 2.5:

#	Actor	Action
1	Supplier	Submit the form
2	System	System check requirement field, return successful notification
3	System	Redirect to list page

Table 2.5: Main flow of use case create product

Alternative Flows is in table 2.11:

#	Actor	Action
1a	User	Submit to cancel
	System	Redirect to previous page

Table 2.6: Alternative flow of use case create product

b, Use case delete product

Brief Description: Supplier want to delete the product

Actors:Supplier

Priority:Optional

Trigger(s):When user click on button delete in list page

Precondition(s):User in list product page, has at least 1 product

Main flows:

#	Actor	Action
1	Supplier	Submit delete product
2	System	System check product in any order
3	System	Notify delete successfully

Table 2.7: Main flow of use case delete product

Alternative Flows:

#	Actor	Action
2a	System	Product already in at least 1 order, system return error

Table 2.8: Alternative flow of use case delete product

c, Use case search product

Brief Description: Supplier search product by name, code, warehouse, category

Actors:Supplier

Priority:Optional

Trigger(s):When user click into "Danh sách sản phẩm" in menu

Precondition(s):User login successfully into system as a supplier

Main flows:

#	Actor	Action
1	Supplier	Submit filter
2	System	Display list product with condition in filter

Table 2.9: Main flow of use case search product

2.3.4 Description use case accept/deny an order

Brief Description: Supplier want to accept/deny an order

Actors:Supplier

Priority:Optional

Trigger(s):When user in order detail page

Precondition(s):User have at least 1 order by distributor created

Main flows:

#	Actor	Action
1	Supplier	Chose warehouse and quantity for each item in order detail
2	Supplier	Accept the order

Table 2.10: Main flow of use case accept/deny an order

Alternative Flows:

#	Actor	Action
1a	Supplier	Deny an order
1b	Supplier	Accept order
	System	Notify that require to choose warehouse for each item

Table 2.11: Alternative flow of use case create product

2.4 Non-functional requirement

Firstly the software should continuously operate correctly and responsively in any general cases. However, a slight drop in performance and response time is allowable in some exceptional cases. Implicitly stated, ideally, the response time for any tasks, with a moderate load, within the system is 1 second. But in the case of peak load, a response time of 3 seconds is permissible.

- The system should be able to serve a good number of users in discrete periods.
- In the case of simultaneous users, the system is expected to serve up to 100 requests.
- The system should run smoothly, consecutively, automatically, and reliably. Ideally, more than 200 hours of operating without failure are acceptable.

Secondly, the user interface must be friendly and bring a good user experience. Thirdly, the system requires maintenance and development, so the architecture and source code should be good. Finally, the system requires a high-security level.

CHAPTER 3. TECHNOLOGY

3.1 Technology

3.1.1 HTML

HTML, known as Hypertext Markup Language, is document-layout. It defines the syntax and placement of special embedded directions not displayed by the browser. Still, it tells it how to display the document's content, including text, images, and other support media.[1] HTML help to create a user interface in the browser.

3.1.2 JavaScript

JavaScript is an object-based language that uses prototype objects to model inheritance. Using JavaScript to create a dynamic page.

3.1.3 CSS

CSS is cascading style sheets used to style elements making an attractive page. CSS describes how it should render elements on the screen, paper, speech, or other media. Use CSS because HTML and JavaScript are not enough to have a user-friendly website. With CSS, I can style the website as I want.

3.1.4 Blazor

Blazor lets you build interactive web UIs using C instead of JavaScript. Blazor apps are composed of reusable web UI components implemented using C#, HTML, and CSS. Both client and server code is written in C#, allowing you to share code and libraries. Blazor is a feature of ASP.NET, the popular web development framework that extends the .NET developer platform with tools and libraries for building web apps. [2] Many frameworks help to create a user interface, such as reactjs, vuejs, etc. However, I chose blazorise because it is a new technology, using C#, which is also the language I write backend. In this research, I use blazor Web Assembly. Many benefits come with using Web Assembly and, therefore, high-level programming languages in client-side Web development, such as ease of development and high performance of web applications. Web Assembly takes advantage of current hardware capabilities to ensure the best speeds possible. Another benefit is that it provides client-side security. This is one of the weaknesses of JavaScript. Web Assembly provides better security than JavaScript by acting as both the backend and frontend. Four major browsers currently support WebAssembly: Chrome, Safari, Firefox, and Edge. We envision WebAssembly will gain more popularity in client-side web development in the near future.[3]

3.1.5 Abp framework

ABP Framework offers an opinionated architecture to build enterprise software solutions with best practices on top of the .NET and the ASP.NET Core platforms. It provides the fundamental infrastructure, production-ready startup templates, modules, themes, tooling, guides, and documentation to implement that architecture properly and automate the details and repetitive works as much as possible. ABP Framework can work with any UI framework such as Angular, MVC page razor, and blazor. It can work with any database provider like entity framework core, MongoDB, and dapper. I implement domain-driven design architecture. There are a lot of features provided by the ABP Framework to achieve real-world scenarios easier, like Event Bus, Background Job System, Audit Logging, BLOB Storing, Data Seeding, and Data Filtering. [4] Abp is so suitable for micro-service, which is a technique applied in software that has an amount of user access simultaneously.

3.1.6 MongoDB

Because I chose abp framework, it only works with three database provider entity frameworks core, MongoDB, and dapper. My problem is e-commerce, so I need a high database speed read data and more scalability. MongoDB is a No SQL database that stores data in JSON-like documents with dynamic schema. Therefore, it simplifies data integration and offers better scalability than traditional relational databases.

3.1.7 Elastic search

E-commerce will have millions of users and expect more than 100 suppliers. The system requires many users to want to get data at the same time with a good response time. The response time is not guaranteed if I only use the database to store. Elastic search is an open-source search engine built on top of Apache Lucene, a full-text search library. Lucene is arguably the most advanced, high-performance, fully featured search engine library today- both open source and proprietary. Elastic search provides a distributed real-time document store where every field is indexed and searchable, distributed search engine with real-time analytics, capable of scaling hundreds of servers and petabytes of structured and unstructured data. [5]

3.1.8 RabbitMQ

As mentioned before, this project uses microservice architecture. To communicate among microservice, I use the message to send. When I have a lot of notes at the same time, how can I control it? I found that RabbitMQ is the solution. RabbitMQ is one of the most popular open source message brokers. Message brokers is

an intermediary program designed to validate, transform and route messages. It is to serve the communication among services. RabbitMQ helps the service send a response for each request quickly instead of running a procedure that takes resources of the system. Pushing messages in a queue is a solution when I want to distribute the message to many receivers and reduce work for workers. At the same time, so many suppliers click to submit to request to be suppliers. So many requests sent to the server will cause problems such as slowness, overload, congestion, etc We need RabbitMq to push requests in queue in the mechanism in figure 3.1:

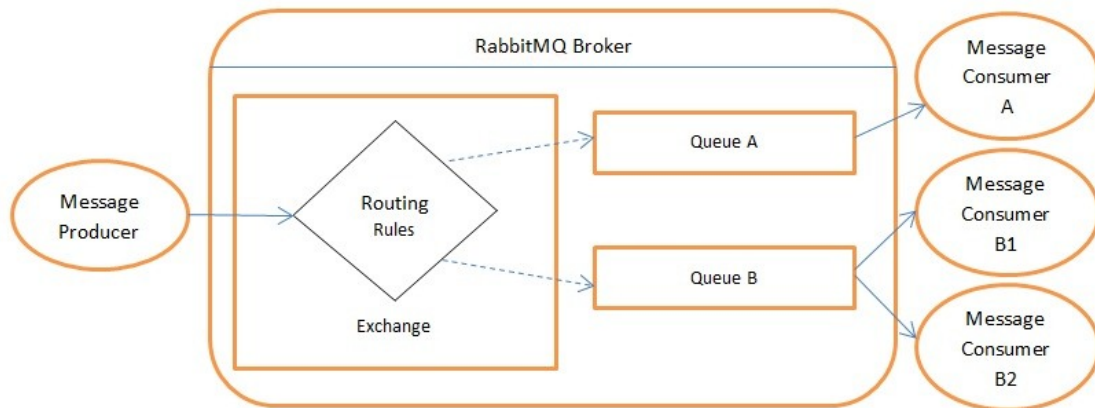


Figure 3.1: Messaging in rabbitMQ

Message producer publishes many requests, rabbitMQ exchanges them and pushes them into an appropriate queue. Finally, consumers receive a message in the queue to process.

3.1.9 Redis

Redis, a remote dictionary server, is an open source to save structure data and can be used as a database, cache, or message broker. It protects data in structure key-value in RAM, arranges, queries, and backups data in disk, helping backup data when the system is in trouble.

Redis can be used as cache, counter, publish/subscriber, and queues. Because of the speed of reading, data can be used as a cache, sharing data between services or temporary databases. It can be used as a counter. Redis supports thread-safe so that it can be synchronous among requests. Redis support creates a channel to exchange data between publisher and subscriber. Finally, it is used to create a queue for requests. For this research, I use Redis to create a queue and as a cache to increase read data speed.

3.1.10 S3

S3 stands for Amazon's simple storage service. S3 provides infinite scalability

and high availability at a low cost. Currently, S3 is used mostly to store multi-media documents (videos, photos, audio) shared by a community of people and rarely updated. This paper aims to demonstrate the opportunities and limitations of using S3 as a storage system for general-purpose database applications involving small objects and frequent updates. Read, write, and commit protocols are presented. Furthermore, such a storage system's cost, performance, and consistency properties are studied. [6]

CHAPTER 4. EXPERIMENT AND EVALUATION

4.1 Architecture design

4.1.1 Software architecture selection

For software architecture, I chose microservices architecture. Micro-services is a software system development method in which different single-function applications are coupled together. Because the system expects thousands of users to access the system simultaneously, with massive data, it is easy to be stuck, falling down the server at any time. Using monolithic architecture requires massive super storage and high configuration to guarantee a maximum response time of fewer than 3 seconds. The advantage of microservices is that each service can be scaled independently without disrupting the others. Because it is a distributed system, a microservices framework is highly scalable and can avoid the bottlenecks of a monolithic architecture.

The system has many microservices: auth service, sales service, partnership service, warehouse service, shipping service, etc., and one gateway called web Gateway service. Each microservice is build base on domain-driven design principles and patterns. Domain-driven design principles are new architecture that includes 3 layers:

- Domain layer : includes business objects and the core (domain) business rules. This is the heart of the application
- Application layer: mediates between the Presentation and Domain Layers. Orchestrates business objects to perform specific application tasks. Implements use cases as the application logic.
- Infrastructure layer: provides generic technical capabilities that support higher layers mostly using 3rd-party libraries.

DDD is mostly interested in the Domain and the Application layers rather than the Infrastructure and the Presentation layers. Apply architecture, the project had a little change follow abp framework. The domain layer in this research includes Domain, Domain.Shared, DbMigrator, MongoDB project. Dbmigrator layer to migrate database and seed data if needed. MongoDB project has context to connect to database and repository interface. The repository is an intermediary layer to getting data in an entity. This layer reduces the dependence of the business model on storage technology and enhances the ability to upgrade and replace technology when necessary. Domain project includes entity and implementation of repository inter-

face in MongoDB project. Domain shared has constant, enum,.. that all layers can access. The application layer contains Application and Application.Contract. The http layer includes Http.Api, Http.Client, Http.Host. These layers are to provide an environment for the client to call remote HTTP services.

4.1.2 Overall design

The UML package diagram of subsystem is shown in figure 4.1

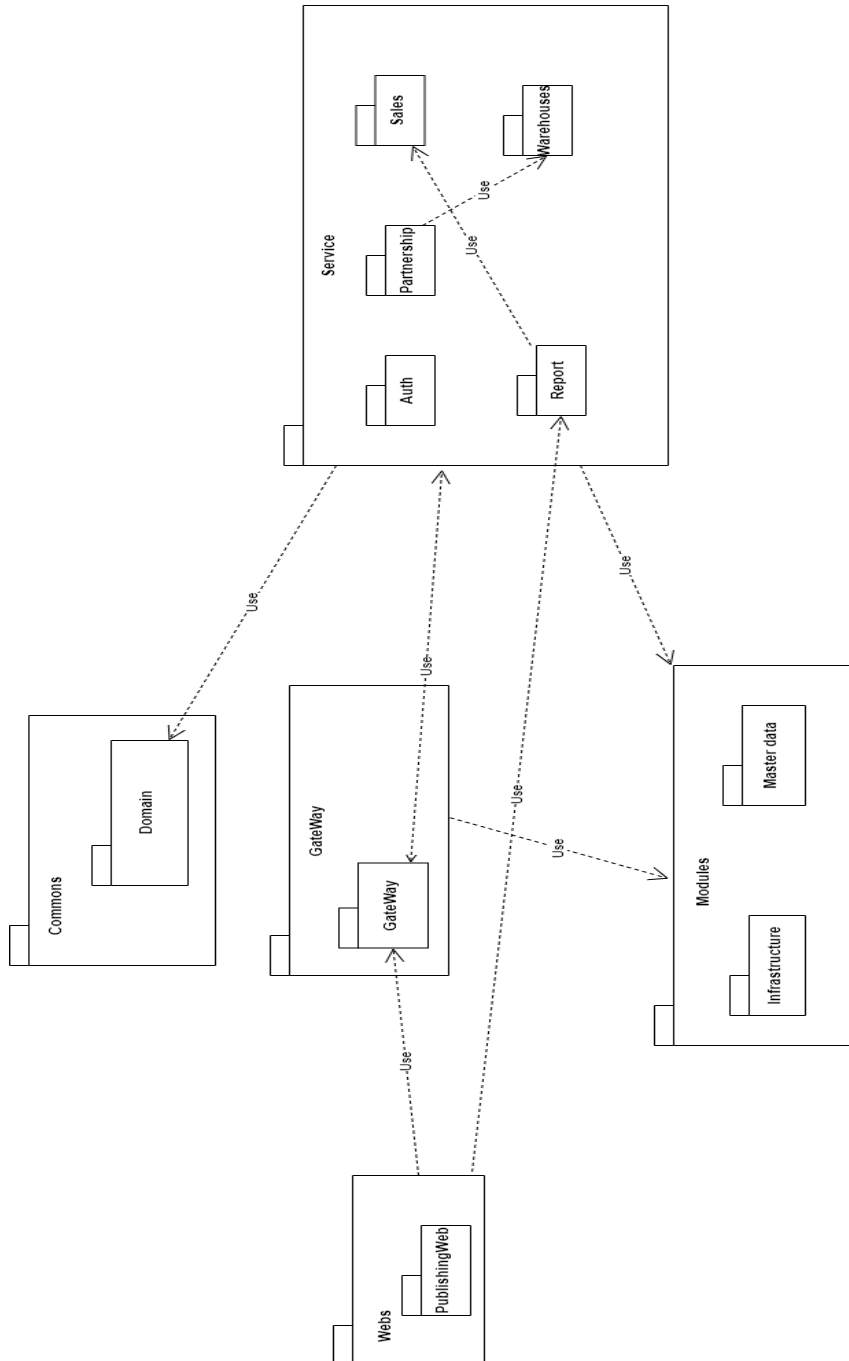


Figure 4.1: UML package diagram of subsystem

Package web has 1 package publishing web which contain the project of user interface application. Publishing web dependence with webbff package in gateway

and report package in services.

Commons package as its name, it contains everything that can be reused by services. It includes 2 packages: Application and Domains. Applications contain statistical classes in common and Domains contain classes in common.

The modules package contains two modules: infrastructure and master data. In addition, infrastructure contains modules like elastic search modules, clock modules, helper modules, and mass transit. Master data contains projects for geometries and setting of system.

Services contain all microservices in the system: authentic, partnership, warehouse, sales, shipping, reviews, and report service. The responsibility of each service is with a specific business. Auth service is to authenticate user and server. There is a list of servers allowed to access the system, defined by the setting. Auth service validates the user account information and generates and saves tokens from enabling the client to access the system. Partnership service responsibility is on the business of the actor's information such as register to be the supplier and approved supplier, register to be a distributor and approve distributor, etc. Warehouse management is the most important part of warehouse service. It is responsibility with the business related to managing warehouse and stock inventory, and assurance stock inventory is always enough for all orders accepted. Shipping service in this project is quite simple with the business of user's shipping address and change state of order to shipping. Perform order-related tasks is in sales service.

In gateway package contains webbff package which works as a gateway for system. It depends on all microservices in services package.

Each microservice has similarity UML diagram and relation to other packages. Figure 4.2 is sample UML package diagram of service reviews:

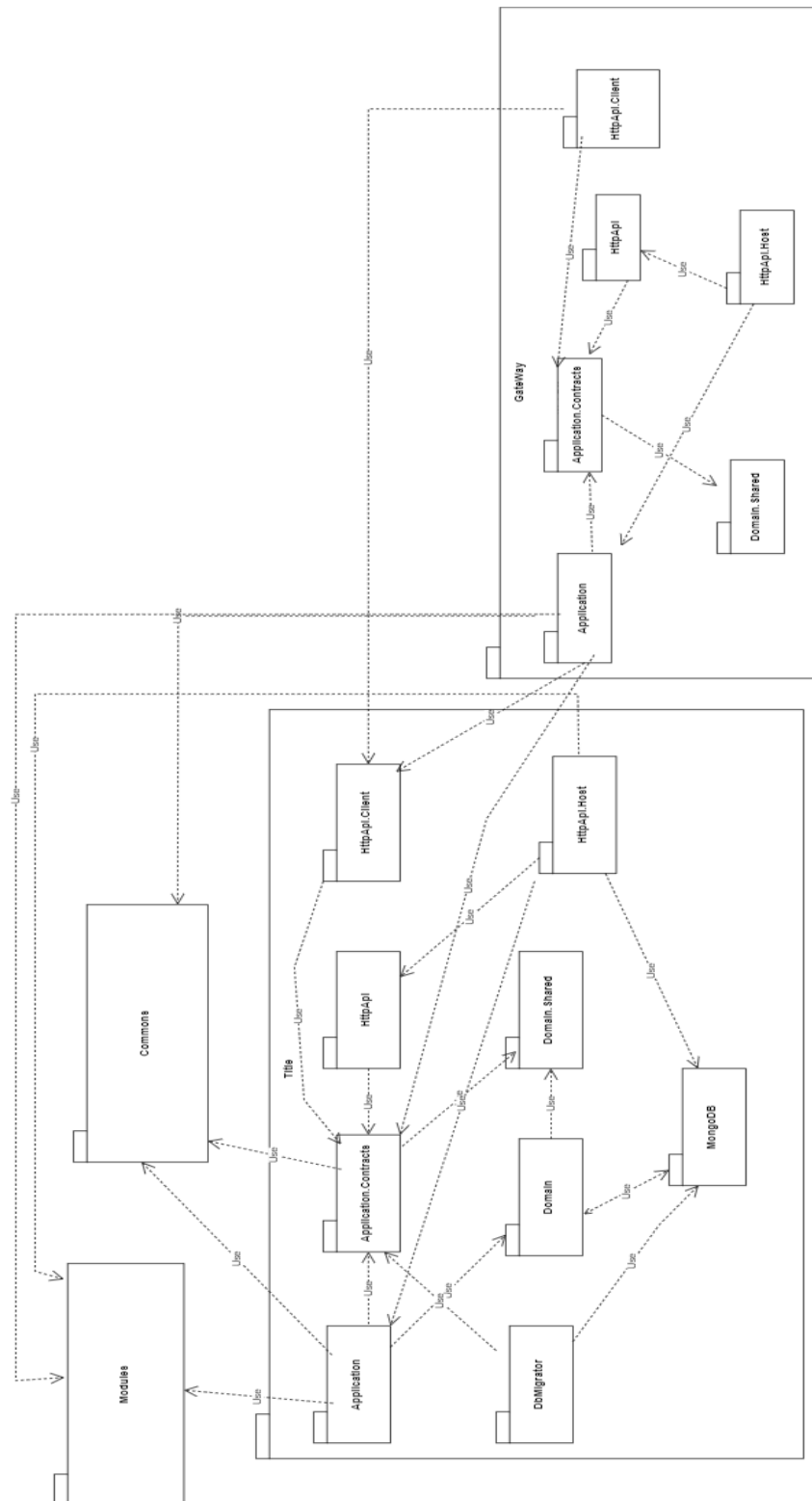


Figure 4.2: UML package diagram of 1 micro service

Application in webbff depends on application contract of micro services if it need. While The Http.Client in webbff depend on all http.client of micro service. In each micro service application package depend on common, modules if it need to use any class in these package.

In each package of micro service, there are 9 small package. These relations package is created base on domain driven design principle and pattern of abp framework. Domain layer include domain share, domain package, DbMigrator and MongoDB. Domain share contains constants, enum or other type can be safely share with all layers. This package can also be shared to 3rd-party clients. Domain package contains domain service interfaces and domain objects. Domain package depends on the Domain.Shared package. Dbmigrator is to migrate database to synchronous object between objects in domains and in database. It also can be use to seed data. Mongoddb contain connection string to connect with database, and implementation of domain service interface.

Application layer includes Application.Contracts and Application packages. Application.Contracts contain contains application service interfaces and related data transfer objects. Application.Contracts depends on Domains.Shared package. Application package contains application service implementations. Application package depends on the Domain and the Application.Contracts packages.

Http.Client and Http.Api is in Http layer. Http.API package to develop a REST style HTTP API for service, depends on Application.Constracts. It contains controller for each application service (generally by implementing their interfaces). These controllers uses the application service interfaces to delegate the actions. It just configures routes, HTTP methods and other web related stuffs if needed. Http Api client to provide client services for the HTTP API package. Those client services implement application interfaces as clients to a remote endpoint.HTTP API Client package only depends on the Application.Contracts package. [4]

4.1.3 Detailed package design

Detailed package of use case request to be supplier is in figure 4.3 below:

I use Dependency Injection pattern for project, example in use case request to be supplier. As mention in overall design, all class in services of package Application implement services in Application.Contracts. Client in user interface application service use interfaces in Application.Contracts by using dynamic client proxy of abp framework instead of calling API thought http request.

Service SupplierRegistrationApplicationService used interface SupplierApproveRegisterResponse in command package, this interface is used in SupplierApproveRegisterRequestProxy. SupplierApproveRegisterRequestProxy extends RoutingSlipProxy, it implement build routing slip adding activities which implements IActivity interface. All requests in routing slip is in queue and they will be send to suitable consumer to handle request.

4.2 Detailed design

4.2.1 User interface design

Display:

Number of colors supported: 16,777,216 colors

Resolution: 1920 × 1080 pixels

Screen

- Location of standard buttons: At the bottom right (vertically) and in the middle (horizontally) of the frame.
- Location of the messages: middle of the screen..
- Display of the screen title: The title is located at the top left of the frame in the middle.
- Consistency in expression of alphanumeric numbers: comma for separator of thousand while strings only consist of characters, digits, commas, dots, spaces, underscores, and hyphen symbol

Control

- Size of the text: medium size (mostly 16px). Font: Averta,Regular. Color: #333333
- Input check process: Should check if it is empty or not. Next, check if the input is in the correct format or not

Below are some image of user interface design:

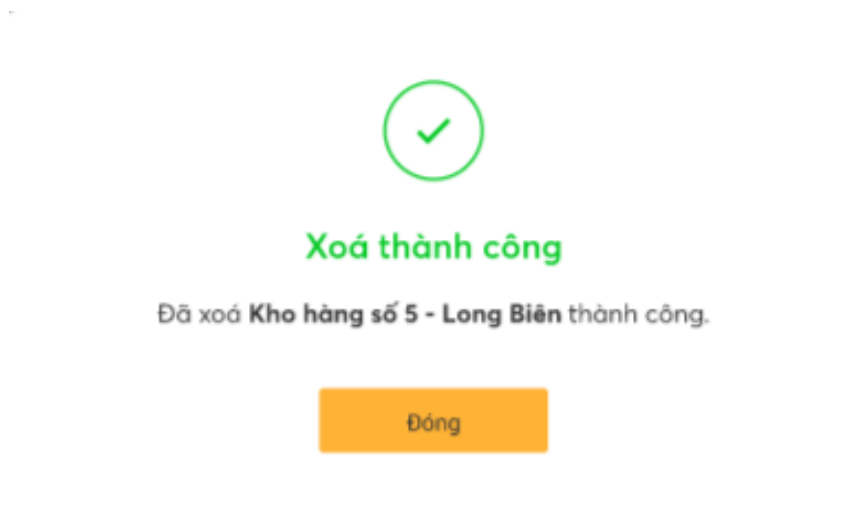


Figure 4.4: success message



Figure 4.5: warning message

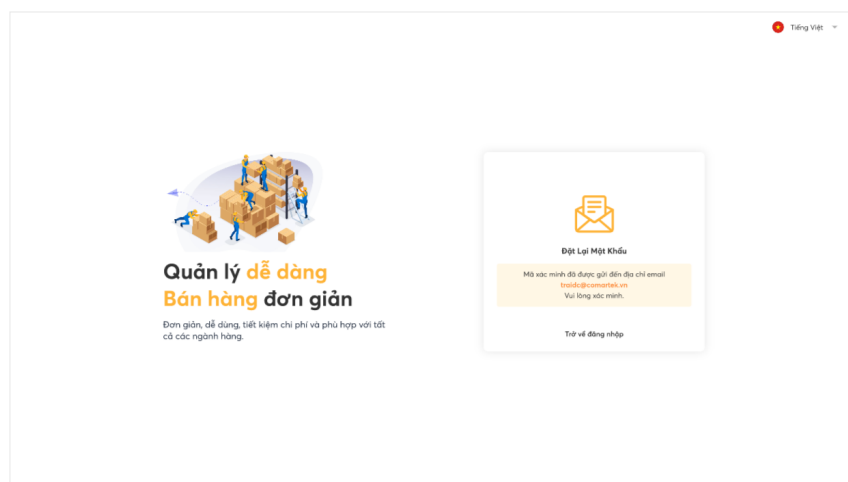


Figure 4.6: reset password page

4.2.2 Layer design

a, Sequence diagram of use case register to be supplier

Sequence diagram of use case register to be supplier is described in figure 4.7 below:

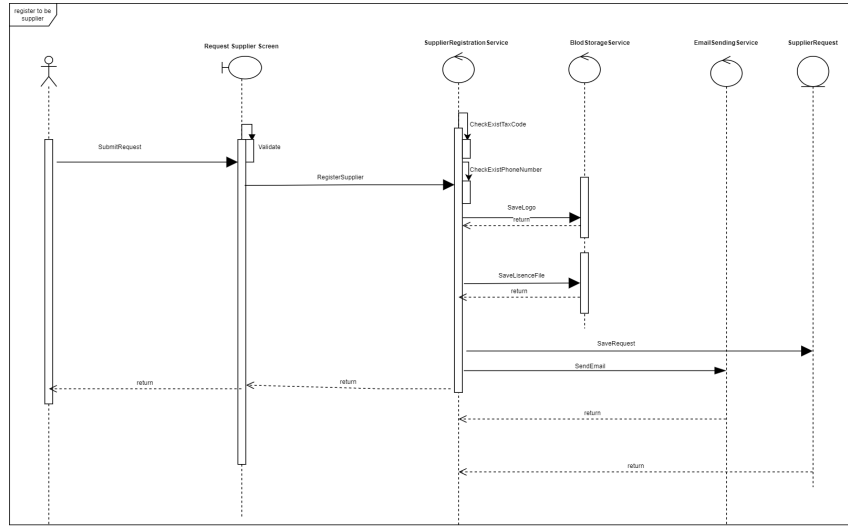


Figure 4.7: Sequence diagram of use case register to be supplier

Suppliers want to login into the system. They must have an account. To do that, they have to request to be suppliers by submitting the form register will all require information filled. If they lack some fields, the system will notify the error. If it is a valid form, the request will be sent to the server to check the existing email or tax code. If the system has existed email or tax code, it will notify the error so that supplier has to correct these invalid fields. If anything is good, a request to be supplier will be created, and an email notifying the request action of the supplier will be sent the email. That email contains the link for the supplier to follow the request. Administrators can view all requests, check information then accept or deny a request. If the request is received, the system will create a supplier record and an account in the database and synchronize supplier information to elastic search. All submissions will have the same tax code, and the email with that request will be rejected. After that system will send an email to a supplier with account information. If the admin denies the request, the system will email to notify the reasons for rejection and give the link to see the appeal. From this link, the supplier can update the request and request to be a supplier again.

b, Sequence diagram of use case manage warehouse

Suppliers will have many warehouses in different locations. Figure 4.8 below described how they can manage warehouses in system.

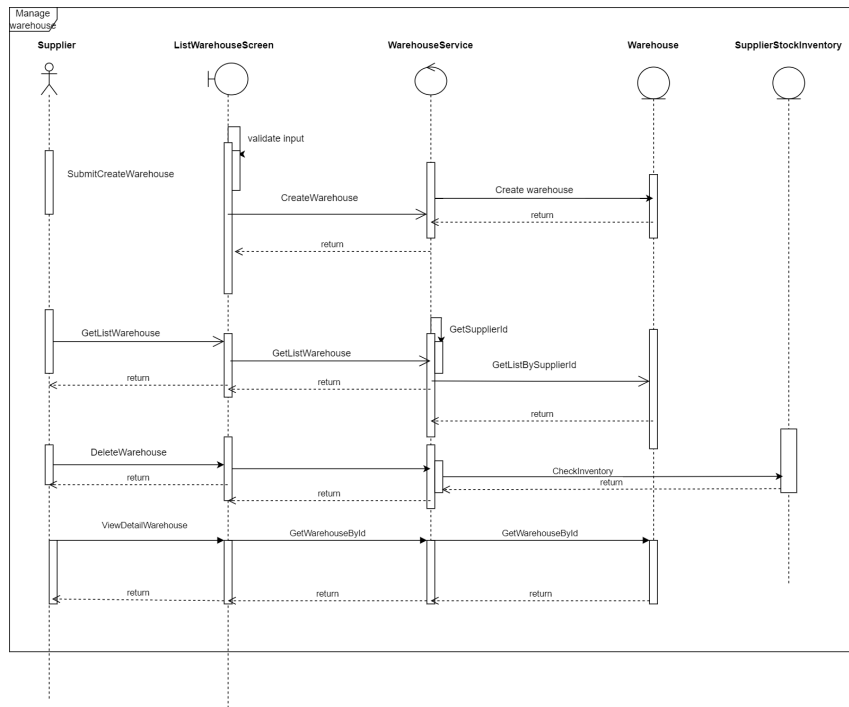


Figure 4.8: Sequence diagram of use case manage warehouse

The supplier can create a warehouse. When they submit the request, the system validates the input and notifies them if they have an error. If it is a valid form, the system will create a warehouse and inform the result of an action. Suppliers also can view the warehouse list. The system will take all warehouses of that supplier to show on the page. They can search warehouses by name and view detailed warehouse information. When suppliers want to delete the warehouse, they have to guarantee that the warehouse is empty. That means there are not any products in that warehouse. If not, the system will return an error.

c, Sequence diagram of use case update product

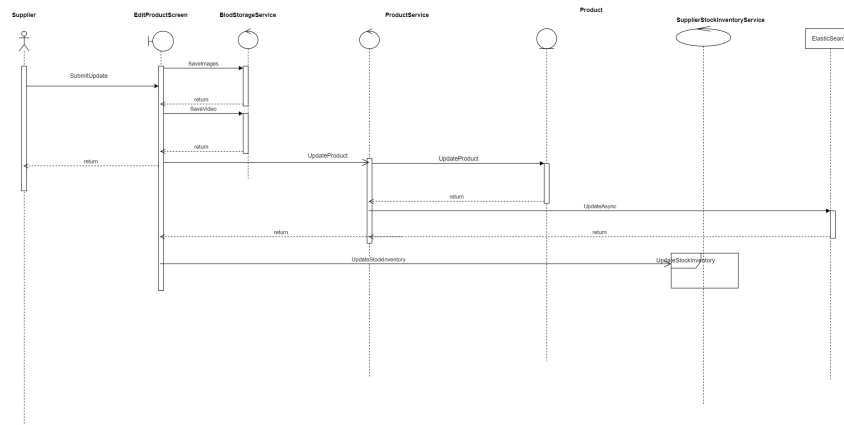


Figure 4.9: Sequence diagram of use case update product

Suppliers can update product as described in figure 4.9. When they request to update a product, the system will check if any new images or videos will upload to blob storage. After that, the system will update the product information in the database and elastic search. Finally, the system updates the stock inventory of that product. If the process has any error, the system will roll back all action and notify to user.

4.2.3 Database design

a, Service authentication

The database of authentication service I used the default of abp framework. In figure 4.10 I only show the most important table and usually be used.

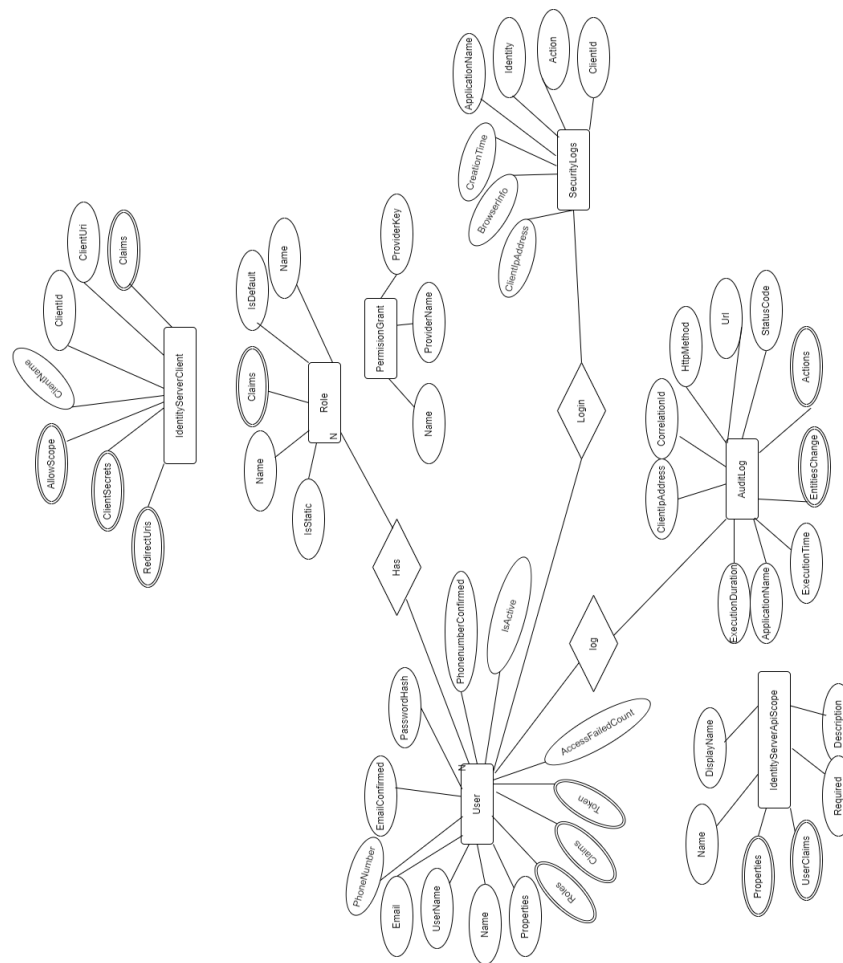
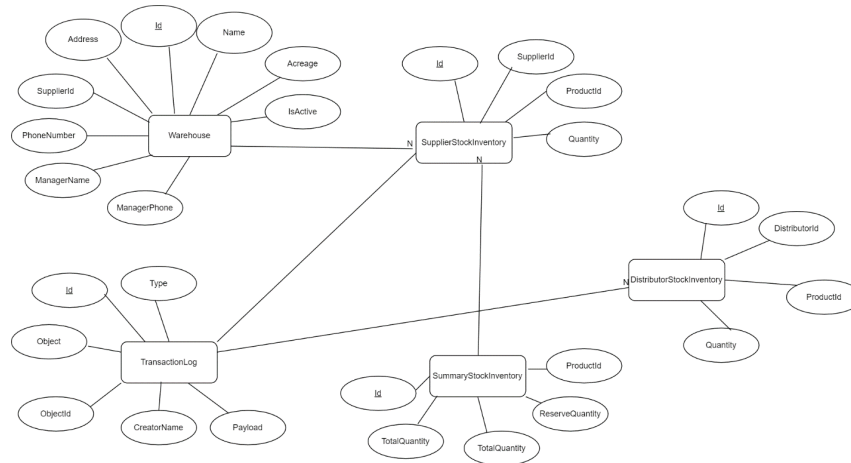


Figure 4.10: ER diagram of authentication service

Table Identity server client saves all servers allowed in the system. If another server is not in table access to the system, it will not be authorized. Table user saves account information; one user can have many roles. Each action of the user to an entity will be logged in the audit log when the user logs in into the system will be logged into the security log. Permission grants that list of permission in the admin

c, Service warehouse**Figure 4.12:** ER diagram of warehouse service

ER diagram of service warehouse is figure 4.12. Supplier stock inventory saves inventory of products in each warehouse. Distributor stock inventory is the warehouse of the distributor. When a change on inventory quantity will be logged in the transaction log. Summary stock inventory saves total product quantity in all warehouses.

d, Service partnership

ER diagram of partnership service is figure 4.13

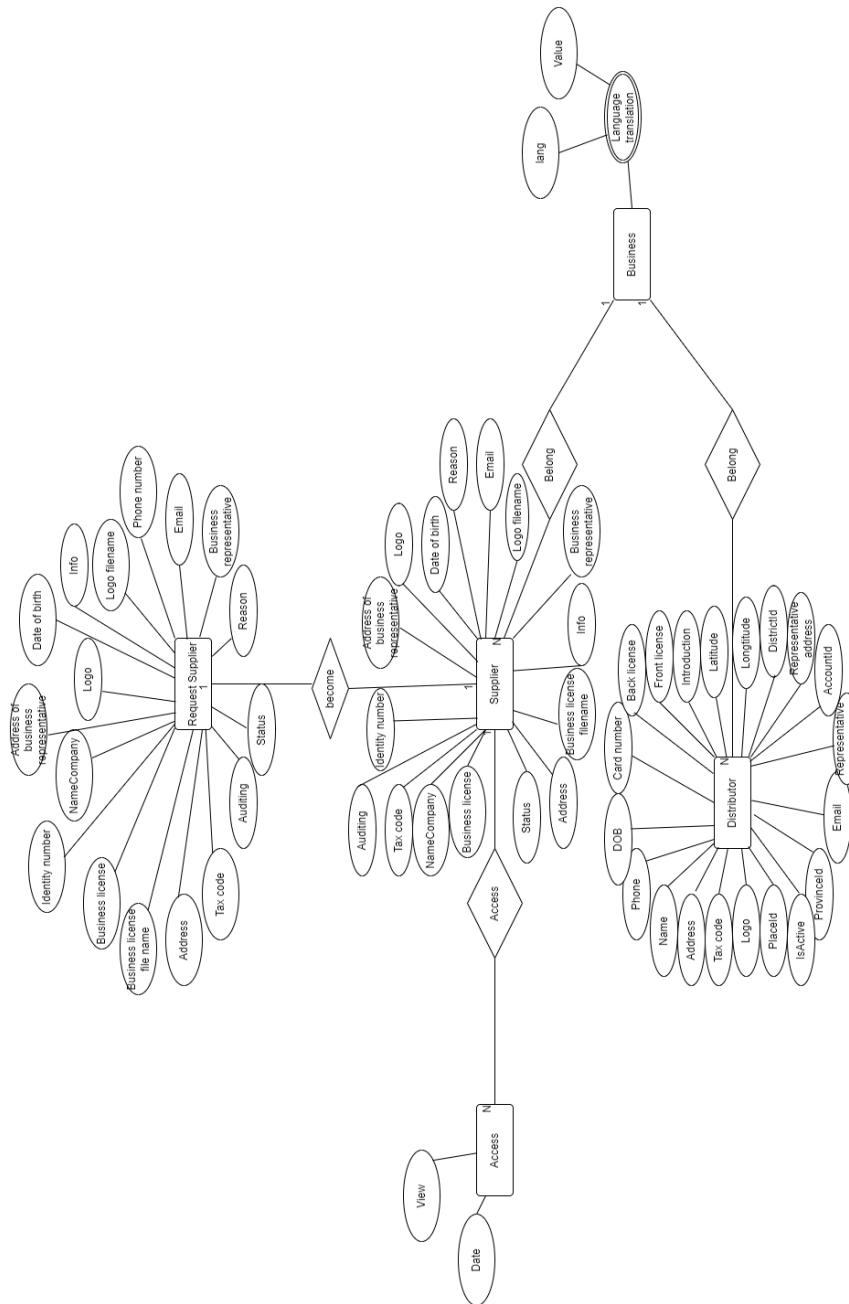


Figure 4.13: ER diagram of partner ship service

A supplier is created from a supplier request. Suppliers save supplier information. Distributors protect distributor information. Suppliers and distributors can have a business attribute. Access table saves users' access time in the supplier store group by date.

4.3 Application Building

4.3.1 Libraries and Tools

The libraries and tool is show in table4.1

Purpose	Tool	URL address
IDE coding	Visual studio 2022	https://visualstudio.microsoft.com/fr/vs/
Backend framework	ASP.Net core 6.0	https://www.microsoft.com/vi-vn
UI library	AntdBlazor v0.11.0	https://www.nuget.org/packages/AntDesign
UI library	Blazorise.Bootstrap5 v1.0.4	https://www.nuget.org/packages/Blazorise.Bootstrap5/1.0.4
Validate phone number	libphonenumber-csharp v8.12.45	https://www.nuget.org/packages/libphonenumber-csharp/8.12.45?src=template
Backend code	Abp package v5.1.4	https://www.nuget.org/packages/Volo.Abp
Generate excel file	NPOI v2.5.6	https://www.nuget.org/packages/NPOI
Mass transit	Masstransti package v7.3.1	https://www.nuget.org/packages/MassTransit
Elastic search	Nest package v7.12.1	https://www.nuget.org/packages/NEST/7.12.1?src=template
Send email	Volo.Abp.Mailkit v5.1.4	https://www.nuget.org/packages/Volo.Abp.MailKit/5.1.4
Support media storage	Volo.Abp.BlobStoring v5.1.4	https://www.nuget.org/packages/Volo.Abp.BlobStoring/5.1.4
Log	Serilog.AspNetCore v4.1.0	https://www.nuget.org/packages/Serilog.AspNetCore/4.1.0

Table 4.1: List of libraries and tools

4.3.2 Achievement

The result of this study is a website subsystem of an e-commerce platform on sales and purchase channels between suppliers and distributors, helping both to have optimal solutions in trade. In addition, this website also allows the parties to manage their resources and business activities on one platform.

This software application includes:

- Number of lines of code: 322329
- Number of executed line code: 85785
- Number of classes : 10473
- Number of interface : 182
- Number of packages :92
- Number of projects :64

4.3.3 Illustration of main functions

a, Login page

The login page is in figure 4.14

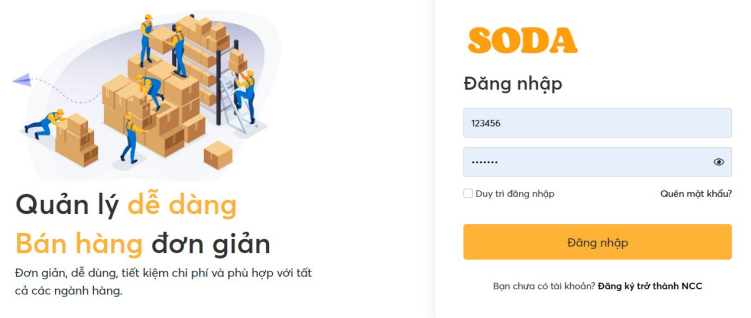


Figure 4.14: Login page

Screen specification of login page is in table 4.2 and define attribute fields in table 4.3

Control	Operation	Function
Area for input tax code	Required	Input the tax code
Area for input password	Required	Input the password
Button login	Initial	Direct to home page
Button register	Initial	Direct to register page
Button forgot password	Initial	Direct to forgot password page
Button remember me	Initial	Remember token to next access


Table 4.2: Screen specification of login page

Item name	Number of digits	Type
Taxcode	200	Text
Password	16	Text


Table 4.3: Define attribute field of login page

b, Register page

Register page is in figure 4.15, screen specification in table 4.4 and defined attribute fields in table 4.5 below:



Tiếng việt ▼ Đăng nhập



Mở tài khoản Nhà cung cấp trên Soda hoàn toàn Miễn phí

Tiếp cận tới hàng triệu khách hàng mọi lúc, mọi nơi!

Đăng Ký Trở Thành Nhà Cung Cấp

1. Hồ sơ công ty

Tên doanh nghiệp

Mô số thuế Logo (ảnh đại diện)

Địa chỉ

Số điện thoại Email

2. Thông tin người đại diện pháp luật

Tên người đại diện pháp luật

Ngày sinh Số CMT/CCCD

Địa chỉ

3. Thông tin thương hiệu kinh doanh

Nhóm ngành nghề kinh doanh Giấy phép kinh doanh

Giới thiệu về công ty, ngành nghề kinh doanh

ĐĂNG KÝ NGAY

Bằng việc đăng ký, bạn đã đọc và đồng ý với các [Điều khoản](#), [điều kiện](#) và [Chính sách bảo mật](#) của Soda

Figure 4.15: Register page

Control	Operation	Function
Area for input company information	Required	Input company information
Area for input representative information	Required	Input the representative information
Area for input business information	Optional	Input the business information
Button register	Initial	Direct to page after register successful

Table 4.4: Screen specification of register page

Item name	Number of digits	Type
Company name	200	Text
Tax code	16	Text of number only
Logo	3MB	Bytes
Company address	200	Text
Company hotline	200	Phone number
Email of company	200	Email
Representative name	200	Text
Date of birth		Date time
Representative phone number	12	Phone number
Representative email	200	Email
Business	32	Selection
Business Liscence	3MB	Bytes
Introduction	200	Text

Table 4.5: Define attribute field of register page**c, Create product page**

Create product page is in figure 4.16, screen specification in table 4.6 and table 4.7

Thêm mới sản phẩm

Ngành hàng *
Chọn ngành hàng

Tên sản phẩm *
Ví dụ: Áo thun nam Cotton Compact phiên bản Premium chống nhăn 0/200
Sử dụng tiếng Việt có dấu, không viết tắt, từ thiểu 10 ký tự. Độ dài cho tất cả tiêu đề là 200 ký tự (bao gồm cả khoảng trắng).

Mô tả
Normal B I U
Tối đa 2000 ký tự. Mô tả sản phẩm Shop là những thông tin giới thiệu đặc tính chất lượng, chức năng, lợi ích... của một sản phẩm được đăng tải trên sản phẩm mang đến cho Người mua các thông tin đầy đủ về sản phẩm để họ có cơ sở lựa chọn khi mua hàng của bạn.

Hình ảnh sản phẩm
Bạn có thể up nhiều hình ảnh cùng lúc và tối đa có thể có 8 hình. Hình ảnh có kích thước tối đa 5MB và không được phép chứa nội dung nhạy cảm.
Chọn file hoặc Kéo thả ảnh của bạn vào đây.

Video sản phẩm
Chọn file hoặc Kéo thả video của bạn vào đây.

Giá bán
Giá bán lẻ ☒
VD: 80.000
Giá nhà phân phối ☒
VD: 80.000 SL tối t...

Thông tin kho hàng (0 kho)
Tổng SL tồn kho: 0
Số lượng có thể bán: 0

Thông tin chi tiết
Thương hiệu *
No brand
Nhà sản xuất *
Nhà sản xuất
Hạn sử dụng
DD/MM/YY
Kích thước đóng gói (cm)*
0 x 0 x 0
Trọng lượng sản phẩm (Kg) *
0
Đơn vị (Chiếc, cái, thùng, kg...) *
Chiếc

1. Kích thước: Tối đa 5Mb
2. Độ dài: 10s-60s
3. Định dạng: MP4 (không hỗ trợ vpr)
4. Lưu ý: sản phẩm có thể hiển thị trong khi video đang được xử lý. Video sẽ tự động hiển thị sau khi đã xử lý thành công.

Figure 4.16: Create product page

Control	Operation	Function
Area for product information	Initial	Input product information
Area for input stock inventory information	Initial	Input the stock inventory information
Button cancel	Initial	Direct to previous page
Button Create	Initial	Direct to list product page

Table 4.6: Screen specification of create product page

Item name	Number of digits	Type
Category	32	Select
Product name	200	Text
Description	2000	Text
Images	3MB	Bytes
Video	5MB	Bytes
Price	10	Number
Distributor price	10	Number
Stock inventory	10	Number
Minimum distributor order	10	Number
Origin	200	Text
Brand	200	Text
Expired date		Date time
Volume	200	Text
Weight	10	Number

Table 4.7: Define attribute field of create product page**d, Create warehouse page**

Figure 4.16 is capture of create warehouse page, its screen specification is in table 4.8

The screenshot shows a web application interface for adding a new warehouse. On the left, there is a sidebar with navigation links: 'Sản phẩm' (Products), 'Kho hàng' (Warehouses), 'Đơn hàng' (Orders), 'Đại lý' (Dealers), 'Nhà phân phối' (Distributors), 'Khách hàng' (Customers), and 'Báo cáo' (Reports). The 'Kho hàng' section is selected, showing a list of warehouses with a 'Thêm mới kho hàng' button. The main content area displays the 'Thêm mới kho hàng' form. The form has a title bar with a back arrow. Below the title bar, there are several input fields: 'Tên kho hàng' (required), 'Địa chỉ' (required), 'Diện tích kho' (optional), 'Tên người quản lý' (optional), 'Số điện thoại' (optional), and 'Trạng thái kho hàng' (optional). Each field has a placeholder text indicating what to enter. At the bottom right, there are two buttons: 'Hủy bỏ' (Cancel) and 'Thêm mới' (Add new).

Figure 4.17: Create warehouse page

Control	Operation	Function
Input warehouse name	Required	Input warehouse name
Input address	Required	Warehouse address
Area	Optional	Warehouse 's area
Manager name	Optional	Name of manager
Manager phone number	Optional	Phone number of manager
Manager name	Optional	Name of manager
Status	Optional	Status of warehouse
Button create	Initial	Redirect to list warehouse page
Button cancel	Initial	Close dialog

Table 4.8: Screen specification of create warehouse page

e, List order page

Figure 4.18 is display user interface of list order page, its screen specification is in table 4.9 and attribute field defined in table 4.10 below

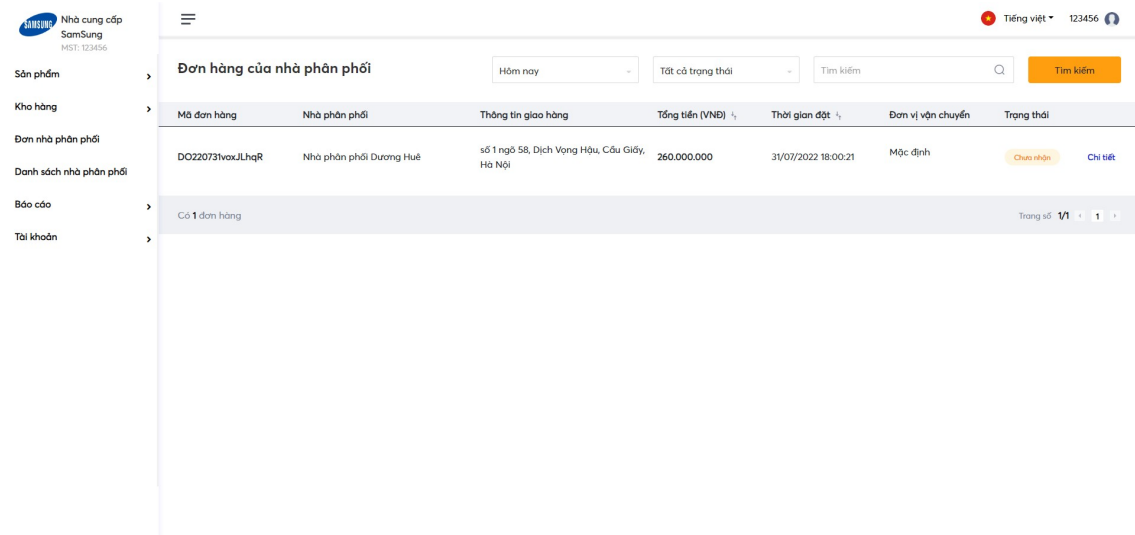


Figure 4.18: List order page

Control	Operation	Function
Area for order list	Initial	List of order
Area for filter options	Initial	Filter options
Filter button	Initial	Get new list order by filter
View button	Initial	Direct to detail order page
Area for pagination	Initial	Pagination

Table 4.9: Screen specification of list order page

Item name	Number of digits	Type
Time selection		Select date time
Search input	200	Text
Status order selection		Select

Table 4.10: Define attribute field of list order page

f, Order detail page

The screenshot displays the 'Chi tiết đơn hàng' (Order Detail) page for an unconfirmed order. The page is divided into several sections:

- Header:** Includes the SamSung logo, user information (Tiếng Việt, 123456), and navigation links (Sản phẩm, Kho hàng, Đơn nhà phân phối, Danh sách nhà phân phối, Báo cáo, Tài khoản).
- Order Information:**
 - Đơn hàng nhà phân phối > Chi tiết đơn hàng
 - Mã đơn hàng: DO220731voxJLhqR
 - Thời gian đặt hàng: 31/07/2022 18:00:21
 - Trạng thái đơn hàng: **Đơn hàng đang chờ duyệt**
 - Xuất hóa đơn: C6
 - Công ty: Công ty in hồ guom
 - MST: 1212121
 - Email: hoGuom@gmail.com
- Thông tin nhà phân phối:**
 - Tên nhà phân phối: Nhà phân phối Dương Huệ
 - Số điện thoại: 45678
 - Email: distributor@gmail.com
 - Mô tả: Địa chỉ: số 1 Duy Tân, Phường Dịch Vọng, Quận Cầu Giấy, Hà Nội
- Thông tin giao hàng:**
 - Tên người nhận: Hạnh Phạm
 - Số điện thoại: 0904566982
 - Địa chỉ giao hàng: số 1 ngõ 58, Dịch Vọng Hậu, Cầu Giấy, Hà Nội
 - Phương thức thanh toán: Thanh toán khi nhận hàng
- Thông tin thanh toán:**
 - Tiền hàng: 260.000.000
 - Phí vận chuyển: 0
 - Tổng cộng: 260.000.000
- Thông tin sản phẩm:**

SẢN PHẨM	MÃ SẢN PHẨM	NGÀNH HÀNG	SỐ LƯỢNG	ĐƠN GIÁ (VNĐ)	THÀNH TIỀN (VNĐ)	
	Tủ lạnh sam sung MC123 rất đẹp	Máy tính	20	13.000.000	260.000.000	Chọn kho hàng (0/1) Nhập nhanh 0/20

Figure 4.19: Detail of unconfirmed order page

Control	Operation	Function
Area for display general order information	Initial	General order information
Area for display distributor order	Initial	Distributor information
Area for display shipment detail	Initial	Delivery detail
Area for display payment detail	Initial	Payment detail
Area for display order detail	Initial	List items in order
Button cancel	Initial	Cancel order
Button confirm	Initial	Confirm order
Button chose warehouse	Initial	Direct to chose warehouse page
Button quick chose warehouse	Initial	Display dialog chose warehouse

Table 4.11: Screen specification of unconfirmed order detail page

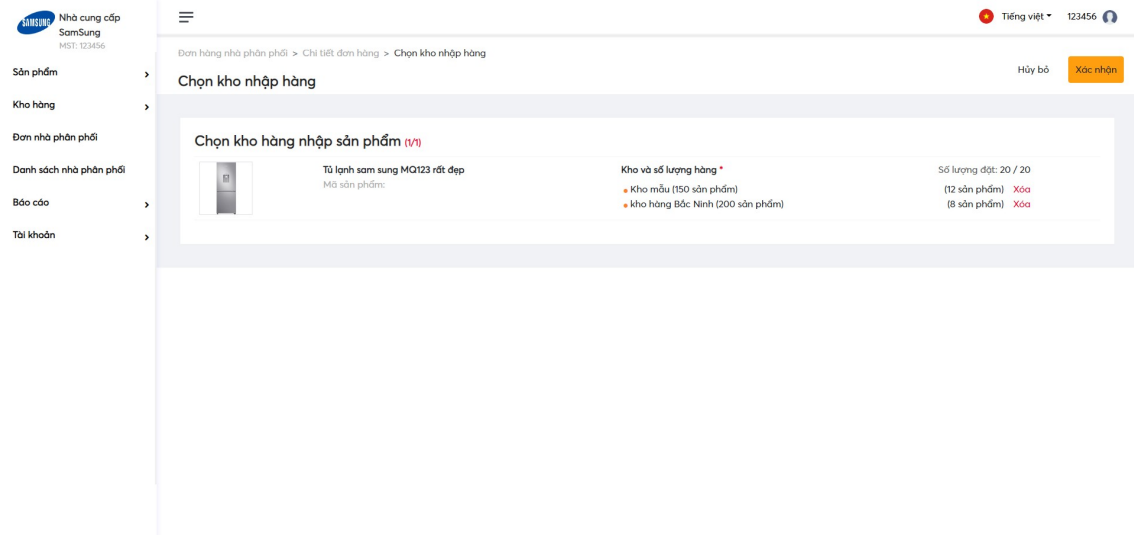


Figure 4.20: Chose warehouse page

Control	Operation	Function
Area for display order detail	Initial	General order information
Chose warehouse button	Initial	Distributor information
Button cancel	Initial	Direct to order detail page
Button confirm	Initial	Direct to order detail page
Button delete	Initial	Remove chosen

Table 4.12: Screen specification of chose warehouse in order detail page

g, Report growth page

Report growth page is capture in figure 4.21, table 4.13 is its screen specification

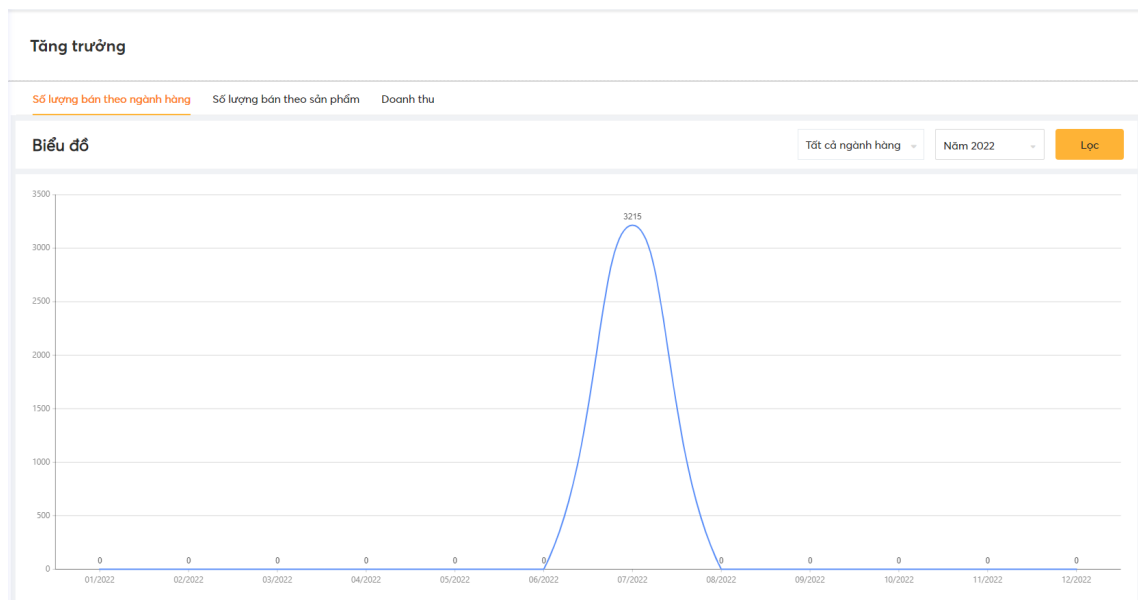


Figure 4.21: Report growth page

Control	Operation	Function
Area for report by category	Initial	Graph of growth by category
Area for report by product	Initial	Graph of growth by product
Area for report by revenue	Initial	Graph of growth by revenue
Area for filter option	Initial	Select category and time
Filter button	Initial	Return graph data by filter

Table 4.13: Screen specification of report growth page

4.4 Testing

Testing result is in table 4.14

#	Functionality	Test	Pass
1	Create request to be supplier	4	3
2	Admin accept request of supplier	4	3
3	Admin deny request of supplier	3	3
4	Admin deny request of supplier	3	3
5	Admin deny request of supplier	3	3
6	Supplier create product	4	3
7	Supplier delete product	2	2
8	Supplier update product	2	2
9	Supplier accept a order	2	2

Table 4.14: Testing result

The test case does not pass because the network connection and server are down. My computer does not have enough performance to build all microservice at the same time. That leads server being down then response time is out of time.

4.5 Deployment

In reality, for deployment, the system needs at least two domains: gateway and user interface application. A server has storage of up to 80GB. The system needs to register some services of Elastic search, mongo cloud storage, and media storage.

CHAPTER 5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

Choosing this topic for graduation research is a challenge for me because I have not had any experience with thousand users, especially e-commerce systems. However, I always want to learn new things and new techniques. In the process of working on this research, I stuck a lot of time.

Firstly, the challenge of new techniques and the framework for front-end and back-end are new for me. They also are new techniques, especially blazor. The documents and the examples project of blazor are pretty small. I spent a lot of time reading the document and the framework's source code to understand it. I learned a lot of techniques when using abp frameworks, such as domain-driven design patterns, dependency injection patterns, singleton patterns, etc., and applied them to this project. I might not learn these things if I had chosen the framework in graduation research one and two.

Secondly, I have no experience working with NoSQL, especially MongoDB. As you know, mongo DB stores data in JSON-like documents. It is so different from the SQL server I have learned at this university. The design database for the SQL server is different from the database in MongoDB too. I had changed the data model so many times to get the final version. The difference in a query put the work at a standstill. However, after searching and reading the documentation, I finally got the solutions.

In graduation research 2, I stored the image in google drive, then got the public URL to return to the website. But this research, I cannot use the old way because the storage of google drive of an account is limited. It is not high security with an e-commerce system. I thought I should have a storage system to store the media. Amazon service solved my problem. I found s3, which is a service of amazon to storage media. Amazon S3 supports both server-side encryption and client-side encryption for data uploads. Amazon S3 offers flexible security features to block unauthorized users from accessing your data.

After learning distributed system course and reading the requirement of all e-commerce systems, I know that this system must be in a microservice architecture. However, this is the first time of mine worked with a micro-service. It takes me a lot of time to make the gateway work. The problem is communications among micro-services. How can I take data from another service? The gateway in this

project also is responsible for aggregating data from micro-services to return to the client. However, when the transaction requires other micro-services works, for example, in the case of accepting a supplier's order, it requires the sale service to change the order state and the warehouse service to update the reserve quantity in each stock. Abp framework support event bus combined with routing slip of mass transit, so that from sale service, the system can send message to warehouse to update reserve quantity in one transaction. If one event fails, a transaction will roll back. I can follow the queue of the message by using rabbitmq.

When the System has many redundant data such as in table supplier request. In case requests are no longer to approved because of some reasons. What should I do to save admin work? I thought, I will reject all request if exist a setting time. After searching, I decided to use background job. The framework provide the class HangfireBackgroundWorkerBase so that I can create my worker inherited it. Then register this service to module of partnership host. It is a first time of mine working with it. I faced with simple case so that it is pretty good.

I had no idea face with the requirement of response time for reading and searching data. The system is for Vietnamese people. Therefore it should allow search words with no accents. In addition, the search time should be fast as possible. Only using MongoDB can not solve this problem. However, I remembered the critical word "elastic search" when I learned machine learning course. After reading the document, this technique will solve the problem of searching data both in response time and with no accent words. Despite that, elastic search require its library to read and write data. That leads to a problem with using the new library .NET. Until now, the system can get, write, read, and search data in elastic search, but I felt that I had just discovered a small part of elastic search.

By passing all the difficulties when working in this research, the result is a sub-system of the supplier in an e-commerce system. The software is created based on similar software, updated following specific businesses requirement. These sub-systems help supplier manage their resource, orders, and views of the report. It has great significance in management resources.

During the process of working in this research, I learned many valuable lessons. Firstly, I applied theoretical knowledge from courses in the Hanoi university of science and technology to the actual software. I had a deeper understanding of what had been learned. Secondly, I learned a lot of new techniques and improved my reading skill, my ability to work independently, and my ability to explore and create.

5.2 Future work

In the future, to complete the subsystem of the supplier, firstly, the project should be integrated with a shipping system and payment system to complete the life cycle from the distributor placing an order to the supplier receiving the money for that order. Secondly, the project should extend the warehouse service into a warehouse system to track each product. The warehouse system helps reduce the situation of buying and selling counterfeit goods. Moreover, for e-commerce, subsystems for distributors, retailers, and customers should be complete in the future.

REFERENCE

- [1] C. Musciano and B. Kennedy, *HTML & XHTML: The Definitive Guide: The Definitive Guide*. " O'Reilly Media, Inc.", 2002.
- [2] *Interactive web ui with c*. [Online]. Available: <https://dotnet.microsoft.com/en-us/apps/aspnet/web-apps/blazor>.
- [3] N. Burkhart, W. Liao, and O. Guzide, "An overview of webassembly," *Proceedings of the West Virginia Academy of Science*, vol. 92, no. 1, 2020.
- [4] A. E. Halil İbrahim Kalkan, *Abp documentation*. [Online]. Available: <https://docs.abp.io/en/abp>.
- [5] C. Gormley and Z. Tong, *Elasticsearch: the definitive guide: a distributed real-time search and analytics engine*. " O'Reilly Media, Inc.", 2015.
- [6] M. Brantner, D. Florescu, D. Graf, D. Kossmann, and T. Kraska, "Building a database on s3," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 251–264.