**Ministry of Education and Training**

**Saigon University**

-- 🙢🕮🙠--



**Report: Midterm Project**

**Subject: Software Engineering**

**Topic: Sales Management Application**

Student group implementation:

|  |  |
| --- | --- |
| **MSSV** | **Student's name** |
| 3121411226 | Trần Quang Vinh |
| 3121411100 | Phạm Quang Khiêm |
| 3121411045 | Đặng Nguyễn Quốc Dương |
| 3121411060 | Nguyễn Trí Đức |

Instructor: *Mr. Đỗ Như Tài*

**Hồ Chí Minh city, May 20, 2025**

**Table of Contents**

[Menu of Image VIII](#_Toc4520)

[Chapter I : Topic Overview 1](#_Toc16009)

[1.1. Survey the current status of the system 1](#_Toc3558)

[1.1.1. Internal 1](#_Toc29696)

[a) Internal organizational structure. 1](#_Toc655)

[b) Internal organizational structure chart 1](#_Toc7160)

[1.1.2. Foreign Affairs 2](#_Toc10605)

[a) Organization and its environment in terms of external relations 2](#_Toc14638)

[b) The business environment affects the software. 3](#_Toc20984)

[1.2. Business process 4](#_Toc7720)

[Business from a manager's perspective 4](#_Toc27107)

[1.3. Functional requirement specification 7](#_Toc23040)

[1.3.1. Customer account management function 7](#_Toc9006)

[1.3.2.Category and product management function 7](#_Toc196)

[Portfolio Management 7](#_Toc27539)

[Product Management 8](#_Toc21406)

[1.3.3. Order processing function (Admin) 8](#_Toc18568)

[1.3.4. Import and warehouse management functions 9](#_Toc12119)

[1.3.5. Customer Email sending function 10](#_Toc1073)

[1.4. Non-functional requirements specification: 10](#_Toc7168)

[Performance: 10](#_Toc15086)

[Scalability 10](#_Toc28349)

[Security 11](#_Toc5752)

[Usability 11](#_Toc14084)

[Reliability 11](#_Toc10906)

[Maintainability & Updatability 11](#_Toc15064)

[Chapter II: System Analysis 13](#_Toc18705)

[2.1. System analysis overview diagram 13](#_Toc8638)

[2.1.1. Business process context level diagram 13](#_Toc3356)

[2.1.2. Entity Relationship Diagram (ERD) 14](#_Toc11396)

[2.1.3. Mapping table 15](#_Toc9399)

[2.2. Use Case: 16](#_Toc2063)

[2.2.1. Define Use Cs ase : 16](#_Toc19617)

[a) Main modules 16](#_Toc31692)

[b) Functional requirements - Use Case: 16](#_Toc23961)

[2.2.2. General Use Case Diagram: 19](#_Toc30841)

[2.2.3. Description of Use Cases: 20](#_Toc31860)

[a) UC Login : 20](#_Toc29278)

[b) UC Employee Management 20](#_Toc19455)

[c) UC Customer Management 21](#_Toc2609)

[e) UC Supplier Management 22](#_Toc14021)

[f) UC Laptop Type Management 22](#_Toc25148)

[g) UC Laptop Management 23](#_Toc22577)

[h) UC Warehouse 23](#_Toc10614)

[i) UC Print invoice 24](#_Toc1962)

[k) UC Statistics 24](#_Toc29877)

[2.3. Use case analysis : 25](#_Toc23957)

[2.3.1. Functional design analysis (UC – AC): 25](#_Toc32290)

[a) Login 25](#_Toc2403)

[b) Customer Management 28](#_Toc26004)

[c) Manage Laptop catalog 32](#_Toc18711)

[d) Supplier Management 36](#_Toc8319)

[e) Laptop Type Management 40](#_Toc5602)

[f) Laptop Management 44](#_Toc32407)

[g) Warehouse 48](#_Toc30413)

[h) Print invoices 51](#_Toc4187)

[i) Statistics 53](#_Toc12683)

[Chapter III: Software development process (Agile process) 55](#_Toc26402)

[3.1. Overview of Agile process 55](#_Toc4827)

[3.1.1. Concept 55](#_Toc8799)

[3.1.2. Source 55](#_Toc4920)

[3.1.3. Characteristics 56](#_Toc7221)

[3.2. Agile process activity diagram 57](#_Toc405)

[3.3. Applying Agile process to Sales Management software (Admin section) 58](#_Toc22428)

[3.3.1. Initiation Phase – Building the Product Backlog 58](#_Toc28032)

[3.3.2. Sprint Planning – Sprint 1 Planning 58](#_Toc9372)

[3.3.3. Sprint Implementation (Iterative Development) 59](#_Toc4836)

[3.3.4. Daily Scrum – 15-minute meeting every day 60](#_Toc21351)

[3.3.5. Sprint Review – Present results to PO 60](#_Toc23865)

[3.3.6. Sprint Retrospective – Process Improvement 60](#_Toc21861)

[3.3.7. Plan for the next Sprints 61](#_Toc1259)

[3.3.8. Sprint 5: Create goods receipt and update goods to the store. 61](#_Toc25447)

[3.3.9. Sprint 6: Process customer orders and create delivery notes 62](#_Toc8507)

[3.3.10. Sprint 7: Advanced Statistics and Reporting 62](#_Toc14999)

[3.3.11. Agile Support Tools 63](#_Toc6634)

[3. 4 . Applying Agile Processes to Sales Management Software 63](#_Toc19948)

[3.4.1. Sprint 1: Shopping Cart and Order 63](#_Toc32615)

[3. 4 .2. Sprint 2: Confirm order and send email 64](#_Toc30753)

[3. 5 . Advantages and disadvantages of Agile model 65](#_Toc17762)

[Chapter IV : Software Architecture and Data Modeling 66](#_Toc23170)

[4.1. Software architecture 66](#_Toc31464)

[4.1.1. Java programming language 66](#_Toc14097)

[4.1.2. Layered Architecture 67](#_Toc13434)

[4.2. Data modeling 73](#_Toc18631)

[4.2.1. Sequence diagram for the export function (Here our group chooses this function as the main point to present) 73](#_Toc16313)

[4.2.2. Class diagram 74](#_Toc29446)

[4.3. Interface design and specification 75](#_Toc13729)

[4.3.1. Customer interface 75](#_Toc10617)

[a) Login function 75](#_Toc26471)

[b) Home page interface 78](#_Toc15336)

[c) Shopping cart management function 82](#_Toc17269)

[d) Order information viewing function 84](#_Toc19923)

[4.3.2. Admin section interface 86](#_Toc31513)

[a) Login function 86](#_Toc16342)

[b) Home page interface 90](#_Toc14051)

[c) Customer management function 92](#_Toc16666)

[d) Order management function 93](#_Toc22221)

[e) Category management function 95](#_Toc12640)

[f) Product management function 96](#_Toc25744)

[g) IMEI management function (sub-function) 97](#_Toc115)

[h) Import management function 98](#_Toc20964)

[i) Export management function 99](#_Toc24055)

[k) Inventory data management function (total inventory) 100](#_Toc10616)

[l) Warranty Management function 101](#_Toc25847)

[m) Statistical management function 102](#_Toc20705)

[Chapter V: System Testing 107](#_Toc2209)

[5.1. Test Plan 107](#_Toc10966)

[5.1.1. Introduction 107](#_Toc29705)

[a) Objective 107](#_Toc9326)

[b) Background 107](#_Toc23794)

[5.1. 2 . Types of testing used 107](#_Toc19779)

[a) Functional Testing 107](#_Toc2535)

[b) GUI Testing 107](#_Toc22944)

[5.1. 3. Testing tools 108](#_Toc3639)

[5.1. 4. Test scope 108](#_Toc10017)

[5.1. 5 . Remove from scope 108](#_Toc4736)

[5.1 .6 . Completion criteria 108](#_Toc24443)

[5.1 .7 . Risks and mitigation plans 109](#_Toc30452)

[5. 2 . Test user interface functionality 109](#_Toc29242)

[5.1.1. Customer account management function 109](#_Toc17542)

[5.1.2. Category and product management functions 109](#_Toc9657)

[5.1.3. Order processing function (admin ) 110](#_Toc19601)

[5.1.4. Import and warehouse management functions 110](#_Toc16549)

[5.1.5. Statistics & reporting functions 110](#_Toc4803)

[5.1.6. Customer Email sending function 111](#_Toc6343)

[5. 3 . Functional Test 112](#_Toc15184)

[5.2. 1. Admin 112](#_Toc30954)

[5.2. 2. Customer 112](#_Toc18352)

[5.2. 3. Product 113](#_Toc7953)

[5.2. 4. Orders 113](#_Toc32386)

[5.2. 5. Order\_Details 114](#_Toc23909)

[5.2. 6. Bill\_Exported 114](#_Toc32580)

[5.2. 7. Bill\_Exported\_Details 114](#_Toc31615)

[5.2. 8. Bill\_Imported 115](#_Toc18683)

[5.2. 9. Bill\_Imported\_Details 115](#_Toc8944)

[5.2.10. Supplier 115](#_Toc29450)

[5.2. 11. IMei\_Product 116](#_Toc14655)

[5.2. 12. IMei\_Product\_Stock 116](#_Toc4547)

[5.2. 13. Insurance 116](#_Toc23228)

[5.2. 14. Discount 117](#_Toc19574)

[5.2. 15. Stock 117](#_Toc1222)

[5.2. 16. Category 117](#_Toc6414)

[5.2. 17. Staff 118](#_Toc13184)

[Chapter VI: Conclusion 119](#_Toc29049)

[6.1 . Results achieved 119](#_Toc5649)

[6.2 . System limitations 121](#_Toc14702)

[6.3 . Future development direction 121](#_Toc21974)

[6.4 . General conclusion 122](#_Toc32595)

[References 123](#_Toc4145)

# Menu of Image

[Image 1: Internal organizational structure diagram 1](#_Toc24972)

[Image 2 : Agile process is implemented in Sprints 56](#_Toc4367)

[Image 3 : Representing the Agile method 56](#_Toc23469)

[Image 4: Simulation of how the classification model works 67](#_Toc22223)

[Image 5: User account registration interface 74](#_Toc24871)

[Image 6: User login interface 75](#_Toc26838)

[Image 7: User's OTP confirmation code input interface 76](#_Toc20436)

[Image 8: New password entry interface 77](#_Toc2601)

[Image 9: Home page interface when user logs in 77](#_Toc4337)

[Image 10: Product details 78](#_Toc4144)

[Image 11: Product information when out of stock 80](#_Toc30944)

[Image 12: User shopping cart interface 81](#_Toc26617)

[Image 13: Order information interface 83](#_Toc18909)

[Image 14: User order status interface 85](#_Toc22164)

[Image 15: Admin account registration function interface 85](#_Toc6520)

[Image 16: Admin account login function interface 86](#_Toc21793)

[Image 17: Interface when Admin forgets account password 87](#_Toc5848)

[Image 18: Admin password reset interface 88](#_Toc11918)

[Image 19: Admin interface after login 89](#_Toc24588)

[Image 20: Admin's customer management interface 91](#_Toc29725)

[Image 21: Admin order management interface 92](#_Toc18172)

[Image 22: Admin order detail interface 93](#_Toc18206)

[Image 23: Admin Category Management Interface 94](#_Toc10926)

[Image 24: Product Management interface helps Admin track 95](#_Toc10632)

[Image 25: IMEI Management interface helps Admin monitor 96](#_Toc28678)

[Image 26: Import Management interface supports Admin tracking 97](#_Toc21595)

[Image 27: Export Management interface supports Admin to check and process orders 98](#_Toc15652)

[Image 28: Inventory Data Management interface helps Admin track 99](#_Toc18914)

[Image 29: Warranty Management Interface 100](#_Toc29962)

[Image 30: 'Bar chart' Statistics Management interface 101](#_Toc19291)

[Image 31: 'Line chart' Statistics Management interface 102](#_Toc7483)

[Image 32: 'Pie chart' Statistics Management interface 103](#_Toc21648)

[Image 33: 'Area chart' Statistics Management interface 104](#_Toc25957)

# Chapter I : Topic Overview

## 1.1. Survey the current status of the system

### 1.1.1. Internal

#### a) Internal organizational structure.

Sales management software operates as a unified system, supporting the management of internal business and operational processes. The organizational structure includes:

* **Management & Operations:** Make decisions, manage systems and monitor operations.
* **Sales Department:** Manage orders, customers, prices and promotions.
* **Warehouse department:** Inventory control, import and export, inventory of goods.
* **Accounting Department:** Processing invoices, payments, accounts receivable and financial reporting.
* **Technical Department (IT):** Ensure stable software operation, technical support and data security.
* **Customer service department:** Customer support, complaint resolution and product consulting.

#### b) Internal organizational structure chart

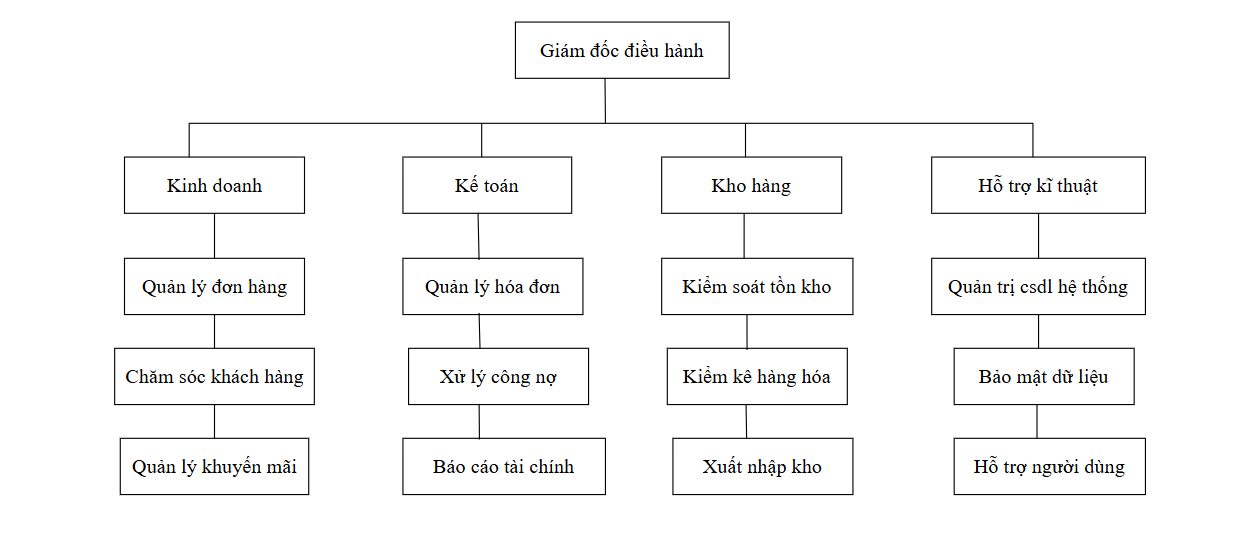
****

Image 1: Internal organizational structure diagram

### 1.1.2. Foreign Affairs

#### a) Organization and its environment in terms of external relations

Sales management software not only works internally but also interacts with external factors such as:

* **Customers:** Communicate through orders, invoices, promotions, customer care.
* **Supplier:** Import link, contract management, product quality control.
* **Tax & Finance Authority:** Meet requirements for accounting, electronic invoices, tax reporting.
* **Bank & payment gateway:** Support financial transactions, online payments.
* **Shipping partners:** Connect with shipping units to track orders and optimize shipping.

#### b) The business environment affects the software.

* **Technology factor:** Update technology trends such as AI, Big Data, IoT to improve efficiency.
* **Legal factors:** Compliance with data security regulations (GDPR, Decree 52/2013/ND-CP).
* **Market factors:** Competition, customer demand, consumer trends.

## 1.2. Business process

Sales management software supports many key operations, including:

|  |  |
| --- | --- |
| **Business** | **Related Processes** |
| Order Management | Create orders, process payments, print invoices, track orders |
| Manage categories and products | Import, export, inventory by supplier |
| Customer Management | Store customer information |
| Supplier Management | Manage supplier information, contracts, and warehouse orders |
| Financial management - accounting | Track revenue, debt, prepare financial reports |
| Reports & Statistics | Export revenue reports, inventory reports, sales data analysis |

#### Business from a manager's perspective

From a management perspective, the most important operations include:

* **Monitor revenue and profit:** Track sales and business performance.
* **Inventory control:** Ensure that there are enough goods to meet demand but not in stock.
* **Customer Management & Customer Care:** Building lasting relationships with customers.
* **Strategic decision making:** Based on data reports to adjust business policies.

|  |  |  |
| --- | --- | --- |
| **Business** | **The stages** | **Related Department** |
| Order Management | Create order → Payment → Print invoice → Update revenue | Manager |
| Import goods | Check goods to be imported → Order from supplier → Receive goods → Update inventory |
| Inventory | Check actual quantity → Compare with data → Adjust if necessary |
| Customer Management | Save information → Track purchase history through orders → Customer care |
| Reports & Statistics | Collect data → Synthesize → Analyze → Export report | Accountant |

**Customer interview questions to get requirements for building business processes:**

**General questions about current process:**

* Can you describe your store/business sales process?
* Currently, what difficulties do you encounter in the sales management process?
* Which steps are taking the most time?

**Questions about product management & inventory:**

* How do you keep track of your current inventory?
* How often do you take inventory? Are there any difficulties in doing this?
* When a large order is placed, is there a delay in warehousing and order processing?

**Customer service question:**

* Do you store customer information? If so, how?
* Do you track customer purchase history? If so, how?

**Reporting & Finance Questions:**

* Do you need to report revenue and profit by day/month/year?
* When you need to check customer debt, how long does it take you to compile the data?
* Is there any problem in tracking cash flow in your business?

**Extended questions about current process**

* In your current sales process, are there any steps that you find redundant or that could be automated?
* Are you using any supporting software? If so, which part do you find not meeting your needs?

**Detailed questions about inventory & product management**

* When a product is about to run out of stock, does the system (if any) warn you? Would you like to have this feature?
* Is the management of import orders from current suppliers done manually or is there a clear process?
* Are you having trouble managing multiple product groups, product codes or shipments?

**In-depth customer questions**

* Have you ever wanted to classify customers by purchase level (loyal, potential, new...)?
* Do you have any promotions, points accumulation, or discounts for each type of customer?

**Questions about delegation and employee management**

* Do you need a system of authorization for each employee? (For example: sales staff cannot edit orders,...)
* Do you track individual employee performance? Is there a need for reporting by employee?

## 1.3. Functional requirement specification

### 1.3.1. Customer account management function

* The system allows users to register a new account by providing personal information such as full name, email, phone number, password,...
* The system allows users to log in to access shopping functions.
* The system allows users to view product lists, view product details, and add products to cart.

**The system allows users to create orders including:**

* Select multiple products for one order.
* Select payment method.
* Enter correct shipping information.
* The system stores the order information into the database and updates the initial order status to "Waiting".

### 1.3.2.Category and product management function

#### Portfolio Management

The system allows admin to:

* Add new product category.
* Edit category information.
* Delete product category.
* Search categories by category code, category name and supplier.

#### Product Management

The system allows admin to:

* Add new products with information: product code, product name, specifications, price, image.
* Each product can have multiple units of goods corresponding to its own IMEI code.
* Update product information.
* Remove product from system.
* Search for products by product code, product name or supplier.

### 1.3.3. Order processing function (Admin)

The system displays a list of orders from customers with the status "Waiting".

**Admin order processing function:**

* Check, confirm order → change status to "Confirming".
* Handling cases of customers who cancel orders ≥ 2 times → delete account and permanently ban shopping.

After confirming the order, the system allows the admin to create a shipping invoice including:

* Delivery note code.
* Employee code in charge.
* Customer information.
* Detailed product list (IMEI code, original price, promotional price if any)

**Promo code:**

The system allows applying discount codes in invoices if they are suitable for programs such as:

* Customer birthday promotion.
* Company anniversary.
* Thank you to our loyal customers.

### 1.3.4. Import and warehouse management functions

**Admin functions:**

* Check product inventory.
* Create new receipt when product quantity is low.
* Update supplier information and import catalog.

**Support system:**

* Import large quantities of products by category.
* Assign a unique IMEI code to each product.
* Once imported, the product is added to the system and available for customers to order.
* Statistics & reporting functions

**The system provides revenue statistics function according to:**

* Time (day, month, year).
* Product catalog.
* Total number of orders and products sold.
* Revenue data is presented in the form of visual chart reports, helping admins easily analyze and develop business strategies.

### 1.3.5. Customer Email sending function

The system integrates the feature of sending emails directly from the system to customers, supporting the following contents:

* Order confirmed successfully.
* Send notifications about promotions and offers.
* Send customer care and after-sales support emails.

## 1.4. Non-functional requirements specification:

#### Performance:

* **Response speed:** The system must respond to user actions within 3 seconds for common functions such as login, product search, and add to cart.
* **Order Processing:** The system must be able to process at least 1000 orders/day without interruption.
* **Product Search:** Search results must be displayed within 2 seconds of the user entering a keyword.

#### Scalability

* The system is designed to be easily scalable as more users, products, or features are added.
* Can integrate with future cloud services or ERP systems without changing the entire system architecture.

#### Security

* Encrypt user passwords using a hash algorithm (e.g. SHA-256 or bcrypt).
* Clear access rights between admin and customer.
* Sensitive data transmission must be via HTTPS protocol.
* The system stores user access and activity logs for monitoring and maintenance purposes.

#### Usability

* The user interface must be friendly, easy to use, and suitable for general users.
* The system should have a mobile-friendly (responsive) version.
* User manuals are available for admins and new customers.

#### Reliability

* The system must remain operational 24/7, unless there is regular maintenance.
* In the event of a temporary loss of connection, the system must be able to automatically restore data once the connection is re-established.
* Order processing and data storage must be secure and intact (no data loss or errors).

#### Maintainability & Updatability

* The system must be programmed in a layered model (Model-View-Controller or similar) for easy maintenance and updating.
* Each function must be packaged separately for easy upgrade or replacement of modules without affecting the entire system.
* Partial automated testing can be performed during development and updates.

**1.5. Proposed solutions**

|  |  |  |
| --- | --- | --- |
| **Ingredient** | **Solution** | **Reason for choosing** |
| Frontend | Java Swing (desktop version) or ReactJS (if you want to extend to the web) | Flexible interface, easy to develop, good interactivity |
| Business logic processing (Backend) | Java (or Python) in MVC pattern | Easy to maintain, clear division between functional layers |
| Database | Microsoft SQL Server | Good security, stable performance, big data support |
| Image storage | Save in local folder combined with path in DB | Avoid database bloat, easy image manipulation |
| Reports - statistics | JFreeChart (Java) or Matplotlib (Python) library | Create visual charts for admin |
| Send Email | JavaMail API or Python integrated SMTP | Send order confirmation, promotions to customers |
| IMEI Management | Each product has its own IMEI list. | Ensure precise control of each machine |

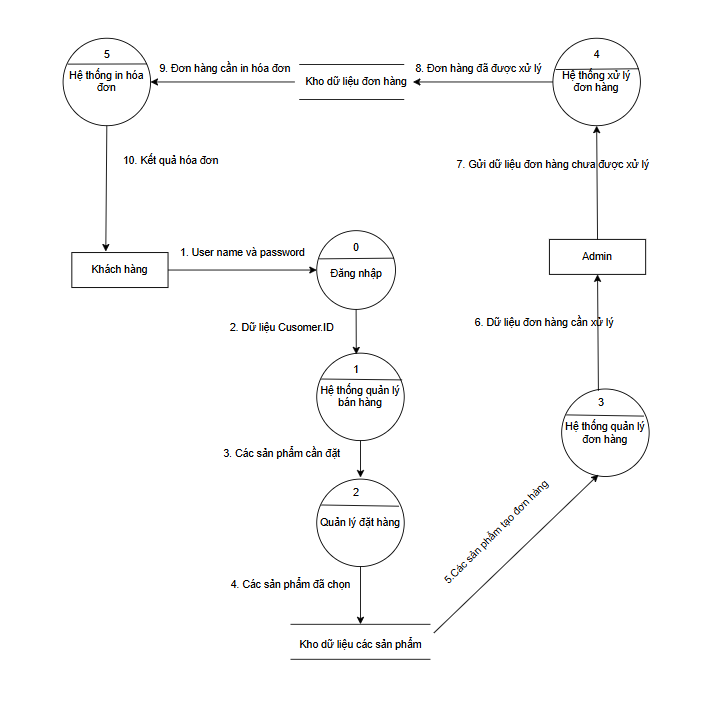
**Future Expansion**

* Connect the system to cloud services to store large data.
* Develop mobile applications using the same database.
* Integrate AI to suggest products and predict sales trends.
* Connect to the shipping API to handle automated shipping.

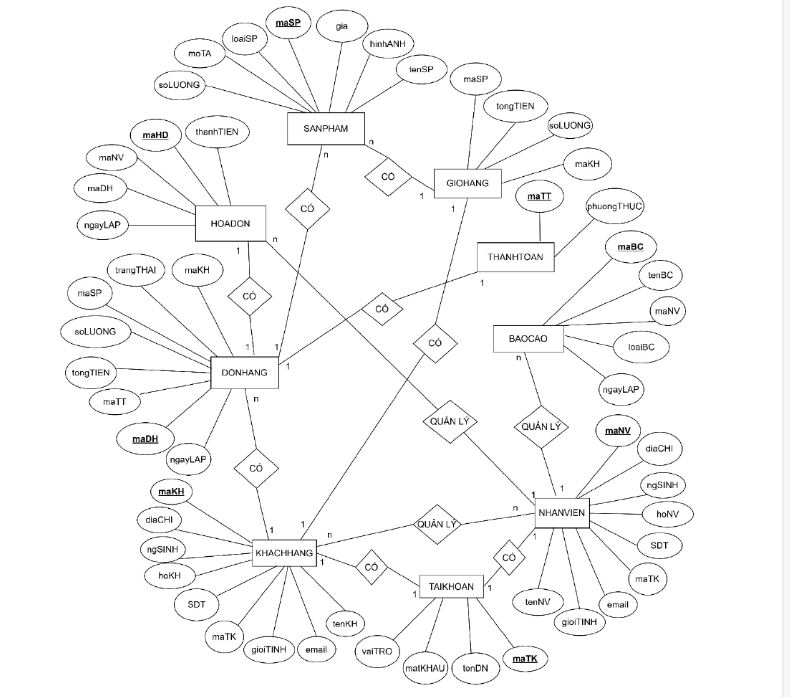
# Chapter II: System Analysis

## 2.1. System analysis overview diagram

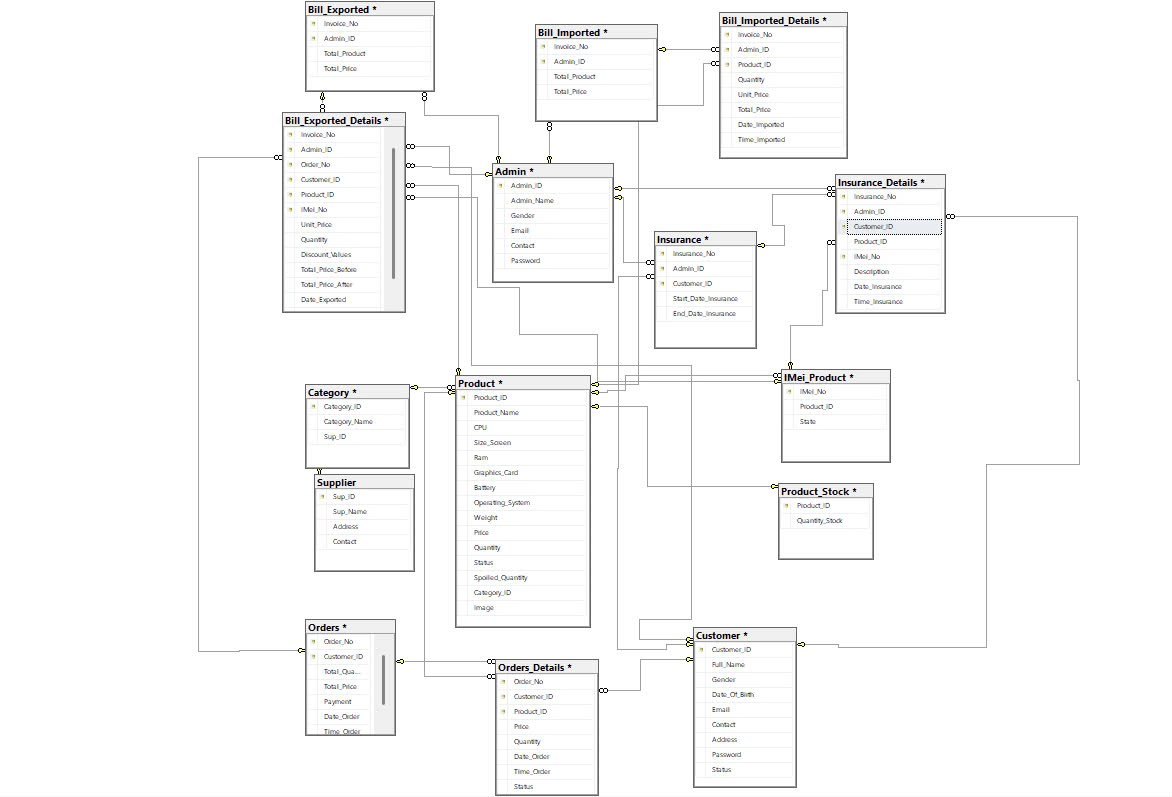
### 2.1.1. Business process context level diagram



### 2.1.2. Entity Relationship Diagram (ERD)



### 2.1.3. Mapping table



## 2.2. Use Case:

### 2.2.1. Define Use Cs ase :

#### a) Main modules

* Manage goods imported to the store
* Laptop brand management .
* Supplier Management
* Invoice Management
* Invoicing
* Customer Management
* Statistical

#### b) Functional requirements - Use Case:

**Sales agent:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Job** | **Job Type** | **Related regulations/formulas** | **Related Forms** | **Note** |
| 1 | Add customer | Storage | Each customer has a unique phone number. |  |  |
| 2 | Print invoice | Storage | - Each invoice records the customer who purchased it and which employee created the invoice.  - Ensure sales volume Laptop type must be less than or equal to the quantity in stock. |  |  |
| 3 | Laptop Lookup | Lookup | - Search by the first characters of the Laptop name |  |  |

**Department - Management:**

Managers can execute Employee use cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Job** | **Job Type** | **Related regulations/formulas** | **Related Forms** | **Note** |
| Add Laptop Brand | Storage | The company name cannot be duplicated. | Laptop Brand Form | Only administrators have the right to add new |
| Add Supplier | Storage | Phone number is valid and cannot be duplicated | Supplier Form | Full contact information and address required |
| Add Laptop Type | Storage | Type names cannot be duplicated. | Laptop Type Form | For example: Gaming, Office, Ultrabook,... |
| Add Laptop | Storage | Product codes cannot be duplicated, have configuration, price, photo,... | Laptop Form | Includes technical information, price, illustrations |
| Import Laptop into warehouse | Warehouse process | Quantity > 0; Check supplier code and product code | Warehouse Entry Form | Can update import price and import date |
| Customer Lookup | Business | Search by phone number, email or full name | Customer Lookup Form | Support quick customer management and search |
| Statistical | Report | Filter by time, product type, brand,... | Statistics Form | Can export as Excel or PDF |
| Total revenue statistics | Report | Total = SUM(Output price × Quantity) | Revenue Form | Can be filtered by month, quarter, or year |

**System functional requirements:**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Content** | **Detailed description** | **Note** |
| 1 | Backup, back up, restore information | Backup deleted information and restore when necessary (To restore , you need to directly access the database) |  |
| 2 | Decentralization of use | Management: Can perform all the business functions of the remaining users.  Sales staff: Only perform the described business functions. |  |

**Non-functional requirements**

|  |  |  |
| --- | --- | --- |
| **STT** | **Content** | **Standard** |
| 1 | Fast and accurate search speed | Effective |
| 2 | Save time,  Reduce storage space and avoid data loss. | Effective |
| 3 | Simple interface, easy to operate | Convenient |

### 2.2.2. General Use Case Diagram:

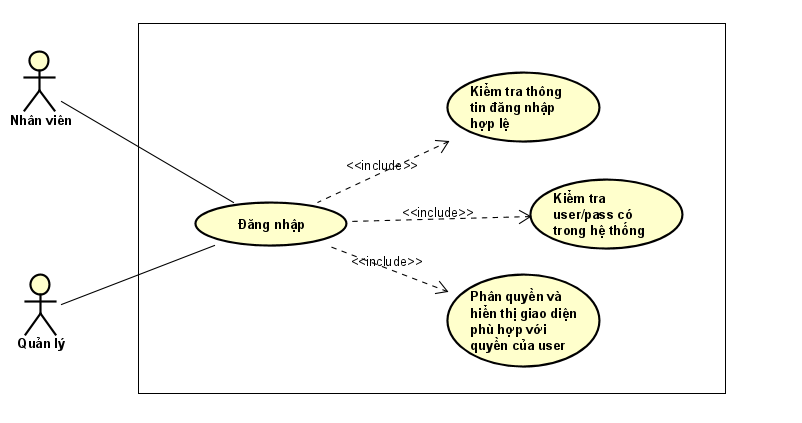
A diagram of a person with text

AI-generated content may be incorrect.

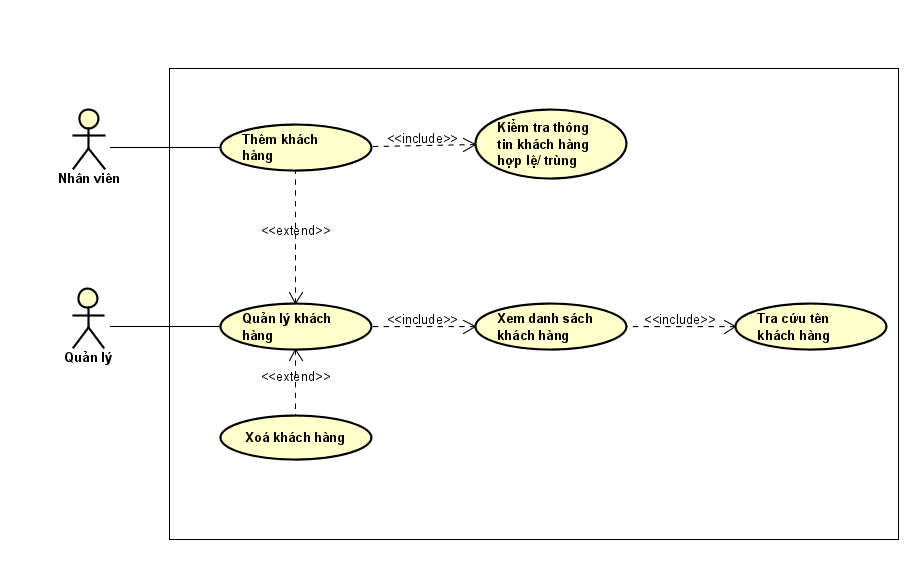
**\*Note:** Managers have full access to Salesperson functions.

### 2.2.3. Description of Use Cases:

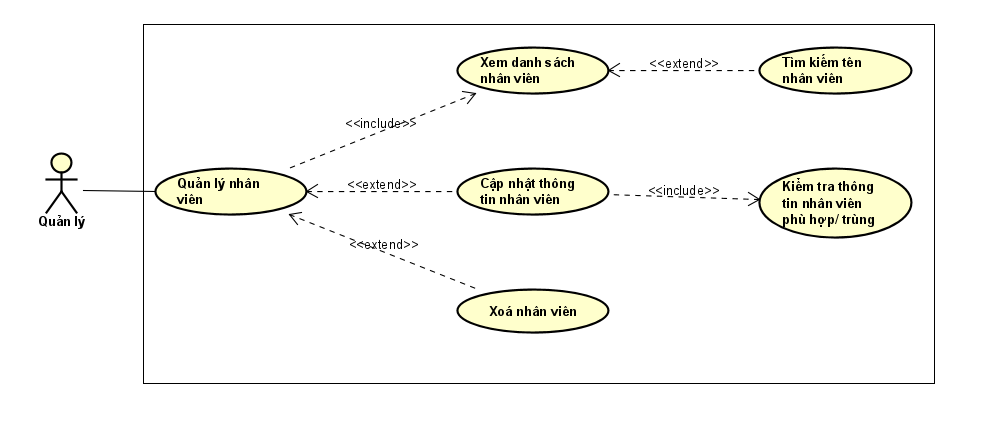
#### a) UC Login :



#### b) UC Employee Management



#### c) UC Customer Management

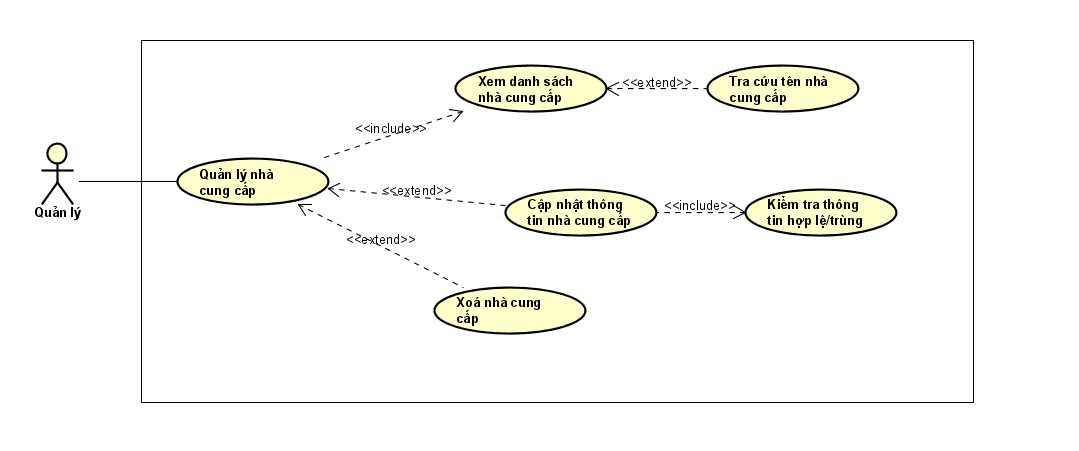


**d) UC Laptop Category Management**

A diagram of a diagram

AI-generated content may be incorrect.

#### e) UC Supplier Management



#### f) UC Laptop Type Management

A diagram of a computer

AI-generated content may be incorrect.

#### g) UC Laptop Management

A diagram of a company

AI-generated content may be incorrect.

#### h) UC Warehouse

A diagram of a diagram

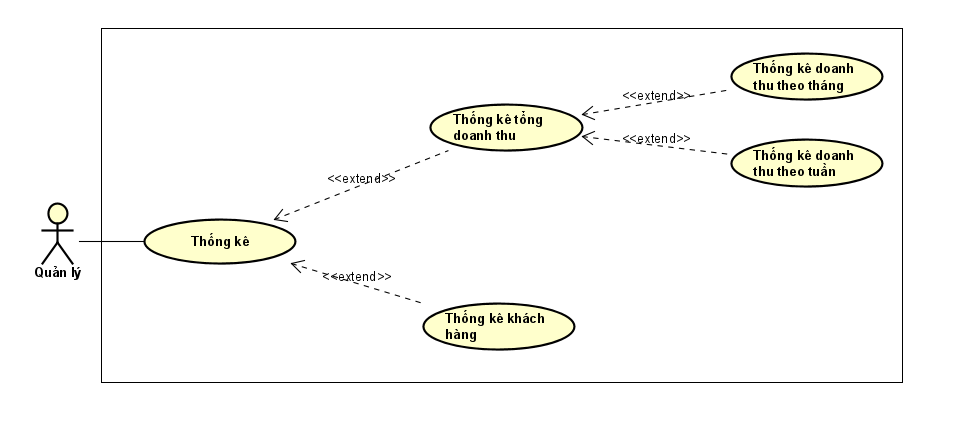
AI-generated content may be incorrect.

#### i) UC Print invoice

A diagram of a diagram

AI-generated content may be incorrect.

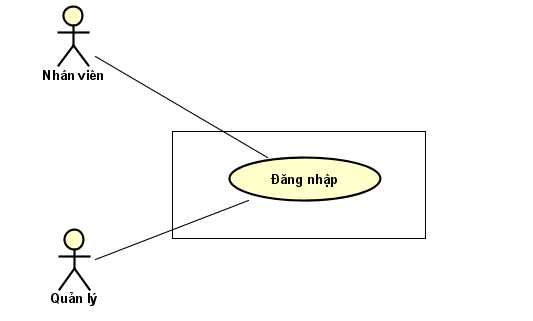
#### k) UC Statistics



## 2.3. Use case analysis :

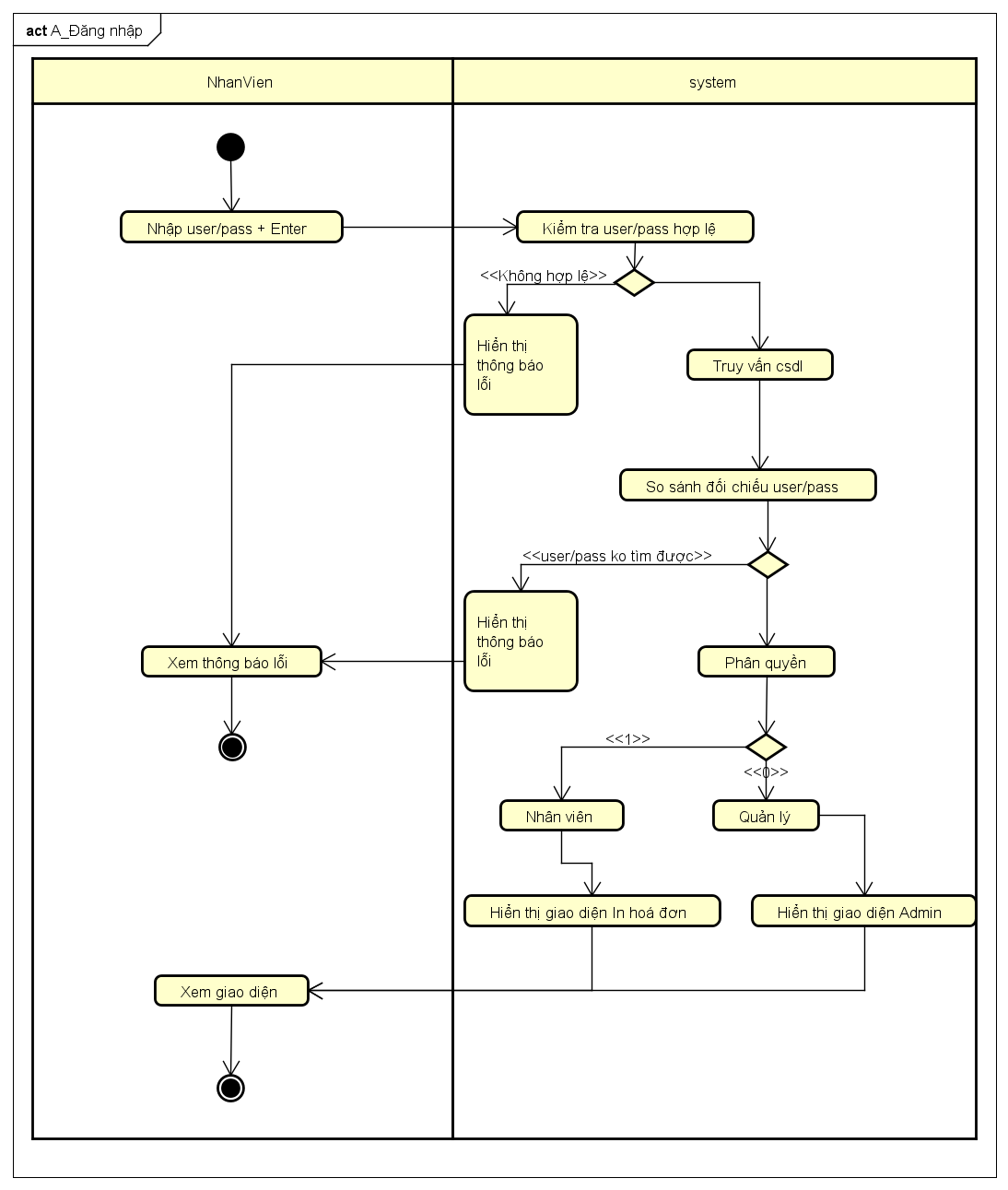
### 2.3.1. Functional design analysis (UC – AC):

#### a) Login



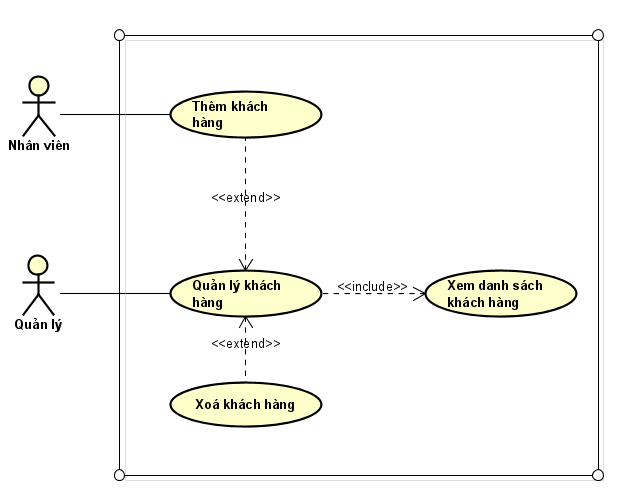
**Describe:**

Employees and managers must log into the system before using the software.



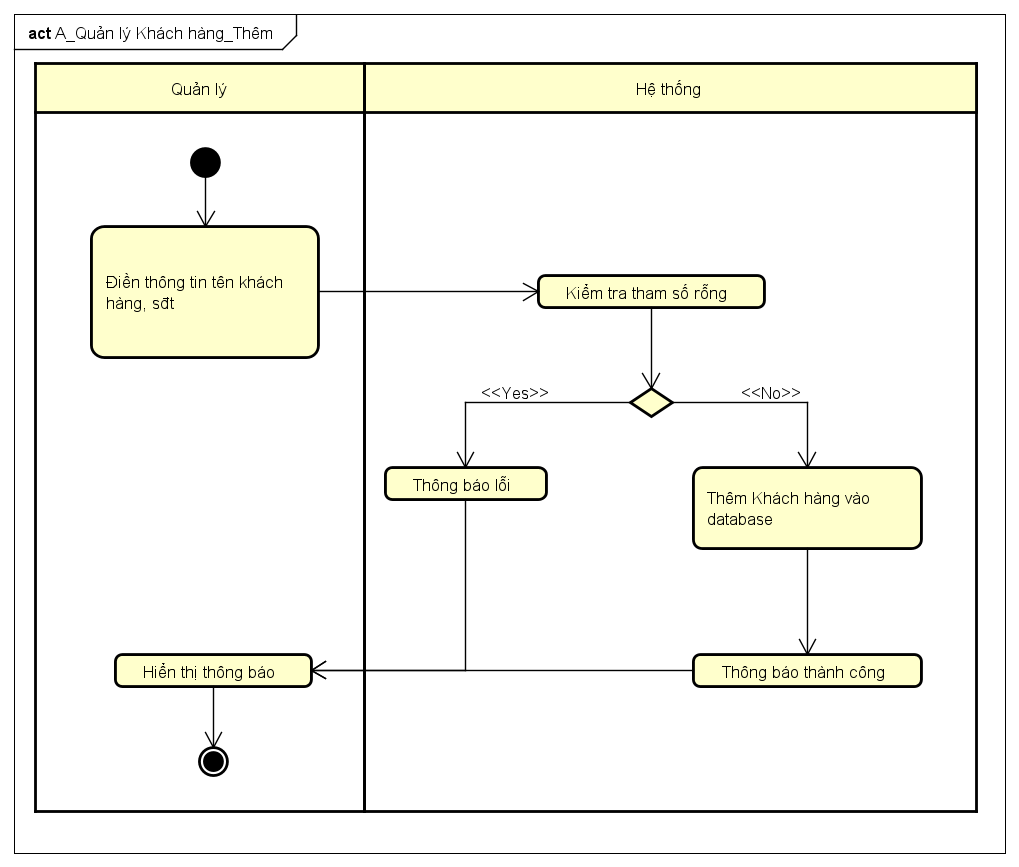
|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Staff, management** | **System** |
|  | 1. Staff and managers enter username/password and click Login |  |
|  | 2. Check username/password is valid, not empty |
|  | 3. If the username/password parameter is valid, get the user object in the database according to the username/password parameter. |
|  | 4. If user exists, perform authorization |
|  | 5. If the user has 0 permissions, open the admin page |
| 6. Manage corresponding page views |  |
| **Alternative flow 1** |  | 5.1. If the user has permission 1, open the Print Invoice page |
|  | 6.1.Staff view the corresponding page |  |
| **Except flow 1** |  | 3.1 Invalid username/password parameter |
|  |  | 4.1 Display error messages |
| 5.1. User views error messages |  |
| **Except flow 2** |  | 4.2. User does not exist or username/password is incorrect |
|  |  | 4.2. Go to error notification step 4.1 |

#### b) Customer Management



**Description:** Both employees and managers have the right to add customers to the system, and managers also have the right to delete customers.

**Add customers:**



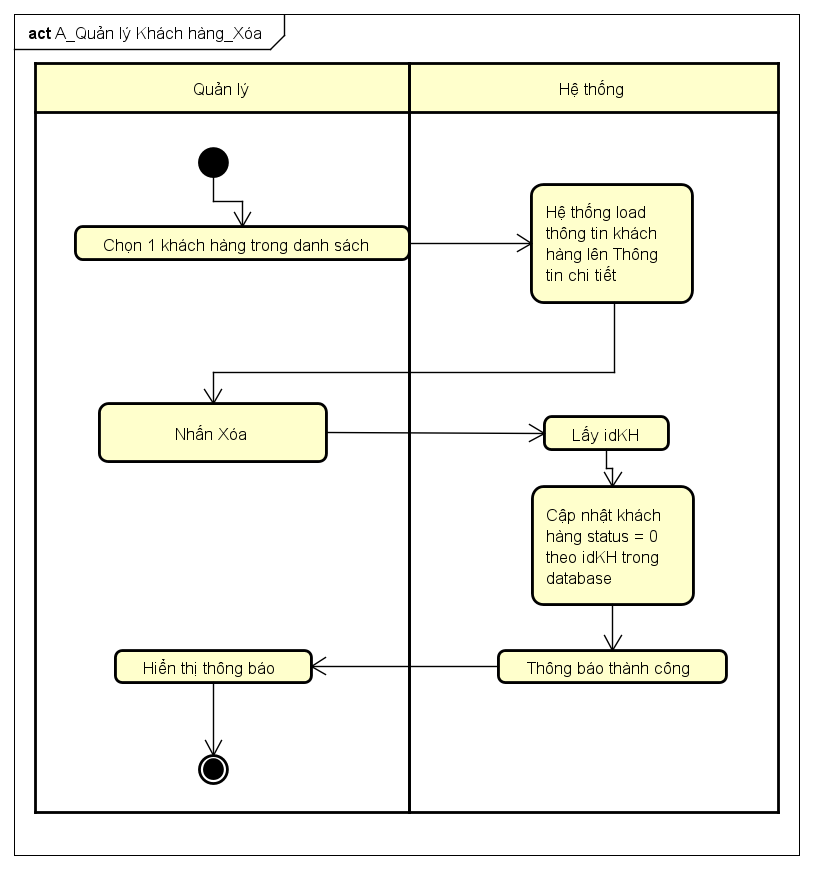
|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Personnel management** | **System** |
|  | 1. Manager, staff enter customer information: name, phone number |  |
|  | 2. Check if the parameter is valid and not empty |
|  | 3. Add customer information to the database |
|  | 4. Show success message |
| 5. Managers and employees view the announcement page |  |
| **Except flow** |  | 3.1 Invalid parameter |
|  |  | 4.1 Display error messages |
| 5.1. User views error messages |  |

**Customer Update:**



|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manage customer information selection |  |
|  | 2. Display selected customer information |
| 3. Edit customer information |  |
|  | 4. Check for valid parameters (not empty) |
|  | 5. If the parameter is valid, update customer information by idKH into the system. |
|  | 6. Show success message |
| 7.Manage view notification page |  |
| **Except flow** |  | 5.1 Invalid parameter |
|  |  | 6.1 Display error messages |
| 7.1. User viewing error messages |  |

**Delete customer**



|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager selects the customer to delete |  |
|  | 2. Display selected customer details |
| 3. Press delete |  |
|  | 4. Get customer idKH |
|  | 5. Update status = 0 according to the selected customer id in the database |
|  | 6. Successful deletion notification |
| 7. Manage viewing of notification pages |  |

#### c) Manage Laptop catalog

A diagram of a diagram

AI-generated content may be incorrect.

**Description:** Manager has the right to view, add, update, delete Laptop brands.

**Add Laptop Brand**

A diagram with text and words

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager fills in the information of the Laptop brand to be added and clicks Add |  |
|  | 2. Check if the parameter is valid and not empty |
|  | 3. If the parameter is valid, add the Laptop brand information to the database. |
|  | 4. Show success message |
| 5. Manage viewing of notification pages |  |
| **Except flow** |  | 3.1 Invalid parameter |
|  |  | 4.1 Display error messages |
| 5.1. User views error messages |  |

**Update Laptop products**



|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager clicks to select the Laptop brand information that needs to be edited |  |
|  | 2. Display information of selected Laptop brand |
| 3. Manage and edit laptop brand information |  |
|  | 4. Check if parameter is valid (not empty) |
|  | 5. If valid, update Laptop brand information according to id\_DanhMucLapTop into the system |
|  | 6. Success notification |
| 7. User sees success message |  |
| **Except flow** |  | 5.1 Invalid parameter |
|  |  | 6.1 Display error messages |
| 7.1. User viewing error messages |  |

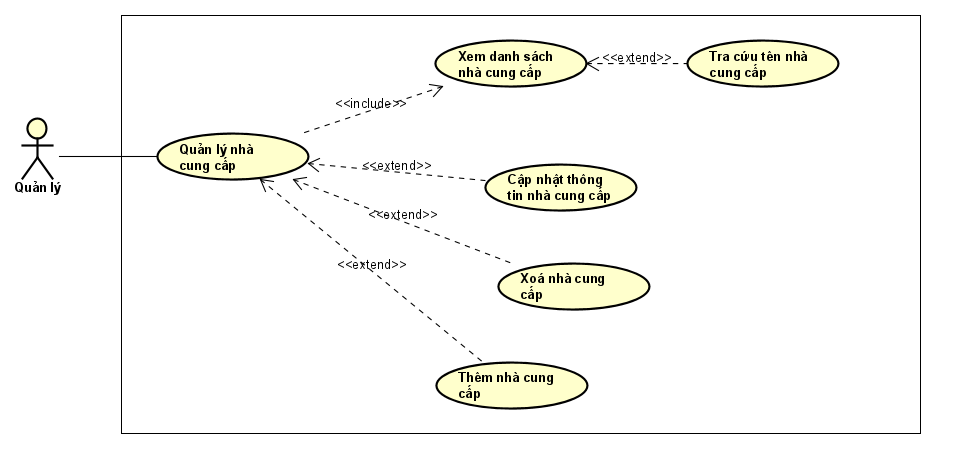
**Delete Laptop Brand**

A diagram of a flowchart

AI-generated content may be incorrect.

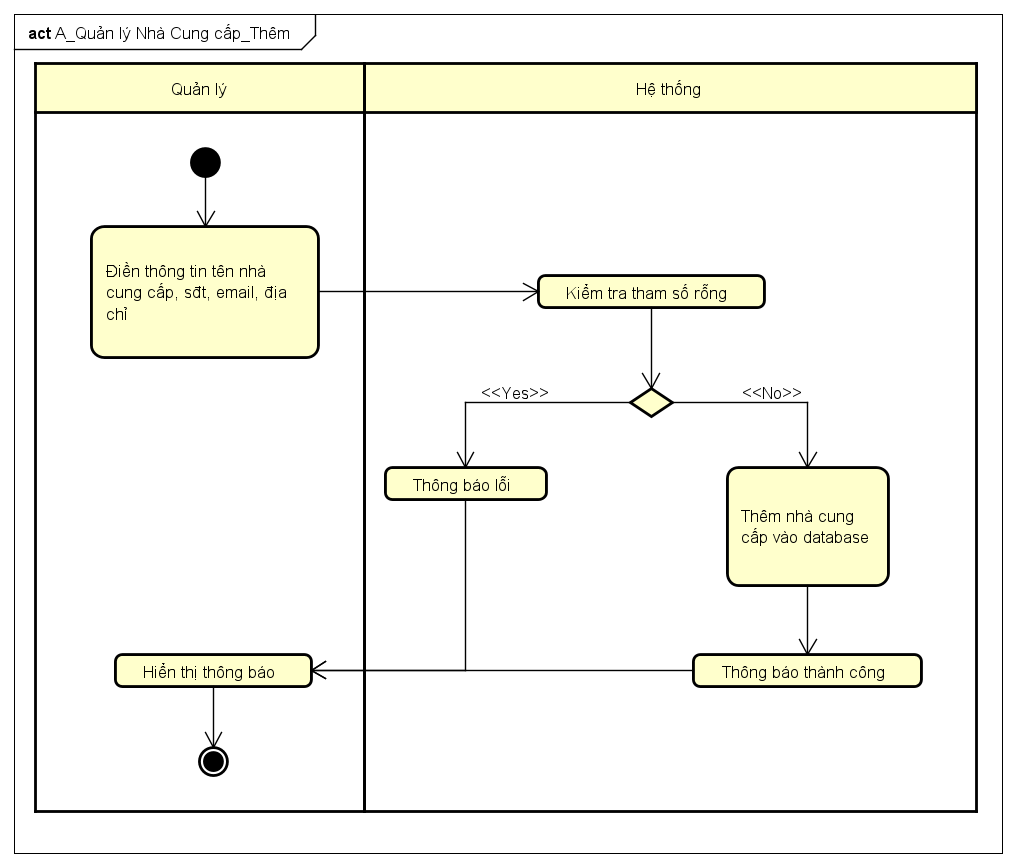
|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1. Manage and select the Laptop brand to delete |  |
|  | 2. Load and display detailed information of selected Laptop product |
| 3. Press delete |  |
|  | 4. Get the idHangGiay of the Laptop brand |
|  | 5. Update status = 0 according to the selected id\_DanhMucLapTop in the database |
|  | 6. Successful deletion notification |
| 7. Manage viewing of notification pages |  |

#### d) Supplier Management



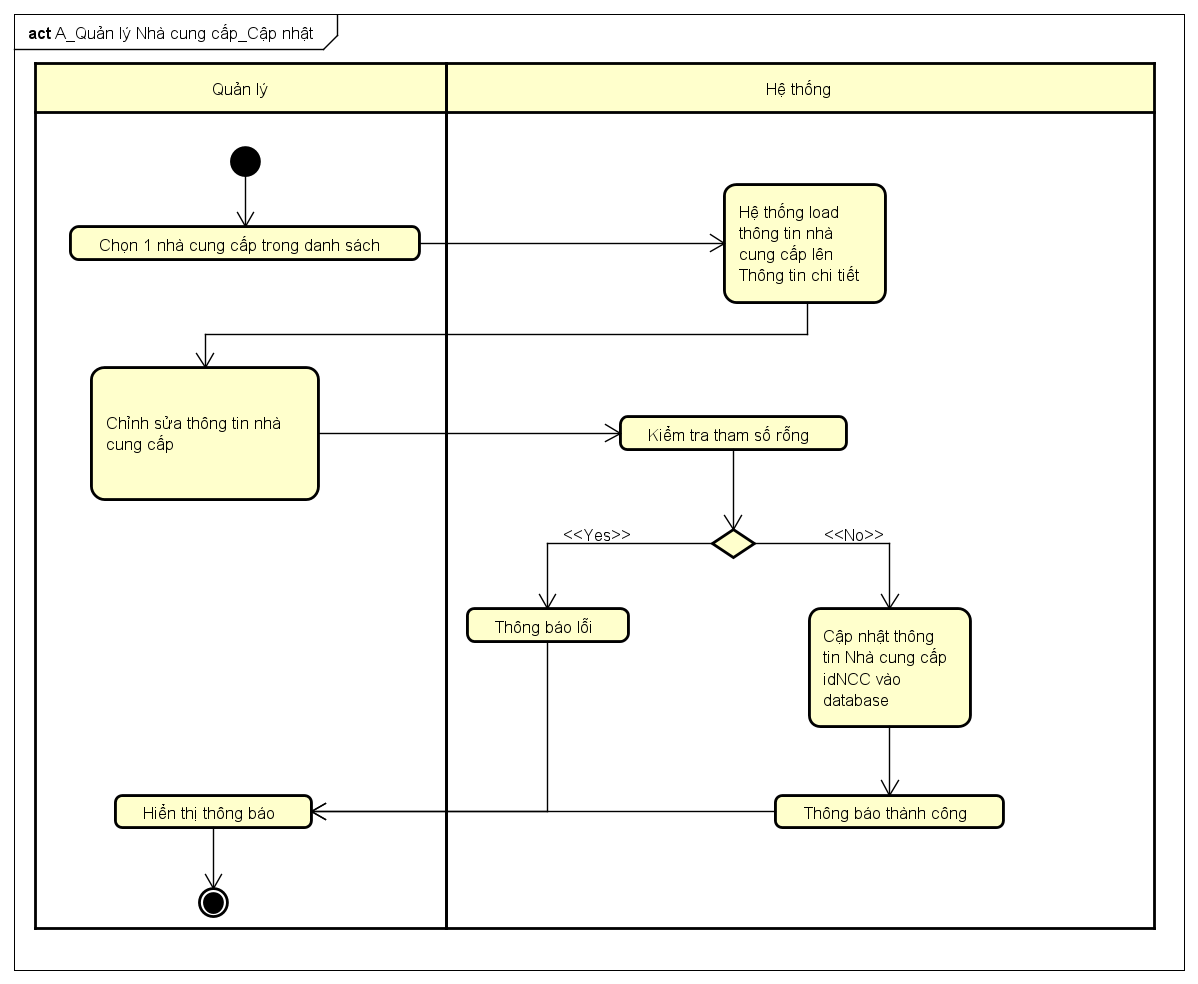
**Description:** Managers have the right to view the list of suppliers, add, update, and delete supplier information.

**Add supplier**



|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager fill in supplier information: name, phone number, email, address and click Add |  |
|  | 2. Check if the parameter is valid and not empty |
|  | 3. If the parameter is valid add the supplier information to the database |
|  | 4. Show success message |
| 5. Manage viewing of notification pages |  |
| **Except flow** |  | 3.1 Invalid parameter |
|  |  | 4.1 Display error messages |
| 5.1. User views error messages |  |

**Update supplier**



|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager clicks to select supplier information to edit |  |
|  | 2. Load and display information about the selected Laptop brand |
| 3. Manage and edit supplier information |  |
|  | 4. Check if parameter is valid (not empty) |
|  | 5. If valid, update supplier information according to idNCC into the system. |
|  | 6. Success notification |
| 7. User sees success message |  |
| **Except flow** |  | 5.1 Invalid parameter |
|  |  | 6.1 Display error messages |
| 7.1. User viewing error messages |  |

**Delete supplier**



|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager selects the supplier to delete |  |
|  | 2. Load and display selected supplier details |
| 3. Press delete |  |
|  | 4. Get supplier's NCC id |
|  | 5. Update status = 0 according to the selected IDNCC in the database |
|  | 6. Successful deletion notification |
| 7. Manage viewing of notification pages |  |

#### e) Laptop Type Management

A diagram of a computer network

AI-generated content may be incorrect.

**Description:** Managers have the right to view the list of Laptop types, add, update, and delete information about Laptop types.

**Add Laptop Type**

A diagram of a workflow

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manage the information of the Laptop type: size, color, quantity, price and click Add |  |
|  | 2. Check if the parameter is valid and not empty |
|  | 3. If the parameter is valid add the Laptop type information to the database |
|  | 4. Show success message |
| 5. Manage viewing of notification pages |  |
| **Except flow** |  | 3.1 Invalid parameter |
|  |  | 4.1 Display error messages |
| 5.1. User views error messages |  |

**Update Laptop Type**

A diagram with text and words

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager click to select the type of Laptop to edit |  |
|  | 2. Load and display information about the selected Laptop type |
| 3. Manage and edit information about Laptop type: quantity, selling price |  |
|  | 4. Check if parameter is valid (not empty) |
|  | 5. If valid, update information according to id\_LoaiLapTop into the system |
|  | 6. Success notification |
| 7. User sees success message |  |
| **Except flow** |  | 5.1 Invalid parameter |
|  |  | 6.1 Display error messages |
| 7.1. User viewing error messages |  |

**Delete Laptop Type**

A diagram with text and black circles

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager selects the type of Laptop in the list to delete |  |
|  | 2. Load and display detailed information of the selected Laptop type |
| 3. Press delete |  |
|  | 4. Get idLoaigiay |
|  | 5. Update Laptop Type status = 0 according to the selected Laptop Type id in the database |
|  | 6. Successful deletion notification |
| 7. Manage viewing of notification pages |  |

#### f) Laptop Management

A diagram of a software company

AI-generated content may be incorrect.

**Description:** Manager has the right to view, add, update, delete Laptops

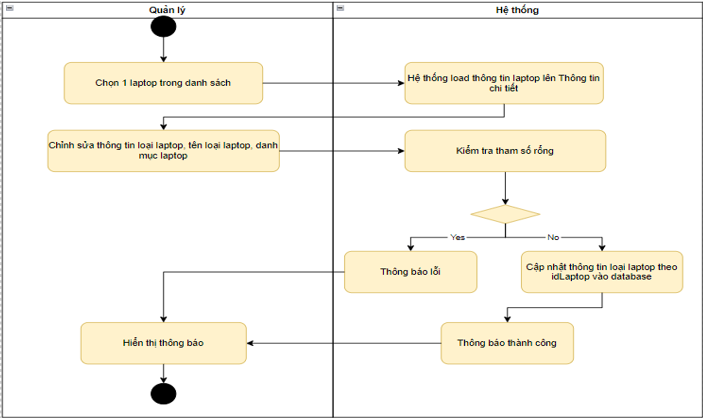
**Add Laptop**

A diagram with text and words

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manage Laptop information: quantity, price and click Add |  |
|  | 2. Check if the parameter is valid and not empty |
|  | 3. If the parameter is valid add Laptop information to the database |
|  | 4. Show success message |
| 5. Manage viewing of notification pages |  |
| **Except flow** |  | 3.1 Invalid parameter |
|  |  | 4.1 Display error messages |
| 5.1. User views error messages |  |

**Update Laptop**



|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager click to select Laptop to edit |  |
|  | 2. Load and display information of selected Laptop |
| 3. Manage and edit information about Laptop type: quantity, selling price |  |
|  | 4. Check if parameter is valid (not empty) |
|  | 5. If valid, update information according to idLaptop into the system. |
|  | 6. Success notification |
| 7. User sees success message |  |
| **Except flow** |  | 5.1 Invalid parameter |
|  |  | 6.1 Display error messages |
| 7.1. User viewing error messages |  |

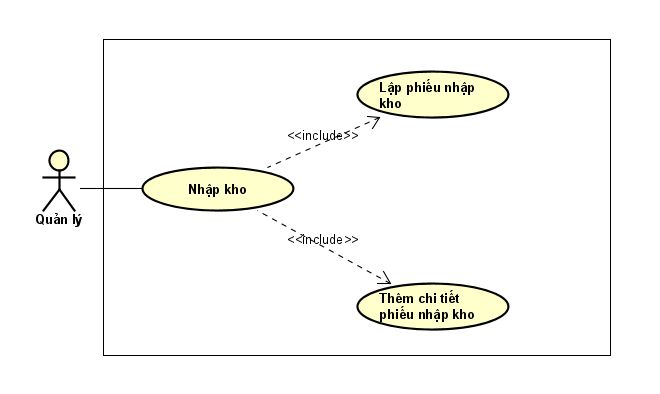
**Delete Laptop**

A diagram of a computer

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager selects Laptop in the list to delete |  |
|  | 2. Load and display detailed information of selected Laptop |
| 3. Press delete |  |
|  | 4. Get idGiay |
|  | 5. Update Laptop status = 0 according to the selected Laptop id in the database |
|  | 6. Successful deletion notification |
| 7. Manage viewing of notification pages |  |

#### g) Warehouse



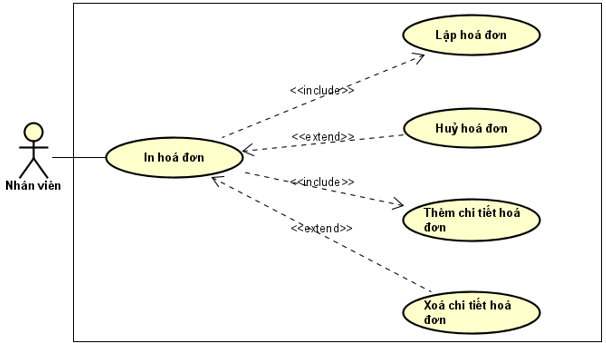
**Description:** Manage to create warehouse receipt then add details of each type of Laptop to the warehouse receipt

A screenshot of a diagram

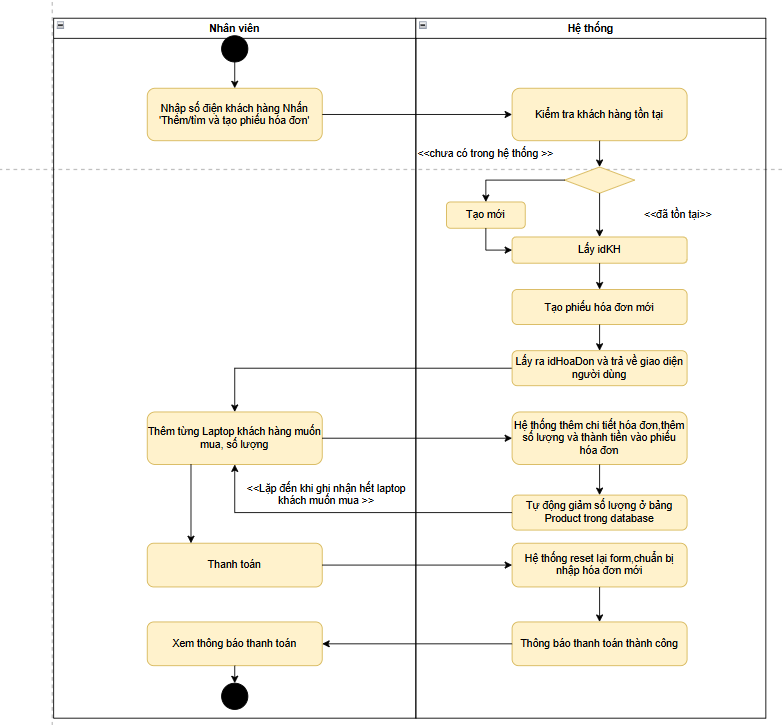
AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| **Basic flow** | **Manage** | **System** |
|  | 1.Manager clicks on create warehouse receipt |  |
|  | 2. Execute creating warehouse receipt |
|  | 3. Return the invoice information just created |
| 4.User clicks on Enter warehouse import details |  |
|  | 5.Display the interface of entering detailed warehouse import invoices |
| 6. User enters into the table that appears: Laptop, size, quantity, price and clicks Add |  |
|  | 7.Add Laptop to Database |
|  | 8. Add up the quantity and total amount into the main warehouse receipt. |
|  | 9. Automatically update the number of Laptops into the Giay table in the database |
| 10.Press x to finish Add |  |

#### h) Print invoices

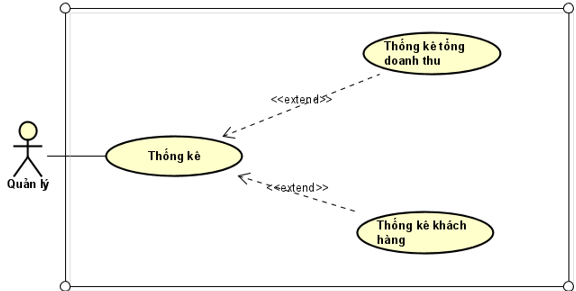


**Description:** Employees have the rights to create new invoices, cancel invoices, add and delete invoice details.

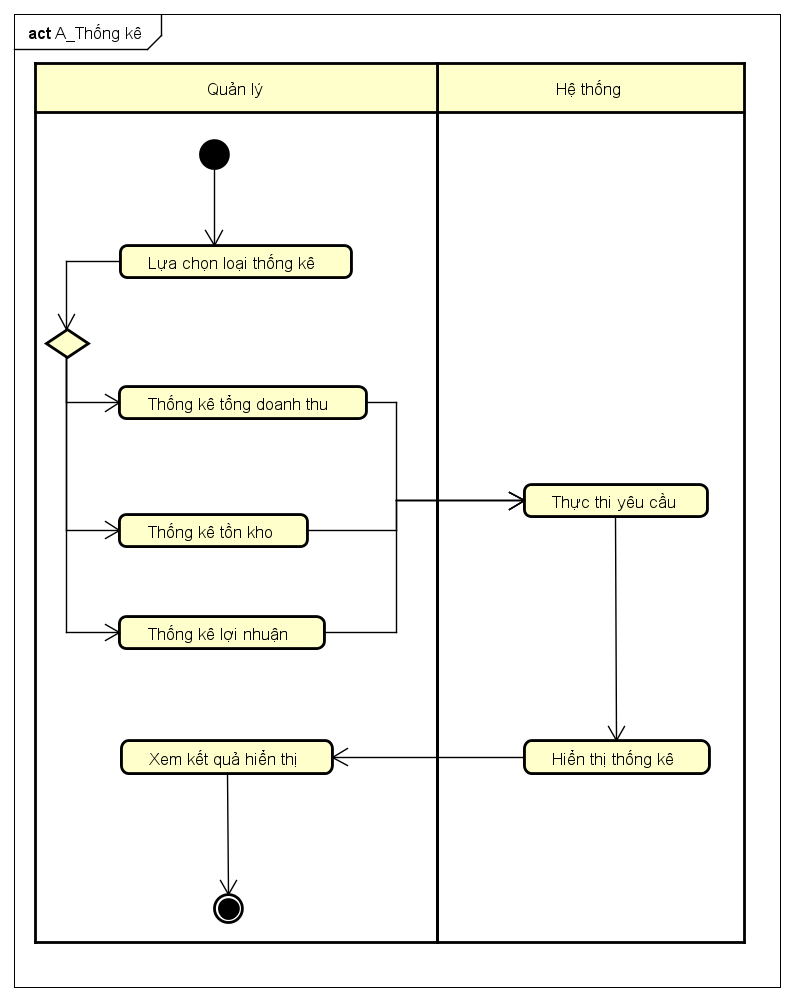


|  |  |  |
| --- | --- | --- |
| **Basic Flow** | **Staff** | **System** |
|  | 1. Enter customer's phone number/name and click "Add/Search to create invoice" |  |
|  | 2. Check if the entered customer exists or not |
|  | 3.1. If it already exists, go to step 4 |
|  | 3.2 If not in the system, create a new one and go to step 4. |
|  | 4. Get customer ID |
|  | 5. Create new invoice |
|  | 6. Get the invoice ID and return it to the user interface |
| 7.Add each Laptop, the quantity the customer wants to buy |  |
|  | 8. The system adds invoice details, quantity and total amount to the invoice. |
|  | 9. Automatically reduce quantity by Laptop name in Laptop table of database |
| 10. Click Checkout |  |
|  | 11. The system resets the form, preparing to enter a new invoice. |
|  | 12. Payment success notification |
| 12. View payment notification |  |

#### i) Statistics



**Description:** Manage statistics of total store revenue and top suppliers, categories and products most purchased by customers



|  |  |  |
| --- | --- | --- |
| **Basic Flow** | **Manage** | **System** |
|  | 1. Select the type of statistics by choosing 1 of steps 2, 3, 4 |  |
| 2. Total revenue statistics |  |
| 3. Inventory statistics |  |
| 4. Profit statistics |  |
|  | 5. Receive requests and execute requests |
|  | 6. Display statistics |
| 7.View the displayed results |  |

# Chapter III: Software development process (Agile process)

## 3.1. Overview of Agile process

### 3.1.1. Concept

**Agile (short for Agile Software Development)** is a flexible software development method, implemented using short iterations of 1 to 4 weeks. The goal of Agile is to help shorten product development time, bringing products to customers as soon as possible.

**In essence, Agile** is a methodology, a philosophy based on the principle of iterative and incremental development, so it possesses high flexibility. This characteristic goes against traditional project management methods - which inherently implement stages in a linear manner and are extremely passive in the face of unexpected changes.

### 3.1.2. Source

**Before Agile, the Waterfall** model was considered the gold standard for software development. Typically, the process would start with a business analyst writing a business requirements document, describing the requirements they needed from the software. These documents were often very long and detailed, containing everything from the overall strategy to the technical specifications. Once all the documents were complete, developers would start coding the software, refining it, and testing it. This entire process could take several years to complete and the software would be ready for use.

Software development began to change with the rise of the Internet. This was also the time when the effectiveness of the Waterfall method was questioned. It consumed too much time and effort from the implementation team, while lacking feasibility as it was built entirely from research without real feedback from users.

### 3.1.3. Characteristics

* **Iterative:** The project will be implemented in repeated segments, usually with a short time frame (from 1-4 weeks). In each segment, the project development team will perform all necessary tasks such as planning, requirements analysis, design, implementation, testing to produce small parts of the product.
* **Incremental & Evolutionary:** At the end of each iteration, the team delivers small pieces of the final product, which are usually complete, working, thoroughly tested, and usable. Over time, as iterations follow one another, these working pieces accumulate and grow until the entire customer requirement is met.
* **Adaptive:** Since the segments only last for a short period of time and the planning is also constantly adjusted, changes in the development process (changing requirements, changing technology, changing goal orientation, etc.) can be responded to appropriately.
* **Self-Organizing and Cross-Functional Teams:** These team structures assign work without rigid job descriptions or clear hierarchies. Self-organizing teams have the necessary skills to be empowered to make decisions, manage themselves, and organize their own work to achieve maximum efficiency.
* **Empirical Process Control:** Agile teams make decisions based on data rather than theoretical calculations or assumptions. Agile shortens the feedback cycle to facilitate adaptation and increases flexibility, which leads to process control and increased productivity.
* **Face-to-face communication:** Agile is not against documentation, but it values face-to-face communication over paper. Agile encourages the development team to talk directly to understand what the customer really needs. In internal team communication, Agile encourages direct discussion and agreement on the design of the system and jointly implement the required functions.
* **Value-based development:** One of the core principles of Agile is that “working products are the measure of progress.” Agile teams often collaborate directly and frequently with customers to understand which requirements have higher priority and can deliver more value to the project as soon as possible.

## 3.2. Agile process activity diagram

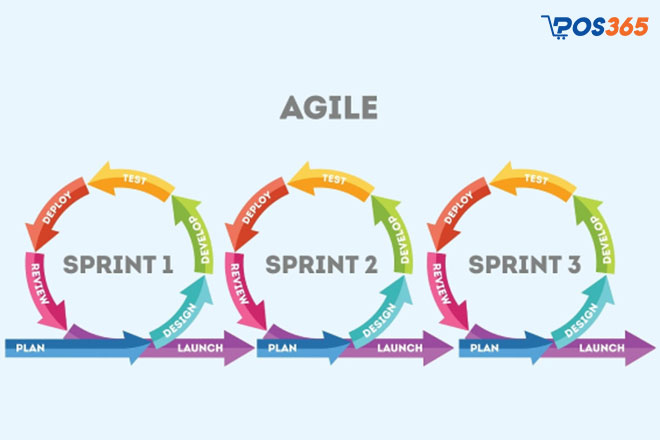


Image 2 : Agile process is implemented in Sprints

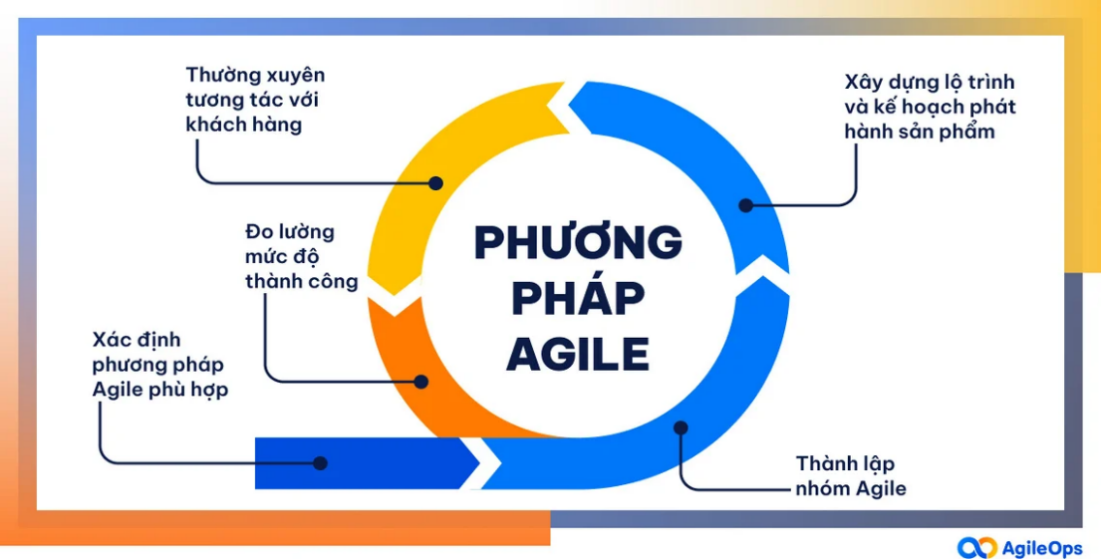


Image 3 : Representing the Agile method

## 3.3. Applying Agile process to Sales Management software (Admin section)

### 3.3.1. Initiation Phase – Building the Product Backlog

* The Product Owner (PO) coordinates with stakeholders such as admin, sales staff, and managers to list all the features that the system needs to have.
* These requirements are arranged in order of priority, put into a list called the Product Backlog.

**Main Epics (high level functionality):**

|  |  |
| --- | --- |
| **Epic Code** | **Function** |
| EP01 | Manage categories and products |
| EP02 | Order processing |
| EP03 | Warehouse management and import |
| EP04 | Statistics and reports |
| EP05 | Send automatic emails to customers |
| EP06 | Security, performance, scalability |

### 3.3.2. Sprint Planning – Sprint 1 Planning

**Objective:** Improve product and category management functions.

**Time:** Sprint 1 lasts 2 weeks.

**Example User Stories in Sprint 1:**

* As an admin, I want to add new product categories to easily manage product groups.
* As an admin, I want to edit or delete a category to update the data correctly.
* As admin, I want to add product with photo, specifications and price.

→ Expected result: Product and category management module is completed and running.

### 3.3.3. Sprint Implementation (Iterative Development)

The group is divided into:

* **Frontend** : Java Swing (desktop).
* **Backend:** Java based on MVC model.

|  |  |
| --- | --- |
| **Day** | **Job** |
| Day 1 | Category & Product UI Design |
| Day 2–4 | Code CRUD functions (add/edit/delete) |
| Day 5–6 | Frontend ↔ backend integration |
| Day 7–10 | Write test cases, internal testing, bug fixing |

### 3.3.4. Daily Scrum – 15-minute meeting every day

Answer 3 questions:

* What did I do yesterday?
* What am I going to do today?
* What problems do I have?

→ Goal: Grasp progress, support each other, prevent early blockages.

### 3.3.5. Sprint Review – Present results to PO

Demo group:

* Interface to add/delete/edit categories and products.
* Search for products by name or code.
* Data is stored persistently on SQL Server.

**Response from PO:** "Add feature to preview product image before saving."

→ This feature will be put back into the Product Backlog.

### 3.3.6. Sprint Retrospective – Process Improvement

* **Good:** The team communicates well and divides tasks reasonably.
* **Need to improve:** Testing time is urgent → spend 1.5 more days for testing in the next Sprint.

### 3.3.7. Plan for the next Sprints

|  |  |  |
| --- | --- | --- |
| **Sprint** | **Main objective** | **Expected results** |
| Sprint 2 | Process orders, create invoices, apply promotional codes | Confirm order, create delivery note, apply promotion |
| Sprint 3 | Import and warehouse management | Assign IMEI, bulk import, inventory check |
| Sprint 4 | Send Email & System Security | Send order email, encrypt password, authorize |
| Sprint 5 | Create a receipt | Create import vouchers, update inventory, automatically assign IMEI codes |
| Sprint 6 | Process customer orders & create delivery notes | Order confirmation, shipping, IMEI recording, inventory update |
| Sprint 7 | Advanced Statistics & Reporting | Revenue chart, filter by category, export PDF/Excel report |

### 3.3.8. Sprint 5: Create goods receipt and update goods to the store.

|  |  |
| --- | --- |
| **Content** | **Detail** |
| Target | Allow admin to create import receipt when product is out of stock |
| Development function | - Create an import slip (select category, supplier, quantity, unit price)  - Automatically assign IMEI code to newly imported products  - Update inventory after successful import |
| User Story | - As an admin, I want to create a receipt to replenish stock when the product is running out.  - As an admin, I want each imported product to have its own IMEI code for easy tracking. |
| Expected results | The system can create import vouchers and update inventory quantities by category. |

### 3.3.9. Sprint 6: Process customer orders and create delivery notes

|  |  |
| --- | --- |
| **Content** | **Detail** |
| Target | Process orders from customers and create delivery notes with IMEI information |
| Development function | - List of orders with status "Waiting"  - Confirm or reject orders  - Create delivery note (including IMEI code, original price, promotional price)  - Deduct corresponding inventory |
| User Story | - As admin, I want to confirm the order and export the corresponding goods.  - As admin, I want to track the bombed order from the customer. |
| Expected results | Order is processed, valid delivery note is generated, inventory is deducted, and IMEI information is saved. |

### 3.3.10. Sprint 7: Advanced Statistics and Reporting

|  |  |
| --- | --- |
| **Content** | **Detail** |
| Target | Provide detailed reports on business and inventory activities |
| Development function | - Revenue statistics by day/month/year  - Statistics by category, best-selling products  - Data charting using JFreeChart library  - Export PDF or Excel reports |
| User Story | - As an admin, I want to see the revenue chart to evaluate business performance.  - As an admin, I want to filter reports by category or product. |
| Expected results | Clear revenue charts, easy-to-export reports, support strategic decisions. |

### 3.3.11. Agile Support Tools

|  |  |
| --- | --- |
| **Purpose** | **Recommendation Engine** |
| Work management | Trello, Jira, GitHub Projects |
| Source code management | Git + GitHub/GitLab/Bitbucket |
| Automation (CI/CD) | GitHub Actions, Jenkins |
| Group communication | Group Zalo, Slack, Google Meet |

## 3. 4 . Applying Agile Processes to Sales Management Software

### 3.4.1. Sprint 1: Shopping Cart and Order

|  |  |  |
| --- | --- | --- |
| **STT** | **User Story** | **Priority level** |
| 1 | Show list of available products | High |
| 2 | Add product to cart | High |
| 3 | Edit/remove products from cart | High |
| 4 | Select payment method before clicking 'Order' button | High |
| 5 | Submit order, change status to 'waiting for confirmation' | High |

**Daily implementation plan**

|  |  |
| --- | --- |
| **Day** | **Job** |
| Day 1 | UI design: product list and shopping cart interface |
| Day 2 | Code to display product from database (status 'In stock') |
| Day 3–4 | Add/Edit/Delete products from cart (save locally or DB) |
| Day 5 | Add payment method selection screen (radio button) |
| Day 6–7 | Submit order → save to Orders table with status 'Waiting' |
| Day 8–10 | Test, debug, internal demo |

### 3. 4 .2. Sprint 2: Confirm order and send email

|  |  |  |
| --- | --- | --- |
| **STT** | **User Story** | **Priority level** |
| 1 | As an admin, I want to confirm the customer's order to change it to processing status. | High |
| 2 | As a system, I want to automatically send an order confirmation email after a customer successfully places an order. | High |
| 3 | As a customer, I want to receive order confirmation emails for peace of mind and proof of transaction. | Medium |

**Sprint 2 Goal:**

* Build order confirmation process from admin side.
* Send automatic confirmation email (content includes order code, product, total amount, payment method).
* Integration with email sending library (JavaMail API or SMTP).

**Daily implementation plan**

|  |  |
| --- | --- |
| **Day** | **Job** |
| Day 1 | Admin screen UI design to manage pending orders |
| Day 2 | Function code for confirming orders and updating order status (→ "Confirming") |
| Day 3 | Design the content of the order confirmation email (order code, product, total amount...) |
| Day 4–5 | Integrate email sending functionality using JavaMail API or SMTP |
| Day 6 | Test the order → confirmation → email process (end-to-end) |
| Day 7–8 | Add popup message when confirmation is successful or has errors |
| Day 9–10 | Fix bugs, retest, internal demo for Product Owner |

## 3. 5 . Advantages and disadvantages of Agile model

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Advantage** | **Disadvantages** |
| Flexibility | Easily adapt to changing requirements throughout the development process. | Can lead to too frequent changes, affecting progress if not well controlled. |
| Customer Feedback | Continuously soliciting customer feedback after each Sprint helps make the product more realistic. | Requires regular engagement from the client – sometimes clients just don't have enough time. |
| Progress and quality | Detect errors early, gradually improve quality through each development cycle. | Easily fragmented if the team is inexperienced or does not manage the Product Backlog well. |
| Group communication | Enhance internal communication and knowledge sharing among members. | Lack of clear formal documentation if only focusing on oral communication. |
| Fast market response | Products can be launched early, explored and improved according to actual needs. | Not suitable for large-scale, fixed-requirement projects that require tight control. |
| Risk Management | Regular testing and adjustment helps reduce overall risk. | Without a clear and committed PO, the team can get off track or lack long-term direction. |
| Project Documentation | Focus on working software rather than redundant documentation. | Stakeholders (e.g. investors) may need detailed documentation to evaluate and approve the project. |

# Chapter IV : Software Architecture and Data Modeling

## 4.1. Software architecture

### 4.1.1. Java programming language

**Java is a high-level, object-oriented programming language** developed by Sun Microsystems (now part of Oracle). First released in 1995, Java quickly became one of the most popular programming languages in the world due to its cross-platform capabilities, high security, ease of use, and rich ecosystem.

**Outstanding features of Java:**

* **Object-Oriented Programming (OOP)** Java follows the object-oriented programming model, which helps organize source code into reusable, easily maintainable, and efficiently managed objects.
* **Platform Independence – "Write Once, Run Anywhere" (WORA)** Thanks to the Java Virtual Machine (JVM), Java can run on many different operating systems without modifying the source code.
* **Automatic Memory Management – "Garbage Collection" Java** has a Garbage Collector system that automatically frees up unused memory, helping to minimize memory management errors.
* **Rich Libraries & Diverse Frameworks Java** has a large ecosystem with thousands of libraries supporting web, mobile, AI, Big Data, etc. application development.
* **High Security Java** provides strong security features like access control, data encryption, which helps protect applications from threats.
* **Java Multithreading support** allows parallel processing of multiple tasks, helping to increase performance for applications that need to process multiple tasks at the same time.

### 4.1.2. Layered Architecture

**BUS, DAO, DTO, GUI** models are often used in enterprise software programming, especially when building applications in a layered architecture. This is an approach that follows the principles of object-oriented programming (OOP), helping the system have a clear structure, easy to maintain, extend and reuse.

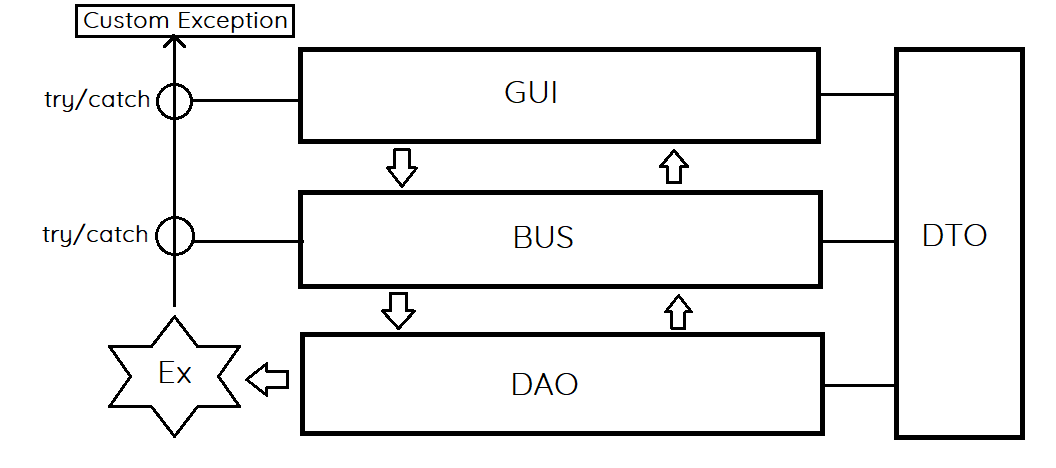
A software system is divided into many layers, each layer undertakes a specific task, creating separation and flexibility in design. Thanks to encapsulation, each layer provides only the necessary functions, hiding the implementation details inside, helping to increase the security and stability of the application.

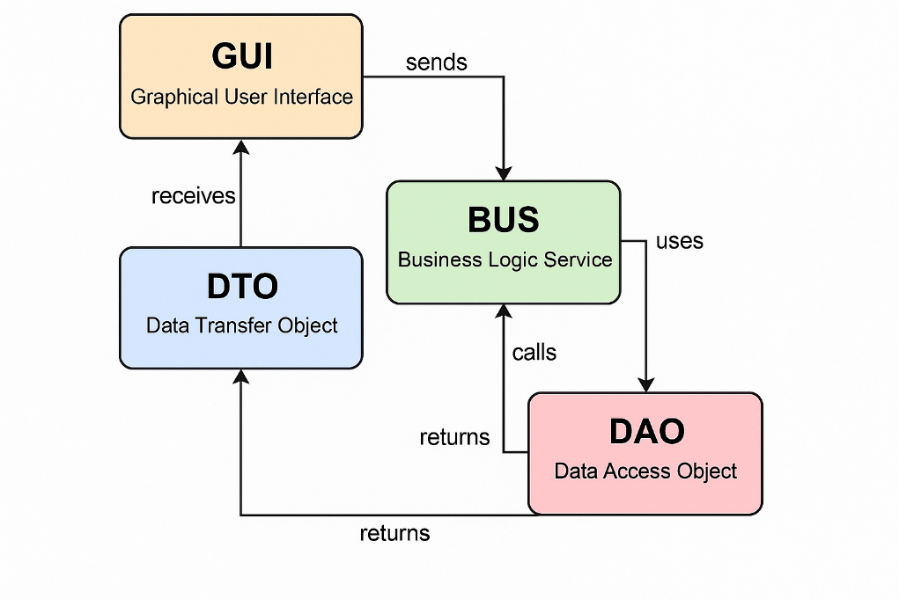
Inheritance allows child classes to reuse methods and properties from the parent class, reducing code duplication and supporting functionality extension without changing the original structure. As a result, this model helps build highly flexible software systems that are easy to update and upgrade.

Additionally, polymorphism ensures that the same method can behave in different ways depending on the class of object it is used in. This makes the code easier to read, easier to maintain, and allows the application to be expanded flexibly without having to modify a lot of the old code.

Finally, abstraction helps programmers focus on core concepts, rather than implementation details. It allows defining abstract methods and classes, which subclasses can implement in detail to suit each real-world use case.

Thanks to the characteristics of OOP, **BUS, DAO, DTO, GUI models** not only help to organize the system scientifically, but also ensure that the application is easy to expand, easy to maintain and has high performance in the enterprise environment.





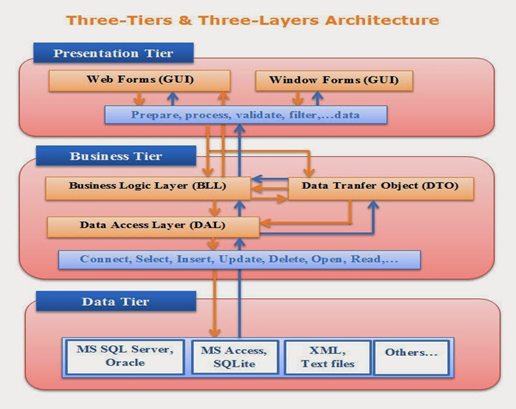


Image 4: Simulation of how the classification model works

**BUS (Business Logic Layer) – Business processing layer**

* Contains business rules, logic processing before sending data to GUI or DAO:
* Act as an intermediary, ensuring that the application works properly according to business requirements.
* Does not interact directly with the database, but gets data from DAO, processes it and returns it to the GUI.

For example:

*import* java . util . List ;   
  
public. public *class* UserBUS {  
 private *UserDAO* userDAO ;  
  
 public UserBUS () {   
userDAO = new *UserDAO* () ;   
}  
  
 public *List* < *UserDTO* > getUsersWithLongNames () {  
 *List* < *UserDTO* > users = userDAO . getAllUsers () ;   
users . removeIf ( user *->* user . getName () . length () < 10 ) ;  
 return users ;   
}   
}

**DAO (Data Access Object) – Data Access Layer**

* Responsible for communicating with the database, performing CRUD operations (Create, Read, Update, Delete)
* Completely separate processing logic from data storage system, making maintenance easier.
* Provides methods to retrieve data from the database and convert it into DTOs for use.

For example:

*import* java . sql . \* ;   
*import* java . util . ArrayList ;   
*import* java . util . List ;   
  
public. public *class* UserDAO {  
 private *Connection* conn ;  
 public UserDAO () {   
conn = *DatabaseConnection* . connect () ; }  
 public *List* < *UserDTO* > getAllUsers () {  
 *List* < *UserDTO* > users = new *ArrayList* <> () ;  
 *String* sql = " SELECT \* FROM users " ;

try ( *Statement* stmt = conn . createStatement () ;  
 *ResultSet* rs = stmt . executeQuery (sql)) {  
 while (rs . next ()) {   
users . add ( new *UserDTO* (rs . getInt ( " id " ) , rs . getString ( " name " ))) ; }   
} catch ( *SQLException* e) {   
e . printStackTrace () ;   
}  
 return users ;

}   
}

**DTO (Data Transfer Object) – Data transfer object:**

* Data encapsulation, which allows information to be exchanged between layers without directly interacting with the database.
* Contains no processing logic, only properties (fields), constructors, getters/setters.
* Helps reduce dependencies between layers and makes data transfer easier.

For example:

public *class* UserDTO {  
 private *int* id ;  
 private *String* name ;  
 private *String* email ;  
  
 public UserDTO ( *int* id , *String* name , *String* email ) {  
 *this* . id = id ;  
 *this* . name = name ;  
 *this* . email = email ;   
}  
  
 public *int* getId () { return id ; }  
 public *String* getName () { return name ; }  
 public *String* getEmail () { return email ; }   
}

**GUI (Graphical User Interface) – Interface layer**

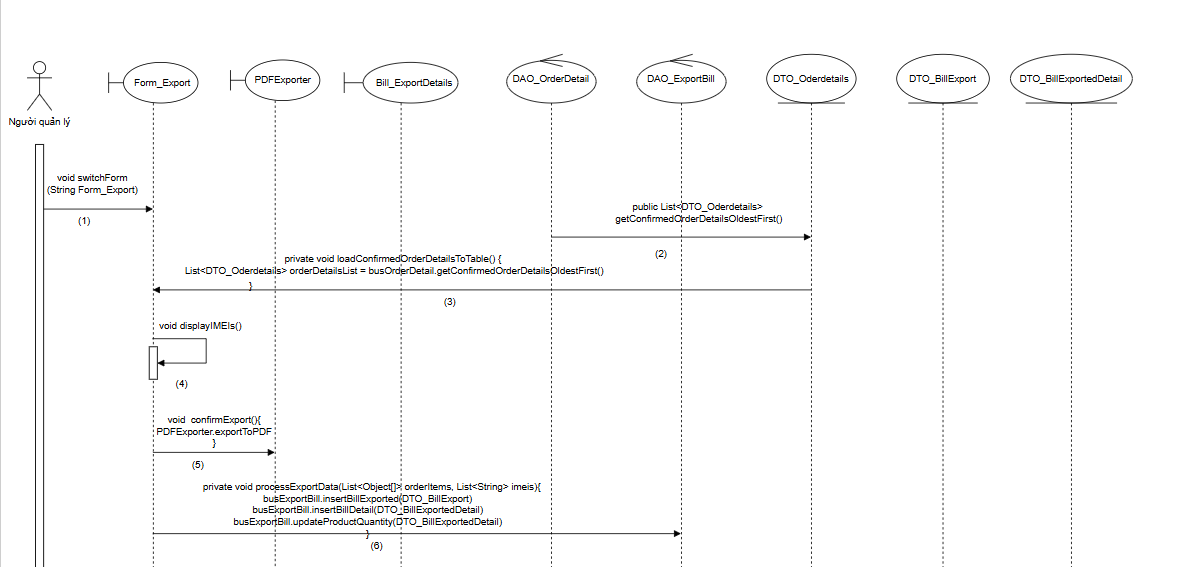
* Display information and interact with users, allowing users to enter, view, and edit data.
* Contains no processing logic, only receives data from the BUS for display.
* It can be a web application (HTML, CSS, JavaScript) or a graphical interface (Java Swing, JavaFX).

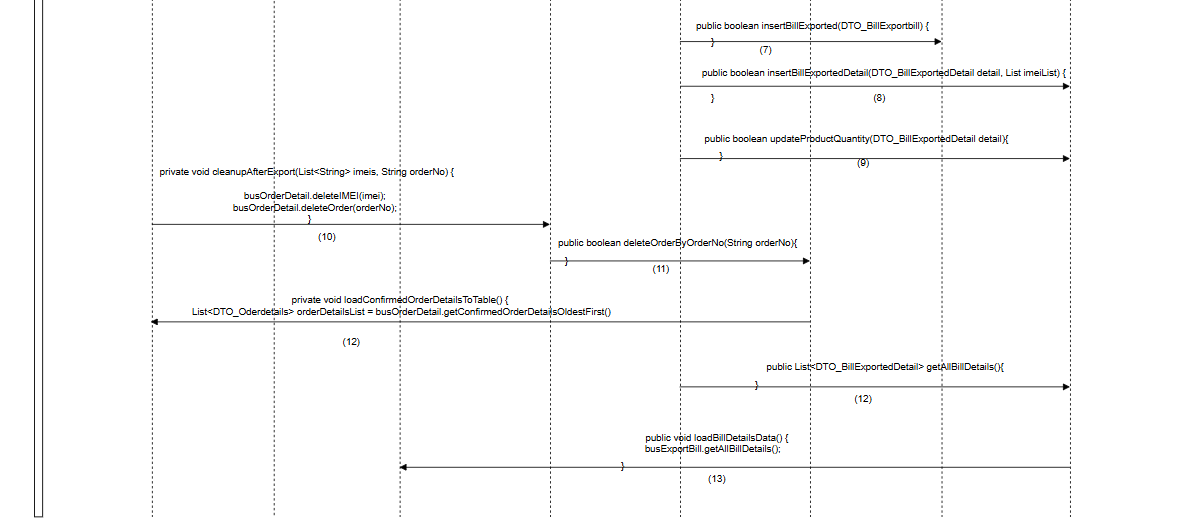
For example:

*import* javax . swing . \* ;   
*import* java . util . List ;   
  
public. public *class* UserGUI {  
 private *UserBUS* userBUS ;  
  
 public UserGUI () {   
userBUS = new *UserBUS* () ;  
 showUsers () ;   
}  
 private *void* showUsers () {  
 *List* < *UserDTO* > users = userBUS . getUsersWithLongNames () ;  
 for ( *UserDTO* user : users) {  
 *System* . out . println ( " " + user . getName () + " - Email: " + user) ;   
}   
}  
 public static *void* main ( *String* [] args ) {  
 new *UserGUI* () ;   
}   
}

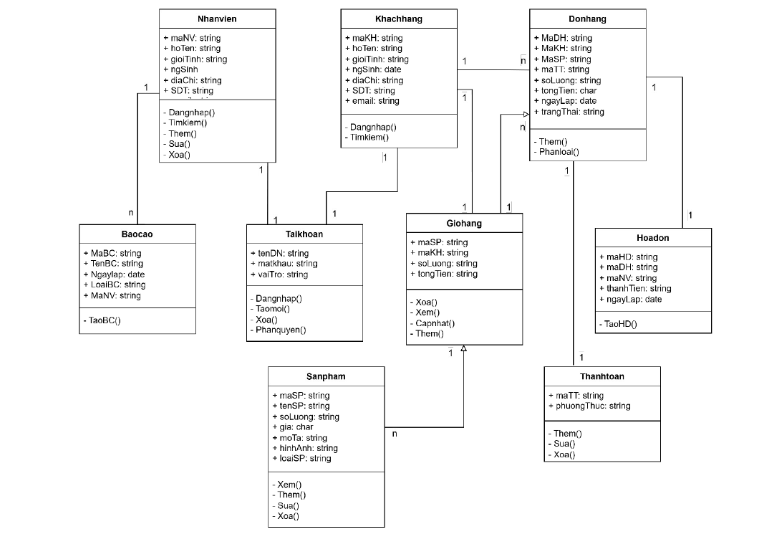
## 4.2. Data modeling

### 4.2.1. Sequence diagram for the export function (Here our group chooses this function as the main point to present)





### 4.2.2. Class diagram



## 4.3. Interface design and specification

### 4.3.1. Customer interface

#### a) Login function

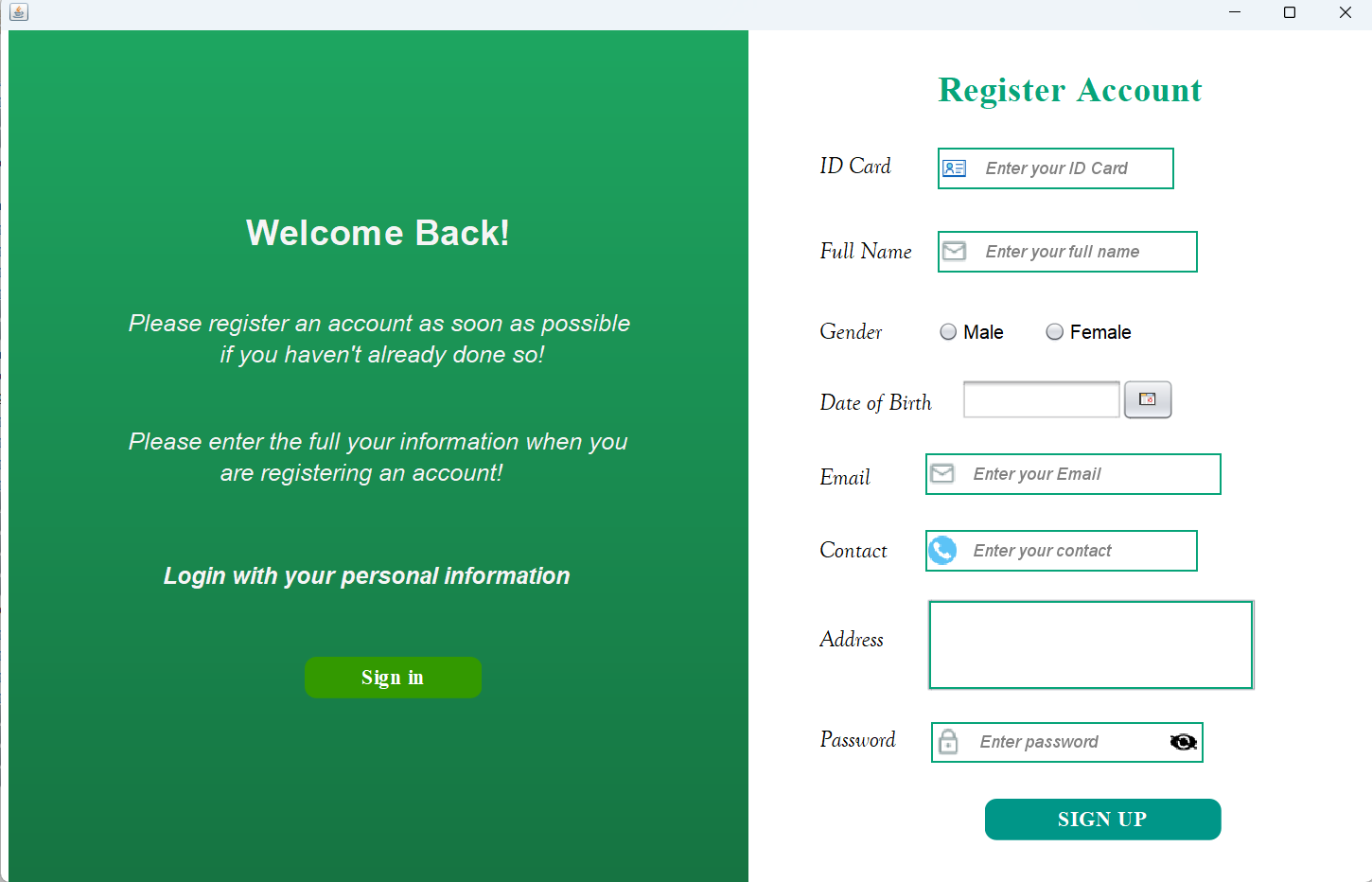


Image 5: User account registration interface

**Step 1 :** When entering the system, if the user does not have an account, they will go to the " *Sign up* " section to register for an account.

*Sign up”* section , customers will enter information such as: ID Card (CCCD Number), Full Name (Full Name), select gender from Radio Button, enter Email, Contact (Phone Number), Address (Address), select date of birth from Date chooser and enter Password (Password)

**Step 2 :** Passwords are encrypted using bcrypt to ensure the security and integrity of customers' personal information, helping to protect against attacks and prevent the risk of data theft.

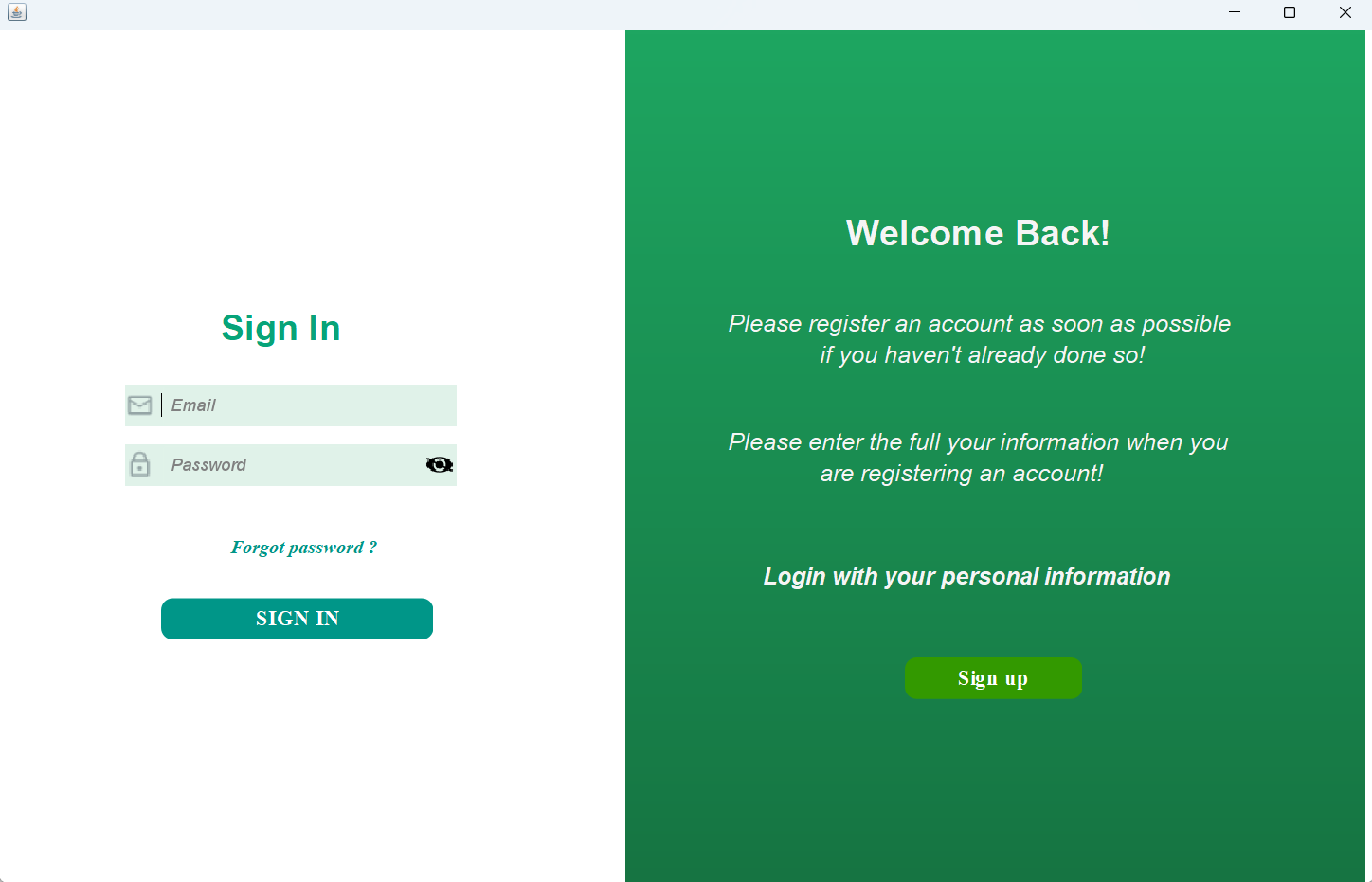


Image 6: User login interface

**Step 1 :** After completing registration, customers will select **Sign in** to log into the system.

**Step 2: At the** login interface, users need to fully enter Email and Password, ensuring the following requirements are met:

* **Email:** Must be in correct format, authentic and match the previously registered email. Otherwise, the system will display an error message.
* **Password:** Must be the password set during initial registration.

**B3:** If the customer enters missing information or does not meet the above requirements, the system will immediately report an error, ensuring accuracy and security during the login process.

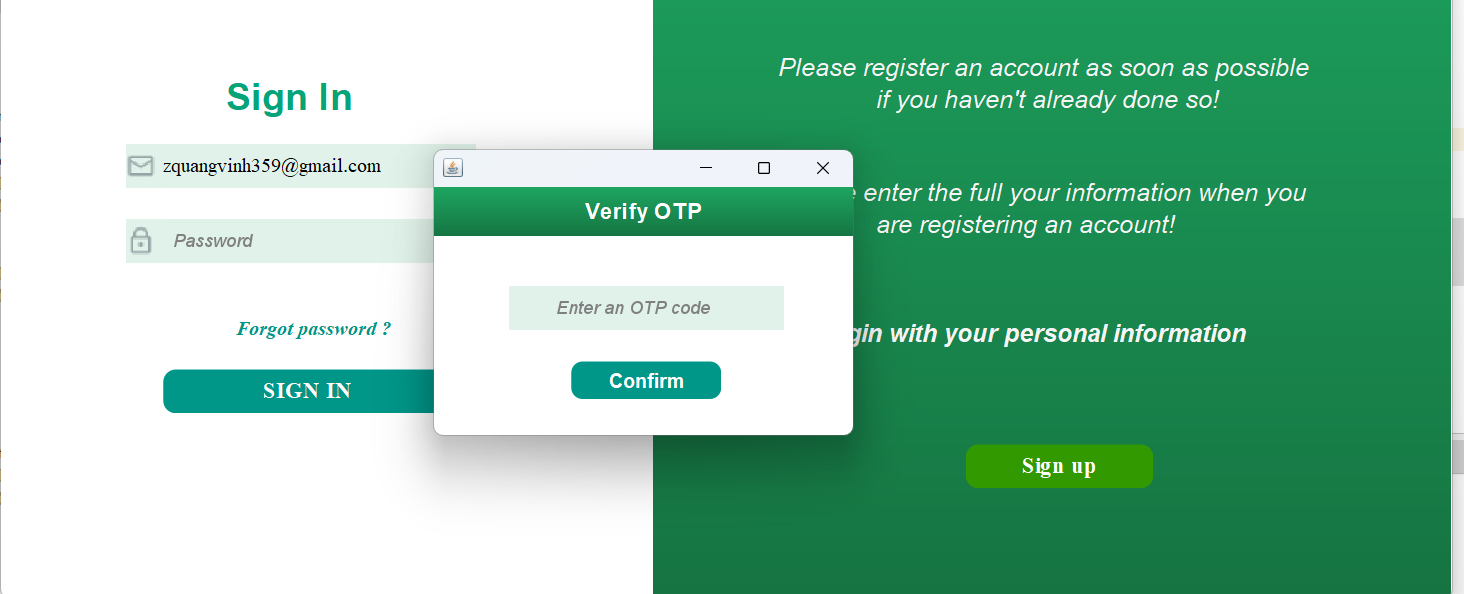


Image 7: User's OTP confirmation code input interface

If the customer forgets the password and clicks on the "Forgot Password?" button, the system will immediately send a 6-digit OTP code to the previously registered email, ensuring the authentication process is safe and accurate.

**OTP Processing:**

* If you enter the wrong OTP code, the system will display an error message and ask you to re-enter it to ensure security.
* If the OTP code is entered correctly, the system will successfully confirm and allow the user to reset a new password, helping to restore access to the account in the most convenient way.

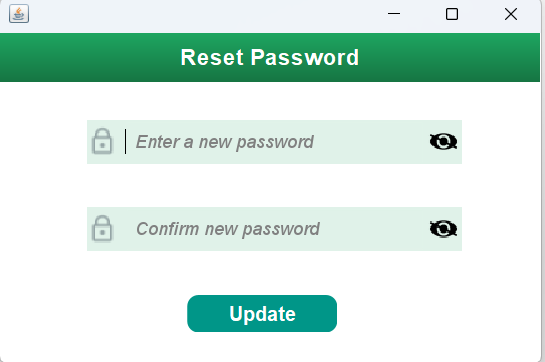


Image 8: New password entry interface

Here, the customer will enter the new password and confirm the new password again. If the user authenticates that the new password does not match, the system will report an error.   
→ From there, the user will use the new password to log in to the system.

#### b) Home page interface

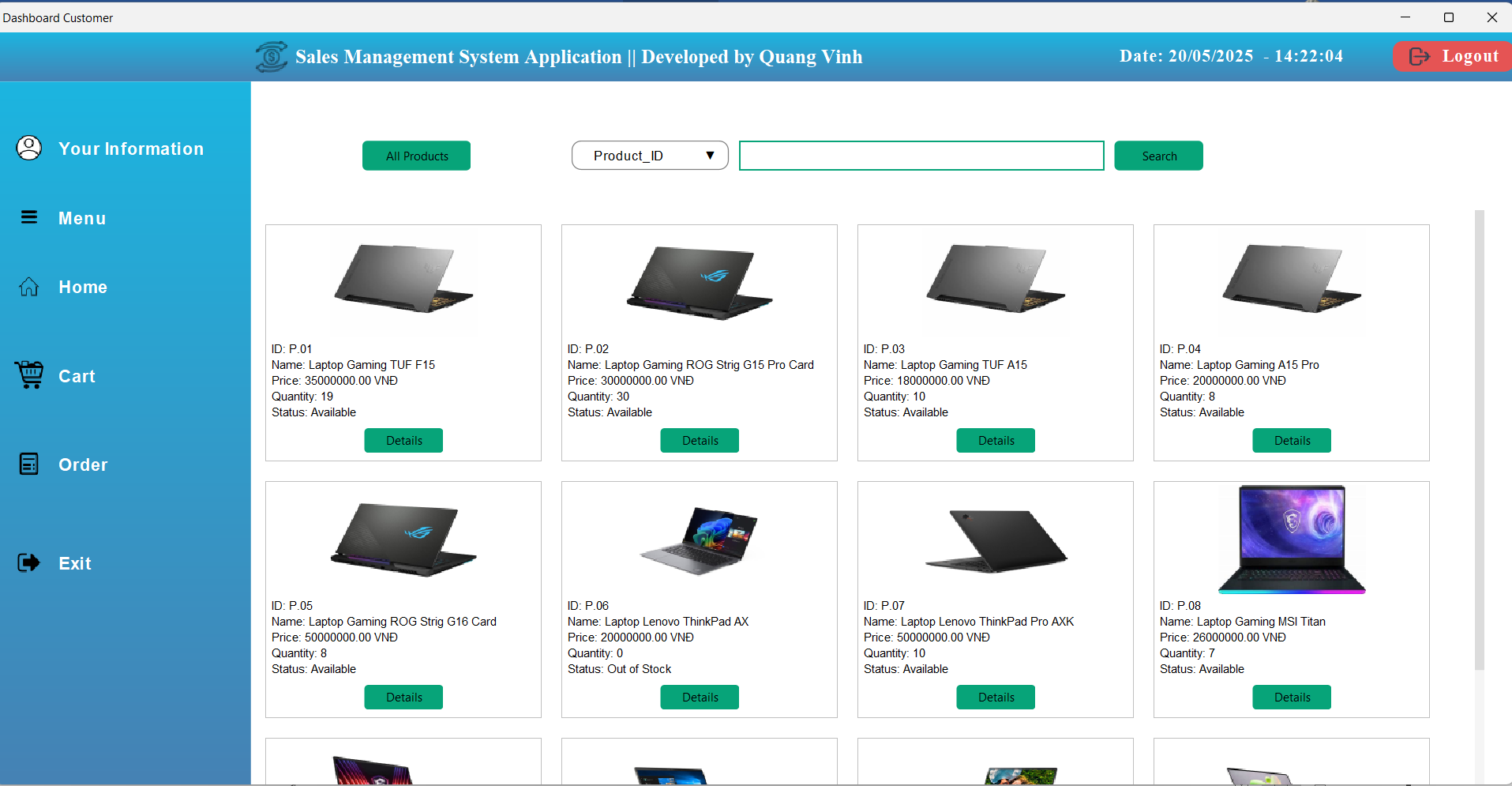


Image 9: Home page interface when user logs in

After successfully logging in, users will be redirected to the home page **-** which opens up an intuitive and convenient shopping space.

Here, customers can easily browse through the product list with full details:

* Product code: identifies each product precisely.
* Product name: helps buyers quickly identify the desired item . Quantity: clearly displays the number of products remaining in stock.
* Selling price: update price of each type, with promotions if any.
* Stock status: shows whether the product is in stock or out of stock, helping buyers make an easy decision.

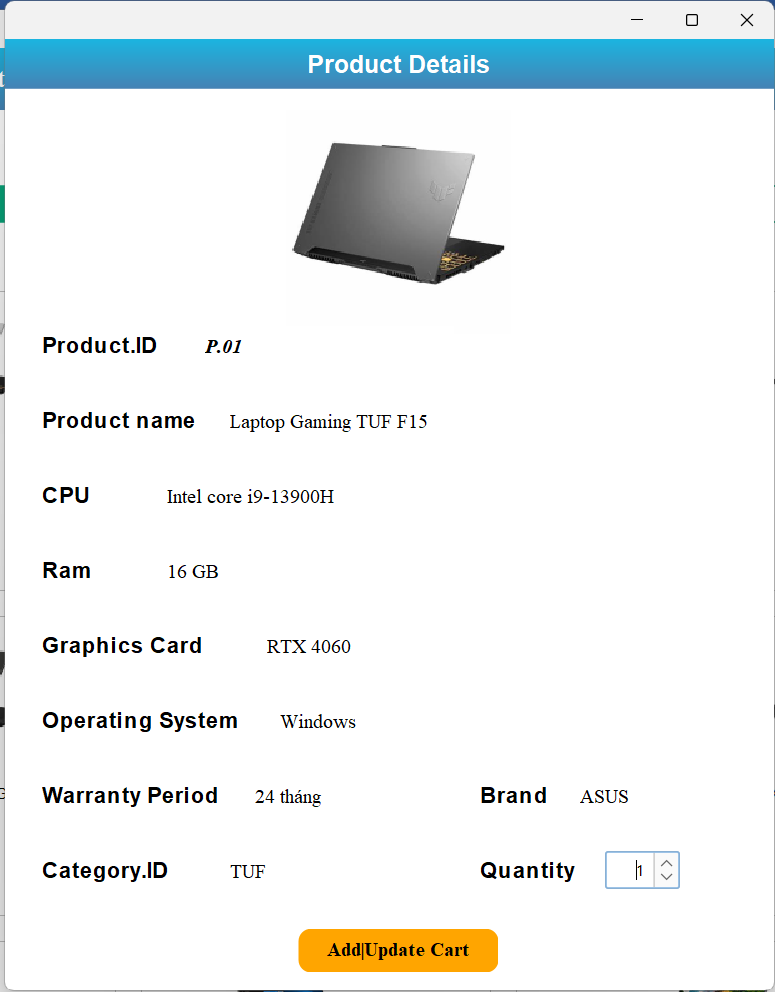


Image 10: Product details

When customers click on the **"Details" button** of a product, the system will display full details, helping them have a comprehensive view before deciding to buy.

**Product** information includes:

* **Product code:** uniquely identify each item.
* **Product name:** helps buyers quickly identify the desired item.
* **Detailed configuration:** includes technical parameters such as CPU, RAM, Graphics Card, and other important information.
* **Manufacturer:** brand name guarantees product quality.
* **Warranty period:** displays the warranty policy that comes with the product.
* **Select purchase quantity:** Customers can select the quantity of products based on the current inventory. The system will automatically limit, ensuring that buyers cannot choose more than the quantity in stock, helping the ordering process to be accurate and avoid errors.

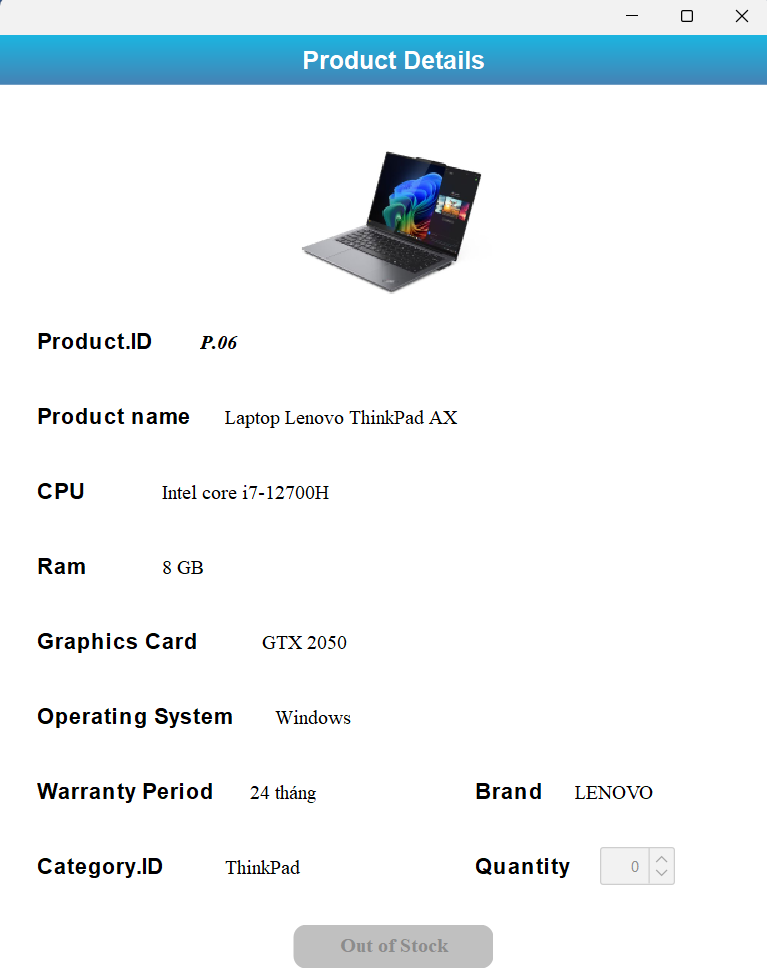


Image 11: Product information when out of stock

When customers view details of an out-of-stock product, the system still displays full product information, helping them have a comprehensive view before making a purchase decision.

However, to ensure the accuracy and integrity of the ordering process:

* The " **Add | Update Cart** " button will automatically change to " **Out of Stock** " and be locked, preventing the product from being added to the cart.
* The product quantity selection field will also be disabled, ensuring that customers cannot order more than the actual quantity in stock.

#### c) Shopping cart management function

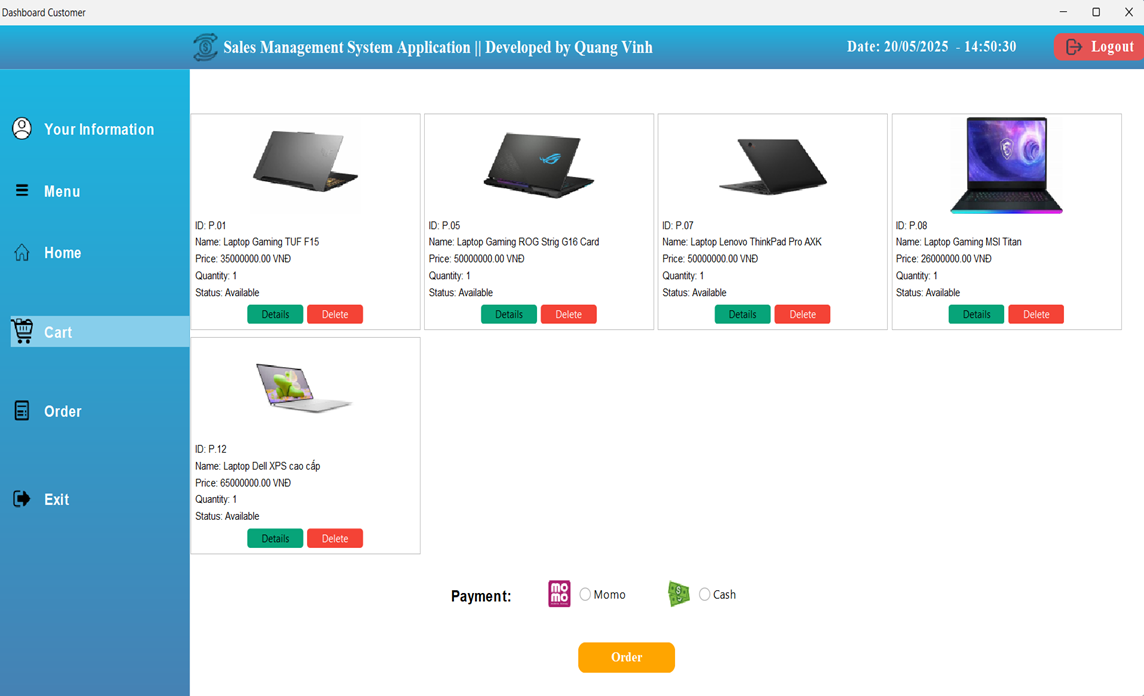


Image 12: User shopping cart interface

After customers choose the desired products with the desired quantity and add them to the cart, the system will display a list of ordered items with full information:

* **Product code:** uniquely identifies each product.
* **Product name:** helps buyers quickly identify their favorite items.
* **Quantity:** shows the number of products the customer has chosen to buy.
* **Stock status:** clearly displays whether the product is in stock or out of stock.

**View cart details:** Here, customers can view all information related to the ordered product, including: product code, product name, specifications, warranty period, and selected quantity.

**Purchase decision:**

* If you no longer want to buy, you can remove the product from your shopping cart by clicking the " **Delete** " button.
* If you want to change the purchase quantity, you can return to the home page, select the product you want to adjust, select the new quantity and click the " **Add | Update Cart** " button to update the shopping cart.

Once the customer confirms the order and selects the desired quantity of products, the system will ask them to choose one of two payment methods:

* **Momo:** Convenient and fast payment via e-wallet.
* **Cash:** Pay directly upon receipt.

**Ordering process:**

* If the customer has not selected a payment method and clicks " **Order** ", the system will immediately display an error message, asking them to complete the payment step before placing the order.
* Once payment method is confirmed, the order will be processed, ensuring all transactions are transparent and accurate.

#### d) Order information viewing function



Image 13: Order information interface

After the customer successfully places an order, the system will automatically create an order, ensuring that all transactions are recorded transparently and accurately.

**Order details:**

* Order No (Order Number) – each order has a unique identifier, which helps distinguish between different orders.
* Customer ID (Customer Code) – ensures the order belongs to the customer, making it easy to manage shopping history.
* Total Price – shows the total amount that the customer needs to pay for the order.
* Date Order, Time Order – records the exact time when an order was created, ensuring fast processing.

**Process after ordering:**

* Orders will be updated in the system, helping customers track status.
* Order information can be easily retrieved for checking or post-purchase support.

→ By automatically creating and storing orders, the system helps customers have a safe, accurate and convenient shopping experience!

**Note : Once an order is created, customers cannot cancel or change products themselves** , nor adjust the ordered quantity to ensure the processing is accurate and transparent.

**Order cancellation process:**

**B1:** If customers need to cancel their order, they need to contact Admin directly for support.

**B2:** Admin will review the request, check the order status and take appropriate processing steps.

**B3:** The system will update the order status immediately after the Admin confirms the cancellation request.

→ This process helps ensure orders are closely managed, avoiding confusion or errors in transactions and shipping!

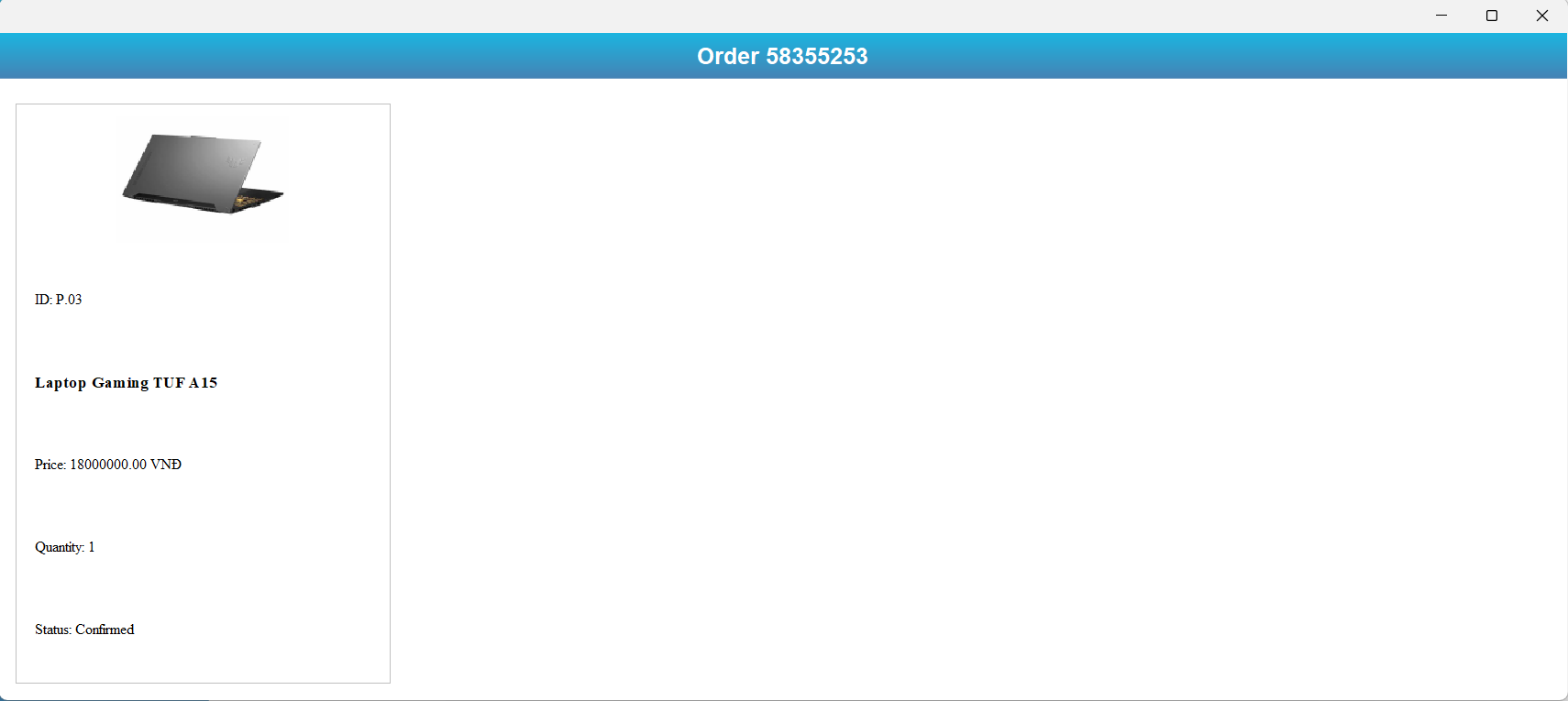


Image 14: User order status interface

If customers want to see what they ordered as well as the order status, click on **"View Details"** to see the orders that have been created.

### 4.3.2. Admin section interface

#### a) Login function

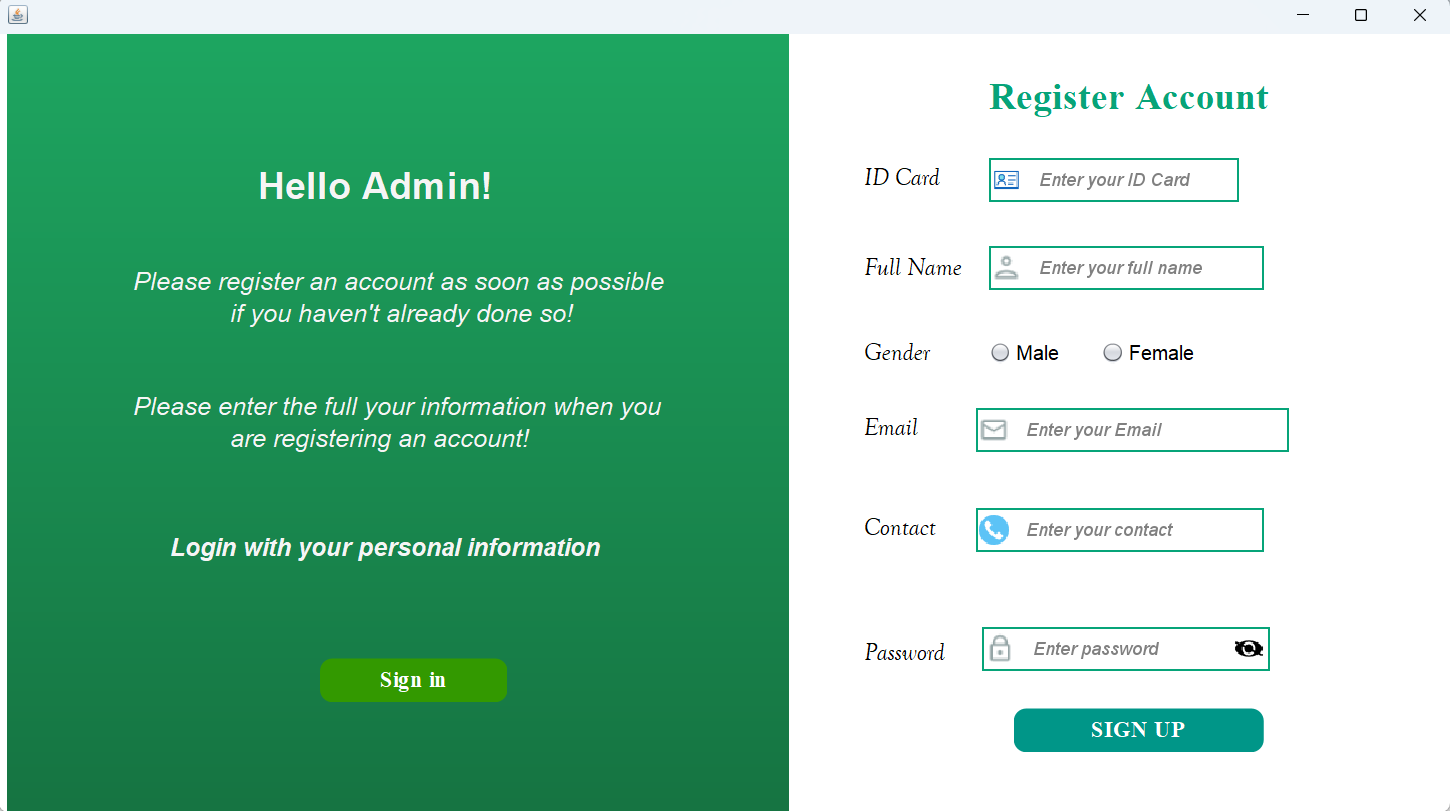


Image 15: Admin account registration function interface

When entering the system, if the administrator does not have an account, he/she will go to the **"** *Sign up* **" section** to register for an account.

*Sign up”* section , the admin will enter information such as: ID Card (ID card number or issued ID), Full Name (Full name), select gender from Radio Button, enter Email, Contact (Phone number) and enter Password (Password)

**Note:** Passwords are encrypted using bcrypt to ensure the security and integrity of customers' personal information, helping to protect against attacks and prevent the risk of data theft.

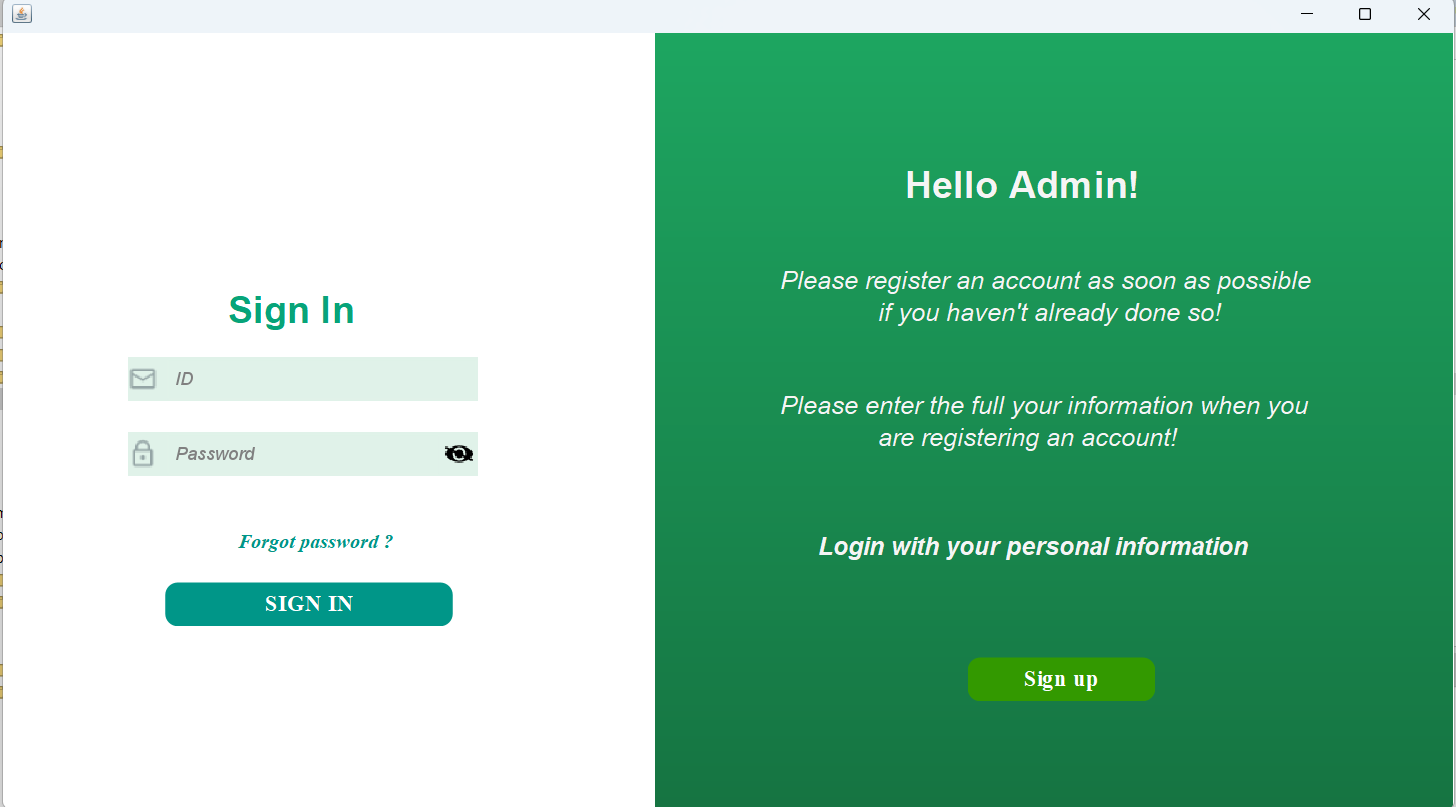


Image 16: Admin account login function interface

After completing registration, customers will select Sign in to log into the system.

At the login interface, users need to fully enter Email and Password, ensuring the following requirements:

* ID: Must be the owner and match the previously registered ID. Otherwise, the system will display an error message.
* Password: Must be the password set during initial registration.

→ If Admin enters missing information or does not meet the above requirements, the system will immediately report an error, ensuring accuracy and security during the login process.

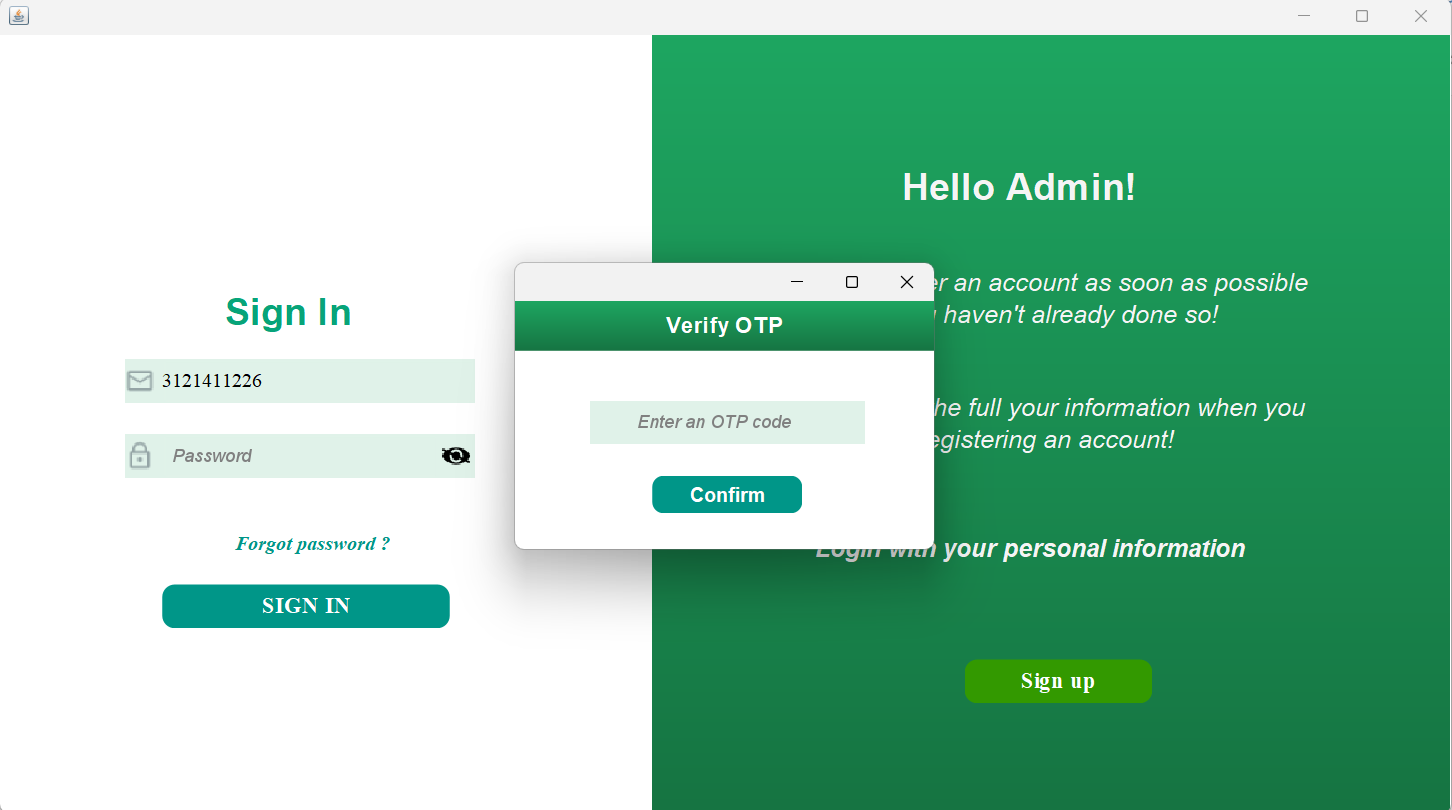


Image 17: Interface when Admin forgets account password

If the administrator forgets the password and clicks on the “Forgot Password?” button with the condition that the registered ID must be entered, the system will immediately send a 6-digit OTP code to the previously registered email, ensuring the authentication process is safe and accurate.

**OTP Processing:**

* If you enter the wrong OTP code, the system will display an error message and ask you to re-enter it to ensure security.
* If the OTP code is entered correctly, the system will successfully confirm and allow the user to reset a new password, helping to restore access to the account in the most convenient way.

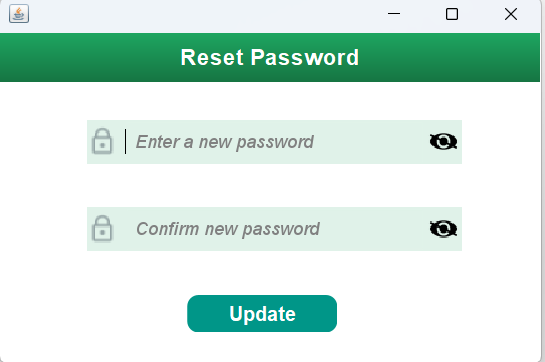


Image 18: Admin password reset interface

Here, the administrator will enter the new password and confirm the new password again. If the user authenticates that the new password does not match, the system will report an error.

→ From there, the user will use the new password to log into the system.

#### b) Home page interface



Image 19: Admin interface after login

On this home page interface, there will be an overview of system data including:

* **Total Customer:** Total number of customers
* **Total Orders:** Total number of orders
* **Total Bill Exports:** Total number of export invoices
* **Total Warranties:** Total number of warranties
* **Total Supplier:** Total number of suppliers
* **Total Categories:** Total number of laptop lines
* **Total Products:** Total number of product types
* **Total Bill Imports:** Total number of import bills

The chart provides an overview of the number of products imported and sold, helping to visually evaluate business performance and manage inventory.

**Meaning of data:**

* If the quantity of products exported is lower than the quantity imported, this shows that there is still a lot of inventory, there is a risk of unsold products and it is necessary to consider a reasonable consumption plan.
* On the contrary, if the sales volume increases, it proves that the product is attractive in the market, helping businesses optimize their import strategy.
* From chart analysis, businesses can:
* Make decisions to adjust imports to avoid excess inventory.
* Implement promotional programs to stimulate consumer demand for unsold products.
* Improve business plan, import goods to suit actual needs.
* Chart design helps provide visual data, supporting businesses to effectively manage goods and improve business performance!

#### c) Customer management function

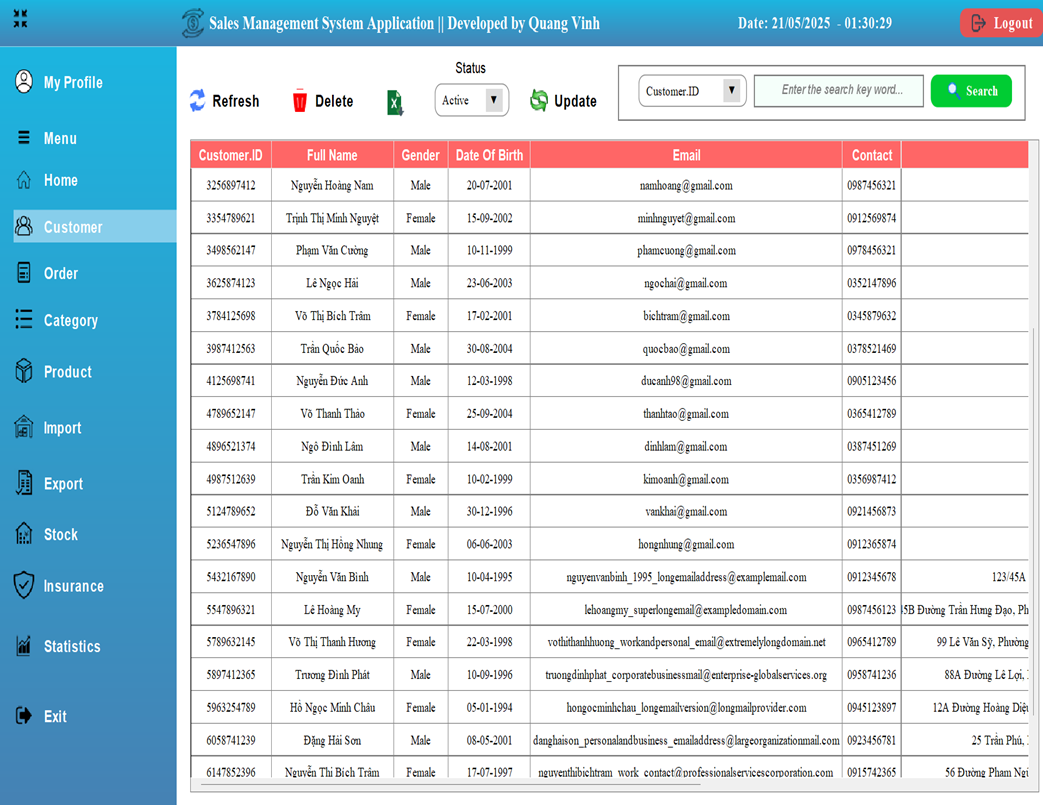


Image 20: Admin's customer management interface

At the customer management interface, the system will display full personal information of each customer as soon as they register an account, except for the password, to ensure security.

**Features for Admin:**

* Delete customer – remove accounts from the system when necessary.
* Update account status – change accounts to “Inactive” status for customers who repeatedly cancel orders, reducing fraud.
* Quickly search for customers – supports searching by ID, full name, phone number or email, helping Admin manage more effectively.
* Export data to Excel file - create a list of grateful customers, support analysis and launch appropriate promotions, optimize customer care strategies.
* The system helps Admin manage easily, filter out potential customers and improve service experience.

#### d) Order management function

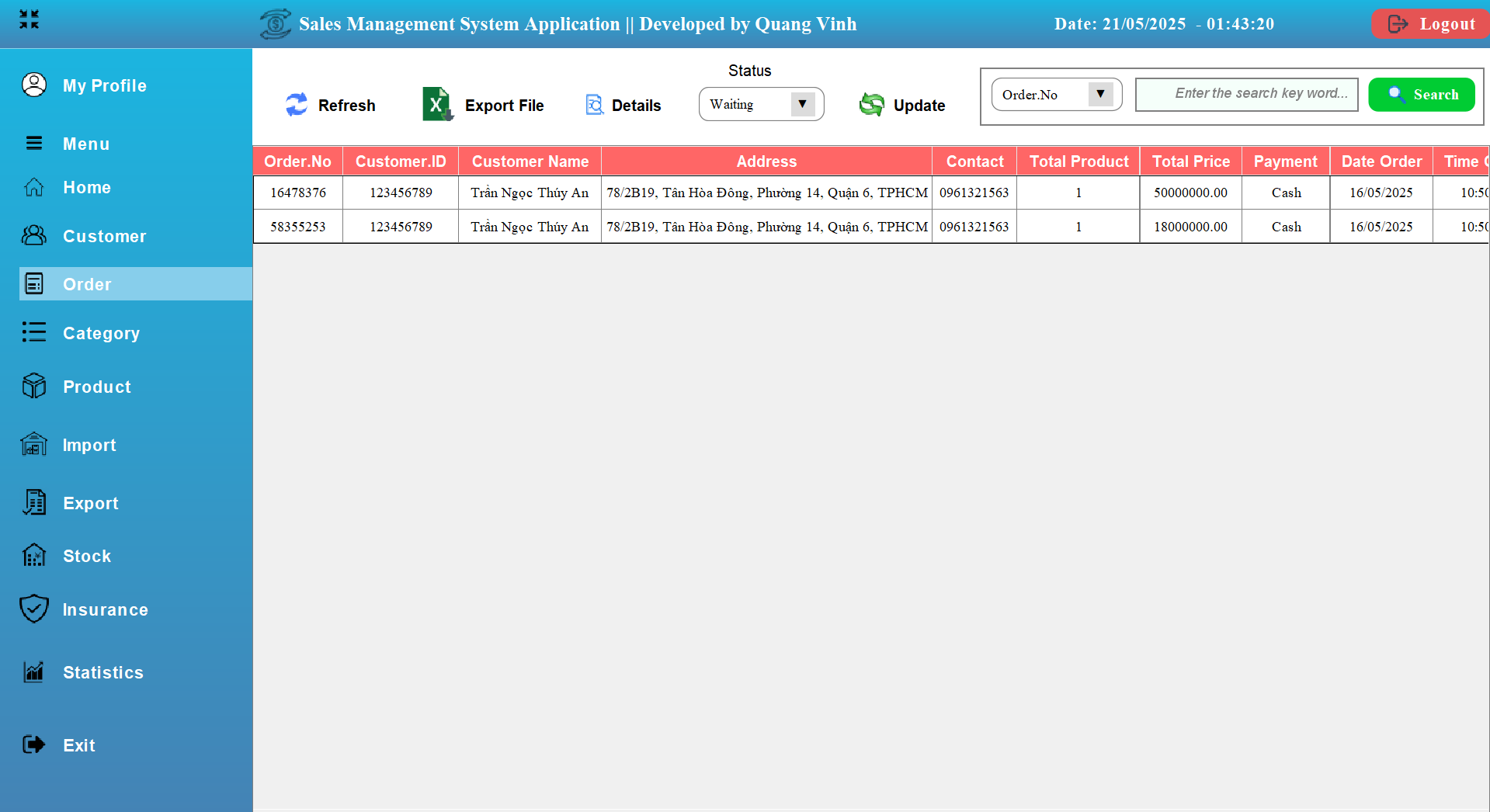


Image 21: Admin order management interface

At **the order management interface** , the **Admin** will:

* View order list – Displays order details by status.
* Search for orders – Filter orders by Order No, Customer ID, or Customer Name. Update status – Move orders from Waiting → Confirming
* Export data – Save order list to Excel file for analysis.
* View order details – Displays product, customer and total order value information.

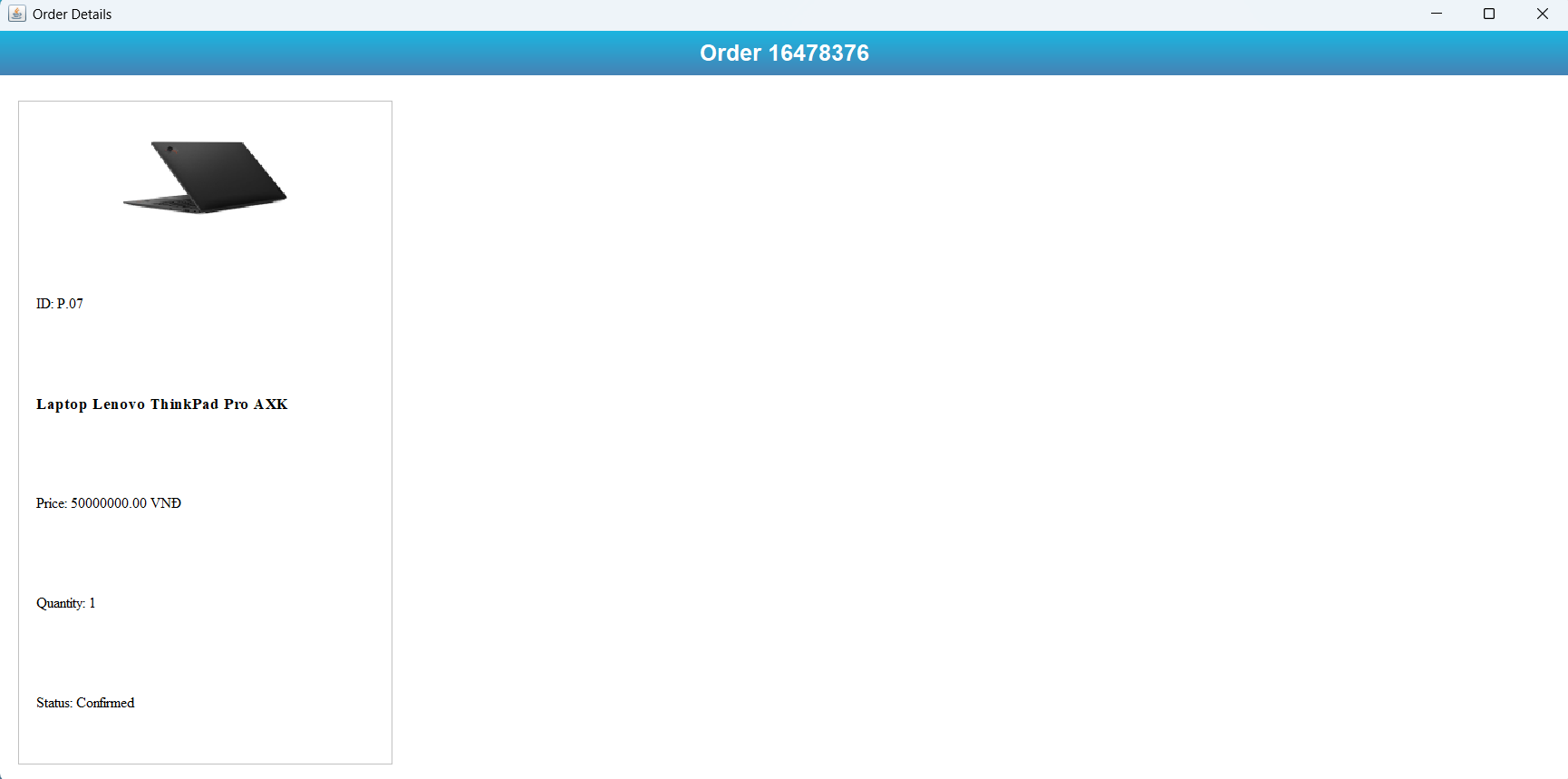


Image 22: Admin order detail interface

The order details interface shows general information of the order placed by **the customer:**

* **Product code:** uniquely identifies each product.
* **Product name:** helps buyers quickly identify their favorite items.
* **Total product price:** calculated by price of 1 product X number of products in the order
* **Quantity:** shows the number of products the customer has chosen to buy.
* **Status:** shows the order status is Waiting or Confirmed by Admin processing.

#### e) Category management function

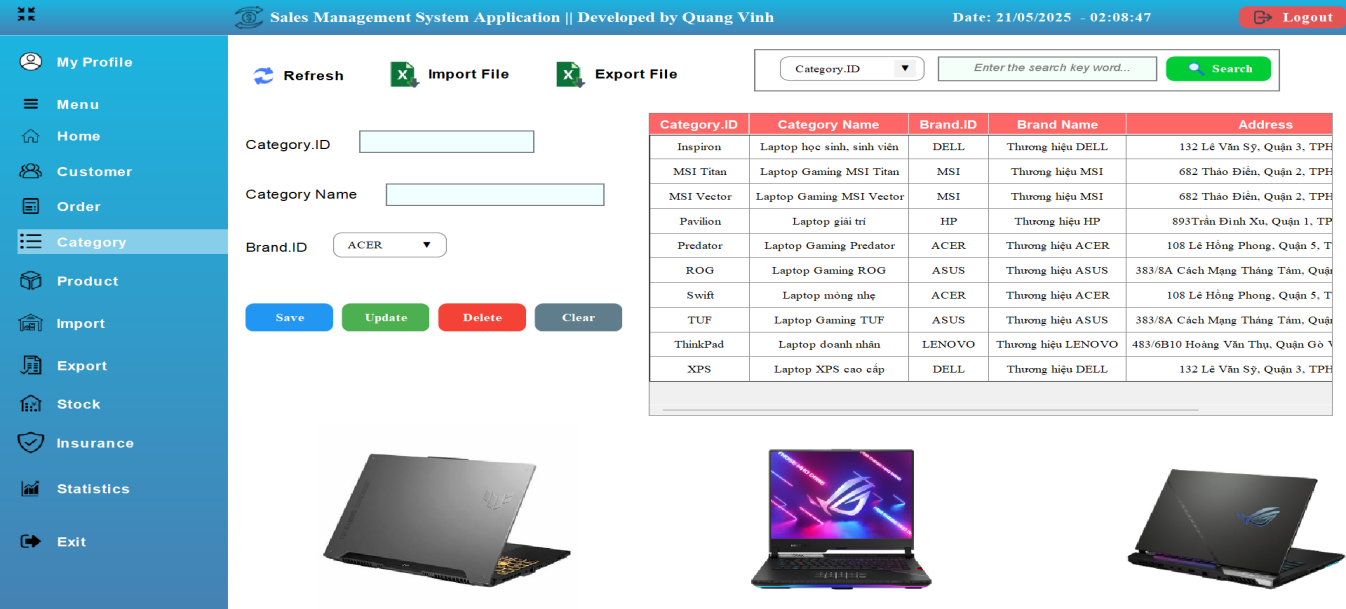


Image 23: Admin Category Management Interface

The Category Management interface helps Admin organize, update and control product category information effectively. The system provides tools for searching, editing and exporting data, ensuring flexible management and optimizing sales inventory.

**Key Features:**

* Add category – Admin can create new product category with full information. Edit category – Update category information when needed.
* Delete Category – Remove categories that are no longer in use.
* Search catalogs – Search catalogs by catalog code, catalog name or supplier.
* Export categories to Excel file – Save category list for reporting and analysis.
* Import catalog from Excel file – The system supports importing data from Excel file, helping to update catalog in bulk without manual entry.

→ The interface is optimally designed, helping Admin manage product categories flexibly, effectively and quickly.

#### f) Product management function



Image 24: Product Management interface helps Admin track

The Product Management interface helps Admin track, update and organize product lists effectively. The system supports adding, editing, deleting, searching and exporting products from Excel files, ensuring sales management is quick and accurate.

**Key Features:**

* Add product – Create new product with full information: Product code, name, specifications, price, image.
* Edit Product – Update product information as needed.
* Delete Product – Remove products that are no longer available in stock.
* Product Search – Look up products by Product Code, Product Name or Supplier.
* Export products to Excel file – Save product list for reporting and statistics.

#### g) IMEI management function (sub-function)

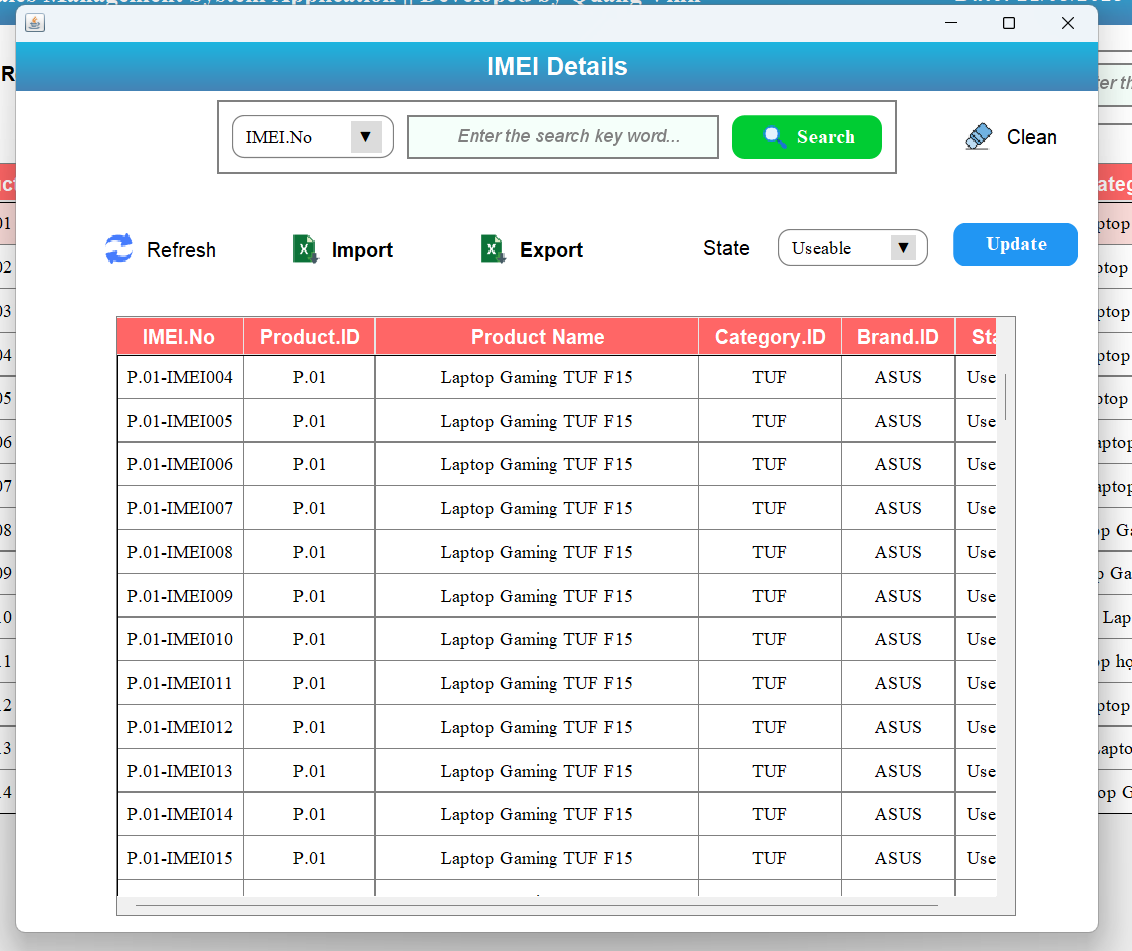


Image 25: IMEI Management interface helps Admin monitor

The IMEI Management interface helps Admin track, manage and control the IMEI code of each product, ensuring uniqueness and accuracy in the system. Supports adding, editing, deleting, searching, importing/exporting IMEI from Excel files, helping to optimize the sales management process.

**Key Features:**

* Add IMEI – Assign IMEI code to new product or add IMEI code to the system.
* Edit IMEI – Update IMEI code information when changes are needed.
* Erase IMEI – Remove invalid or unused IMEI codes.
* IMEI Search – Look up IMEI code by product code or IMEI code.
* Import IMEI from Excel file – The system supports importing IMEI list, helping to manage in bulk quickly.
* Export IMEI to Excel file – Save IMEI code list for reporting and inventory purposes.

→ This interface helps Admin manage IMEI codes accurately, efficiently and quickly, ensuring products have unique information and are easy to retrieve.

#### h) Import management function

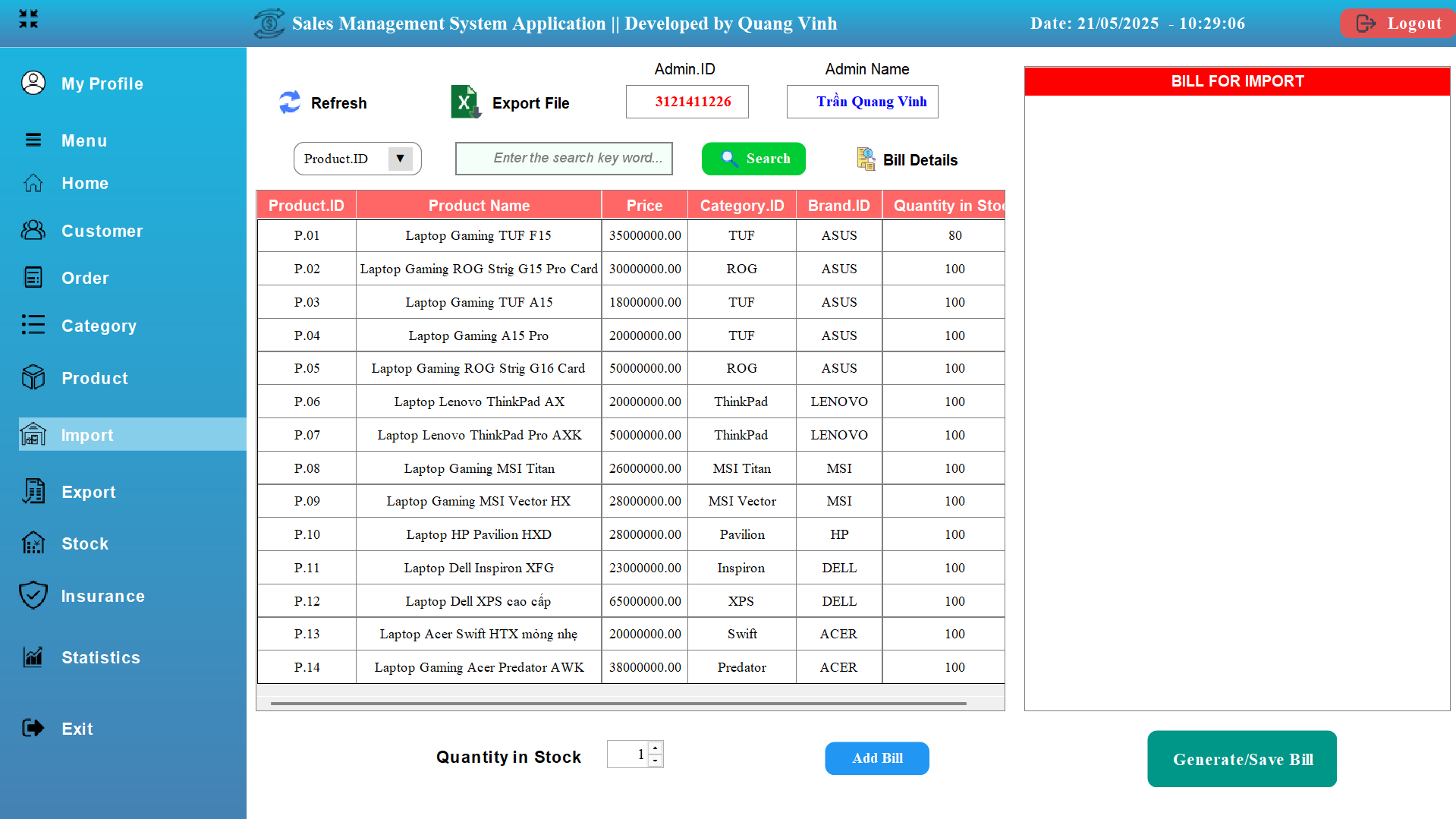


Image 26: Import Management interface supports Admin tracking

The Import Management interface supports Admin in tracking, processing and managing the import of products from the main warehouse to the store, ensuring that goods are always ready for sale. The system allows creating import vouchers, updating imported product information and exporting data for inventory.

**Features:**

* Create a goods receipt – Record information about products imported from the warehouse, including product code, product name, quantity, import price and supplier.
* Edit import slip – Update imported product information before confirming warehouse entry.
* Delete Receipts – Remove invalid or unnecessary receipts.
* Search for import vouchers – Look up import vouchers by voucher code, product name or supplier.
* Export import list to Excel file – Save import information for reporting and inventory purposes.

→ This interface helps Admin closely control the process of importing goods from the warehouse to the store, ensuring that goods are always updated accurately.

#### i) Export management function

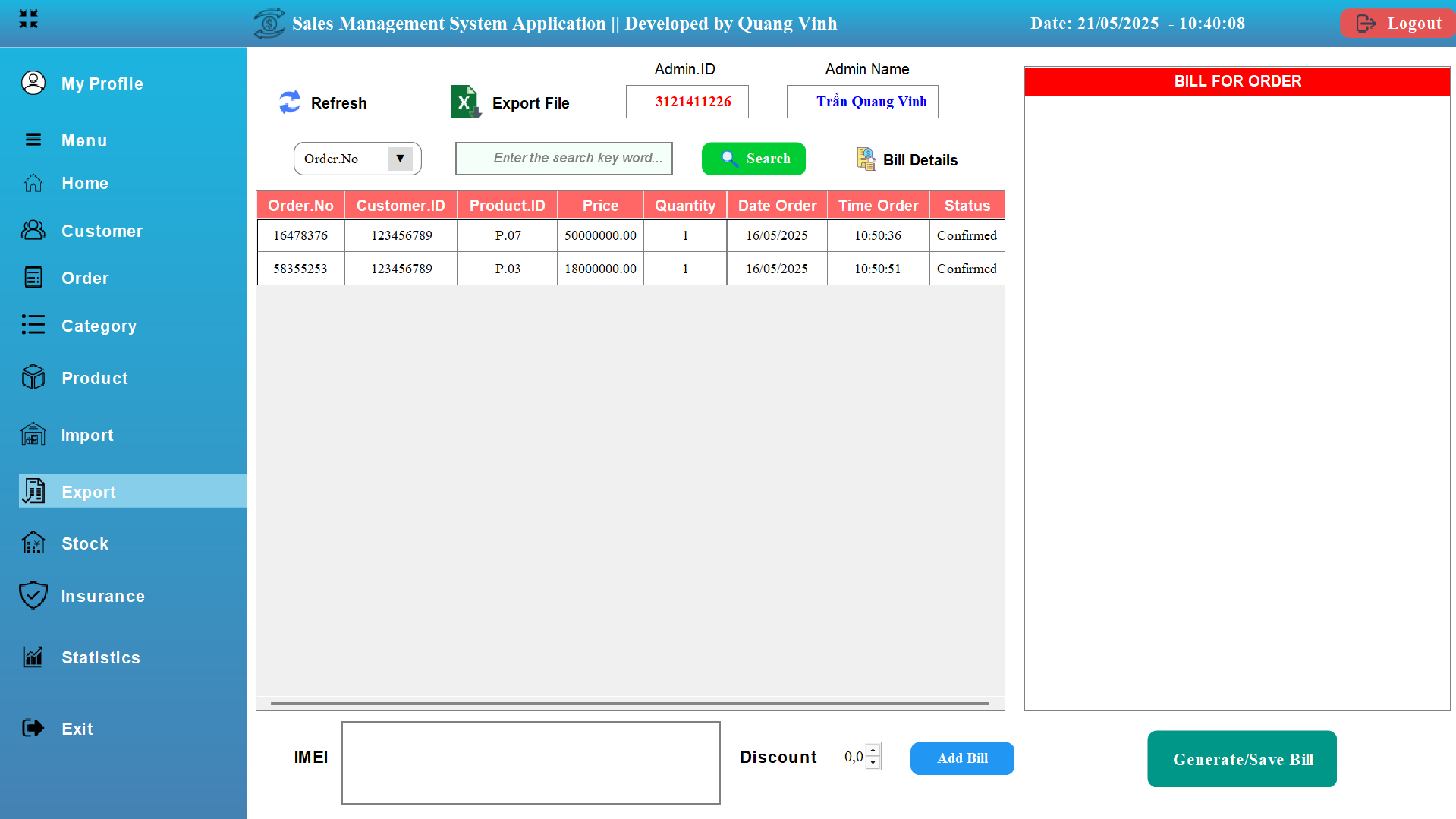


Image 27: Export Management interface supports Admin to check and process orders

The Export Management interface supports Admins in checking and processing orders based on customer purchasing requests. The system allows creating invoices, confirming products for sale and managing order data closely.

**Key Features:**

* Create shipping invoice – Record product information shipped based on order.
* Confirm shipment – Check the quantity of products in stock before making the shipment.
* Search for orders – Look up orders by Order No, Customer ID or Customer Name.
* Export invoices to Excel/PDF files – Save invoices for future reporting, checking and reconciliation.

→ The interface helps Admin manage goods export based on customer orders accurately and professionally.

#### k) Inventory data management function (total inventory)

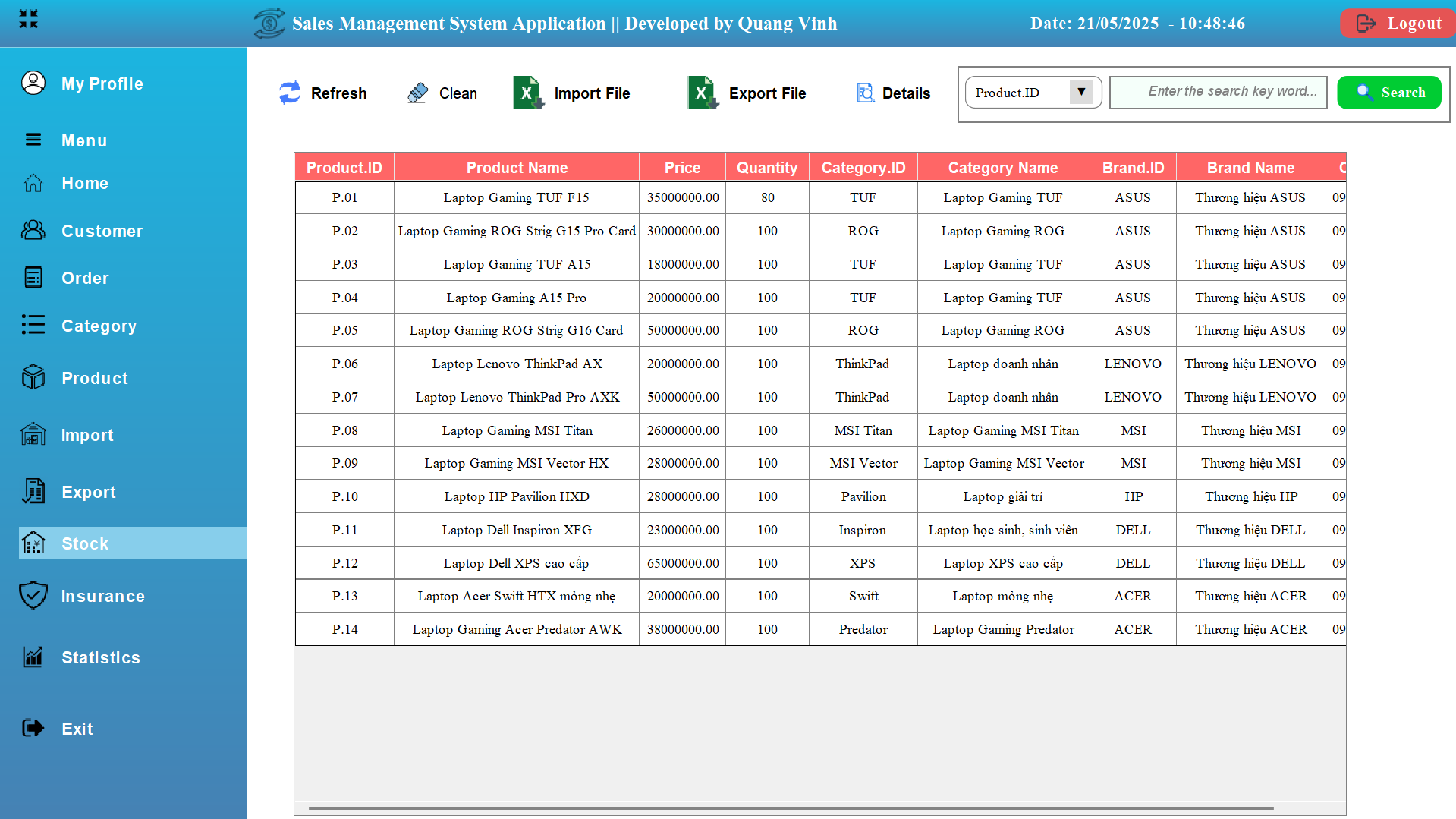


Image 28: Inventory Data Management interface helps Admin track

The Inventory Data Management interface helps Admin track and control the remaining quantity of products in the total warehouse, while supporting data import and export from Excel files, creating the premise for creating effective import receipts to the store.

**Key Features:**

* View inventory data – Displays a list of products in the total inventory, including product code, name, inventory quantity, and import price.
* Search for products in stock – Look up products by Product Code, Product Name or Supplier, for quick inventory.
* Import inventory data from Excel file – Supports importing product lists from Excel files, helping to update mass information quickly and accurately.
* Export inventory data to Excel file – Save inventory product list for reporting and product analysis.
* Check inventory quantity – Ensure the quantity of goods is always updated, avoiding shortage or surplus.

→ The interface helps Admin closely control total warehouse data, optimize the import process and ensure efficient system operation.

#### l) Warranty Management function

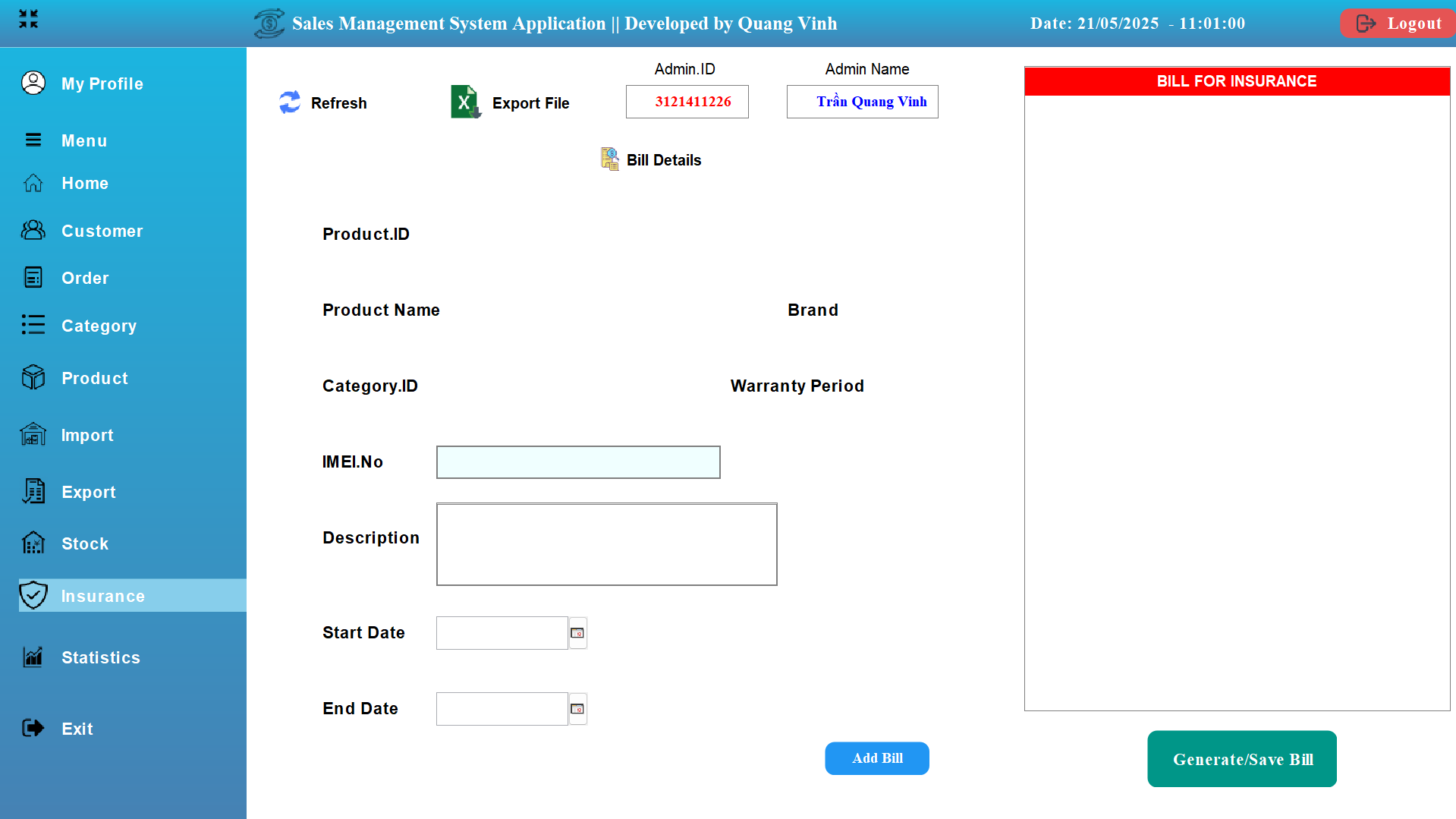


Image 29: Warranty Management Interface

The Warranty Management interface supports Admin in creating warranty cards for customers, ensuring that products are maintained according to the correct procedures. The system helps track warranty information, process warranty requests, and export data for effective management.

**Key Features:**

* Create warranty card – Record product information for warranty, including product code, customer name, warranty activation date and defect description.
* Warranty Confirmation – Check warranty conditions, approve claims and update status.
* Warranty Search – Look up warranty by Warranty Code, Customer Name, IMEI Code or Warranty Request Date.
* Export warranty data to Excel file – Save warranty list for reporting and service inventory.

→ The interface helps Admin closely control the warranty process, ensuring that products are repaired and processed according to requirements.

#### m) Statistical management function

**Bar chart**



Image 30: 'Bar chart' Statistics Management interface

The 'Bar chart' Statistics Management interface helps Admin analyze the popularity of product brands based on the number of products purchased by customers. The intuitive Bar Chart shows comparisons between brands, supporting the assessment of shopping trends and optimizing business strategies.

* Brand Popularity Statistics – Chart showing the number of products sold by brand.
* Shopping Trend Analysis – Compare which brands are most purchased by customers.
* Search and filter data – Admin can filter by time, product category or brand, giving a more detailed view.
* Export statistical data to PDF file – Save reports for analysis, supporting appropriate import strategies.
* Real-time chart updates – Automatically update with the latest data, ensuring data is always accurate.

→ The interface helps Admin track consumer trends, identify popular brands and optimize inventory.

**Line chart**

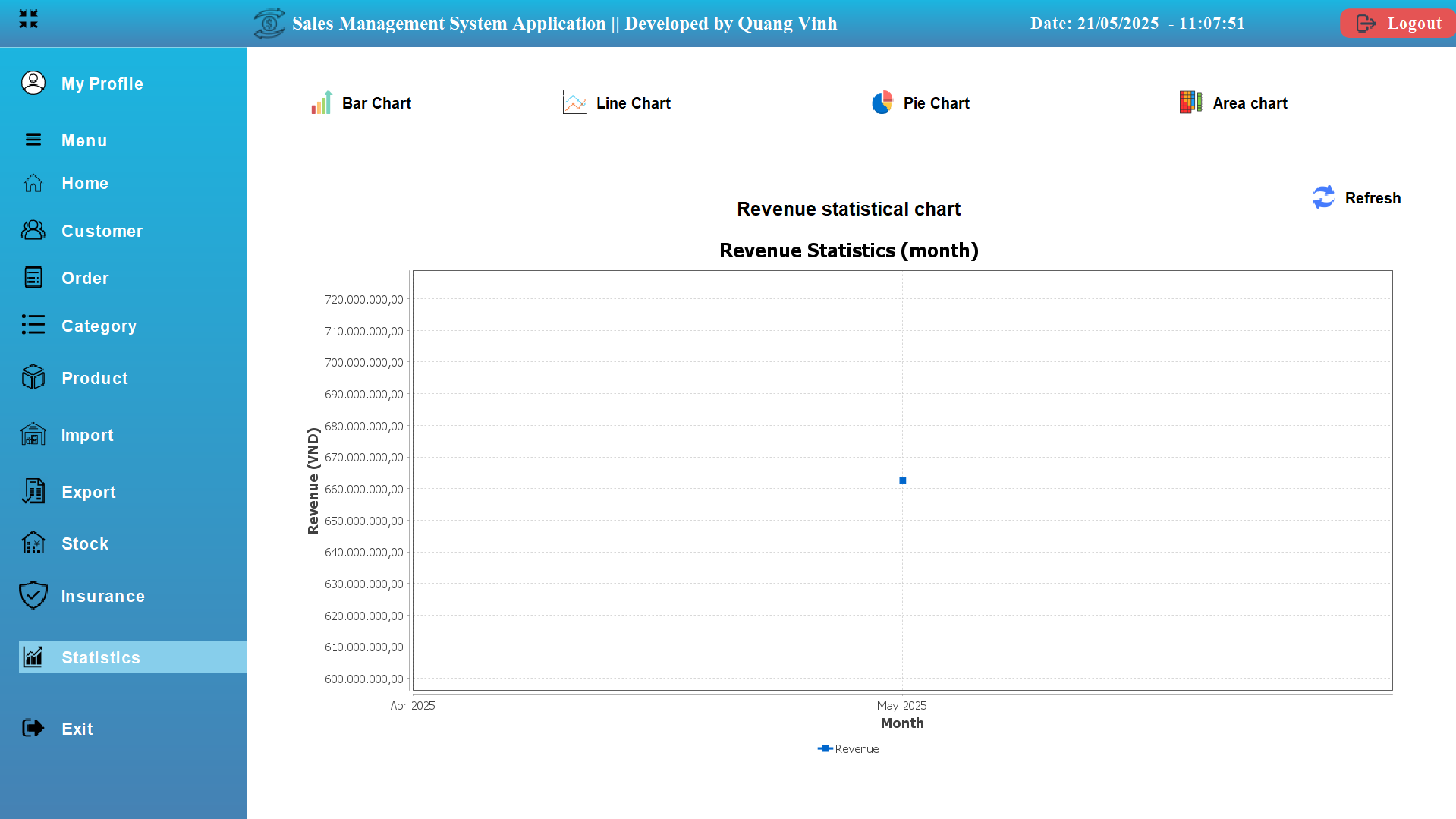


Image 31: 'Line chart' Statistics Management interface

The Revenue Statistics Management interface helps Admin track monthly and yearly revenue fluctuations through intuitive Line Charts. The system supports analyzing business trends, evaluating sales performance and optimizing development strategies.

**Key Features:**

* Revenue statistics over time – Charts display revenue by day, month, year, helping Admin evaluate business performance in each period.
* Sales trend analysis – Compare sales increases and decreases over time to develop appropriate sales strategies.
* Export statistical data to PDF file – Save revenue reports for analysis and business forecasting.
* Real-time chart updates – Automatically update the latest sales, ensuring information is always accurate and timely.

→ The interface helps Admin easily track revenue changes, analyze business strategies and optimize sales activities.

**Pie Chart**

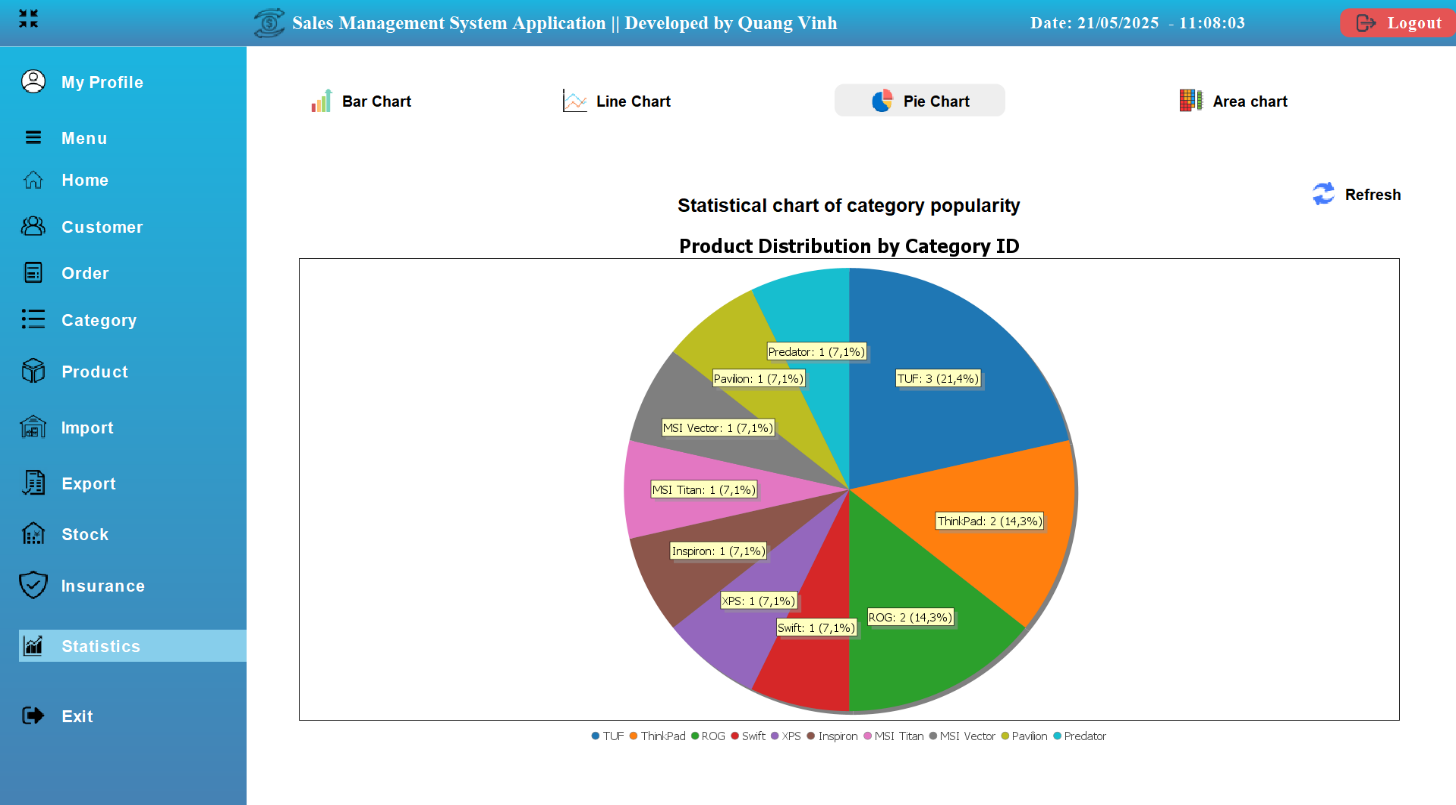


Image 32: 'Pie chart' Statistics Management interface

The Statistics Management interface helps Admin analyze the popularity of product categories based on customer purchases. The intuitive Pie Chart displays the ratio of laptop models by brand that are most purchased, supporting optimal import and business strategies.

**Key Features:**

* Product Category Popularity Statistics – Pie Chart shows the purchase rate by product category.
* Analyze consumer trends – Determine which product lines are purchased the most to optimize inventory.
* Search and filter data – Allows Admin to filter by time, product category or purchase quantity, providing a more detailed view.
* Export statistical data to PDF file – Save reports for analysis, supporting business strategy.
* Real-time chart updates – Automatically update with the latest data, ensuring accurate and timely information.

**Area chart**

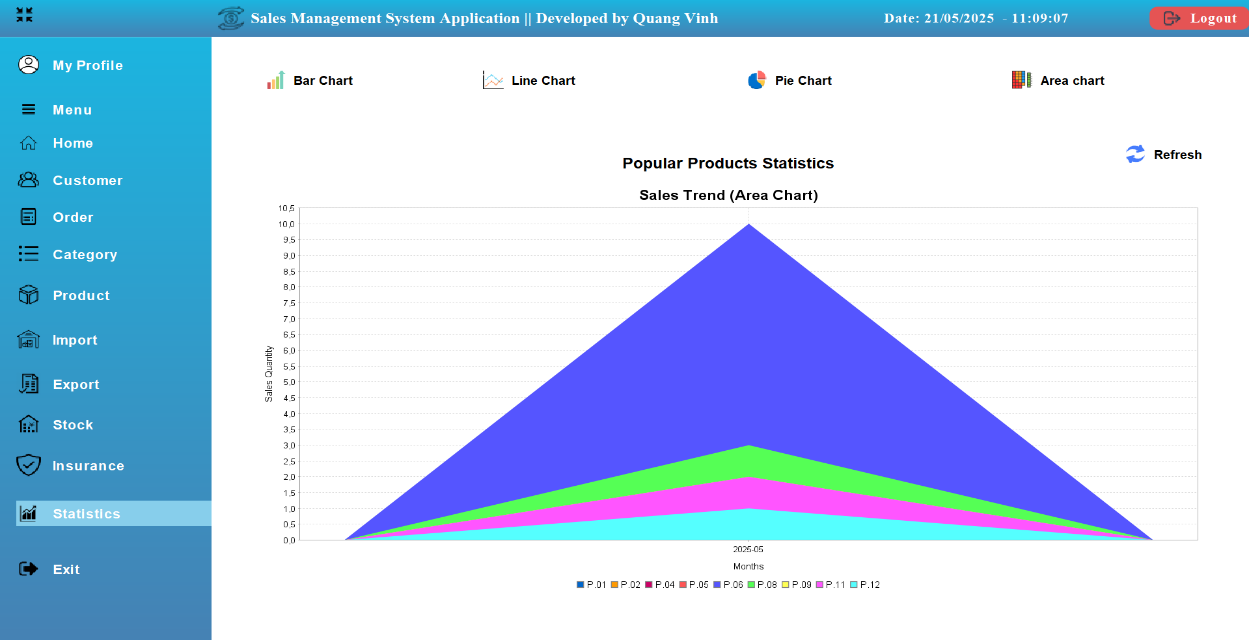


Image 33: 'Area chart' Statistics Management interface

The Statistics Management interface helps Admin track the popularity of each product based on sales volume. The intuitive Area Chart displays the percentage of most purchased products, supporting optimal import and business strategies.

**Key Features:**

* Popular Product Category Statistics – Shows the number of most purchased products by category.
* Consumer Trend Analysis – Helps identify which products are in high demand to adjust business strategies.
* Export statistical data to PDF file – Save reports on product popularity for analysis and import optimization.
* Real-time chart updates – The system automatically updates the latest data, ensuring accurate and timely data.

→ The interface helps Admin track the popularity of each product type, thereby adjusting import plans and business strategies.

# Chapter V: System Testing

## 5.1. Test Plan

### 5.1.1. Introduction

#### a) Objective

This test plan for the NPKStore Website project supports the following objectives:

Ensure all business processes (ordering, importing, product management, authorization) operate correctly as required.

Identify and fix potential errors before the website goes live.

Ensure website stability, security and performance.

Check website compatibility across different browsers and devices.

#### b) Background

NPKStore website is an e-commerce system specializing in trading computers and components, requiring comprehensive testing of functions to ensure user experience and system reliability.

### 5.1. 2 . Types of testing used

#### a) Functional Testing

**Testing Type: Black-box Testing**

* **Objective:** Ensure the system performs the functions requested by users, including ordering, logging in, adding products, processing orders...
* **Strategy:** Based on functional specification to design test cases. No need for tester to understand source code.
* **Example:** Test adding product to cart: enter quantity, press "Add to cart" button, check message and cart updates.

#### b) GUI Testing

**Test Type: Black Box**

**Goal:** Ensure the user interface is intuitive, properly designed, and functions properly when interacted with.

**Strategy:** Test colors, placement, fonts, navigation, button behavior, hover/click feedback...

**For example,** test the "Checkout" button to be right aligned, have a blue background color, and show a confirmation dialog when pressed.

### 5.1. 3. Testing tools

|  |  |
| --- | --- |
| **Tools** | **Purpose** |
| Apache Server | Server running the system |
| SQL server | Database Management |

### 5.1. 4. Test scope

The main functions in the system tested include:

User and Admin Registration/Login

Order, pay and track orders

Product and catalog management

Warehouse and invoice

Apply promotions, product warranty

Interact with the user interface and test it on popular browsers .

### 5.1. 5 . Remove from scope

Elements not included in the current test plan:

In-depth security testing with advanced vulnerability exploitation (penetration testing)

Real-world integrated payment system (using simulated environment)

### 5.1 .6 . Completion criteria

100% of test cases are executed.

Test case pass rate is at least 95%.

No more fatal errors or blocker errors.

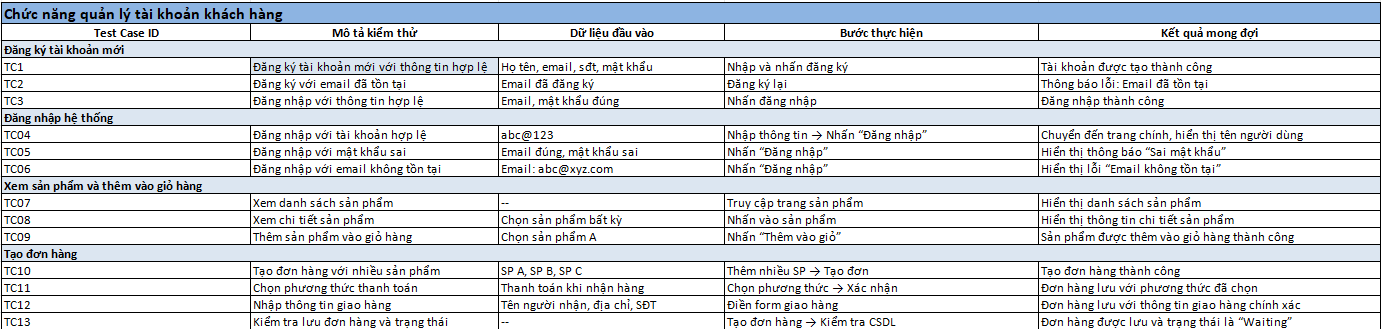
The interface works smoothly, with no display or navigation errors.

### 5.1 .7 . Risks and mitigation plans

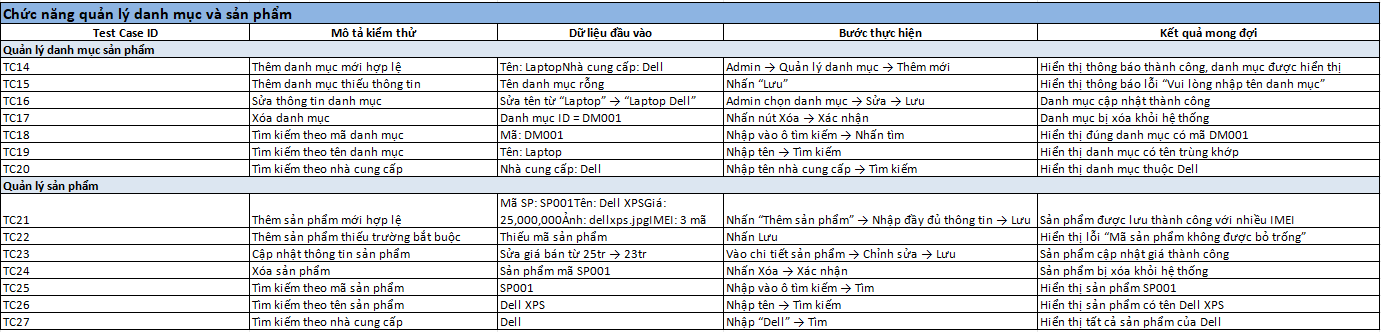
|  |  |
| --- | --- |
| **Risk** | **Mitigation measures** |
| Interface displayed incorrectly in browser | Cross-browser testing (Chrome, Firefox, Edge) |
| Not compatible on mobile devices | Responsive design and real-world testing |
| Late-onset functional errors | Regression testing on updates |

## 5. 2 . Test user interface functionality

### 5.1.1. Customer account management function



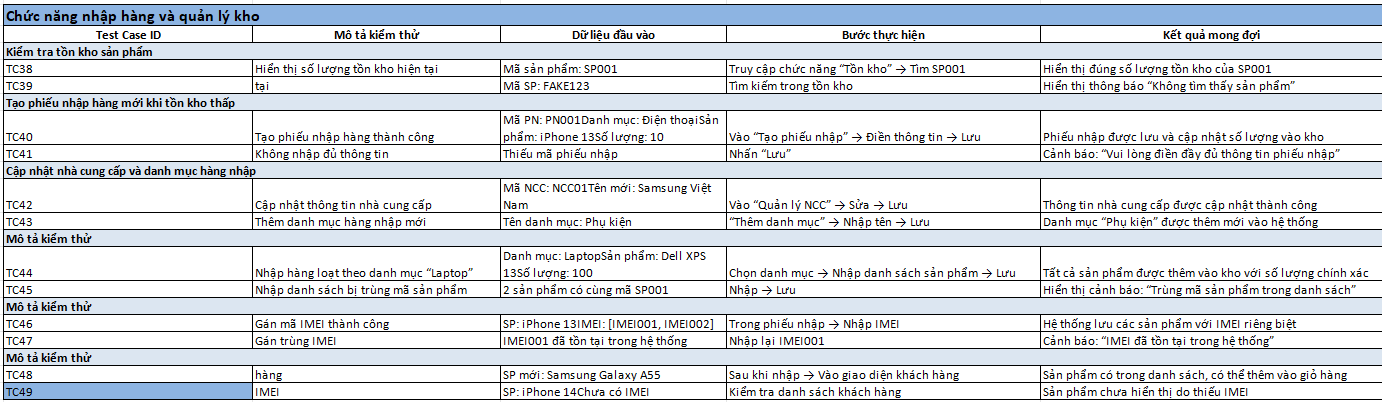
### 5.1.2. Category and product management functions



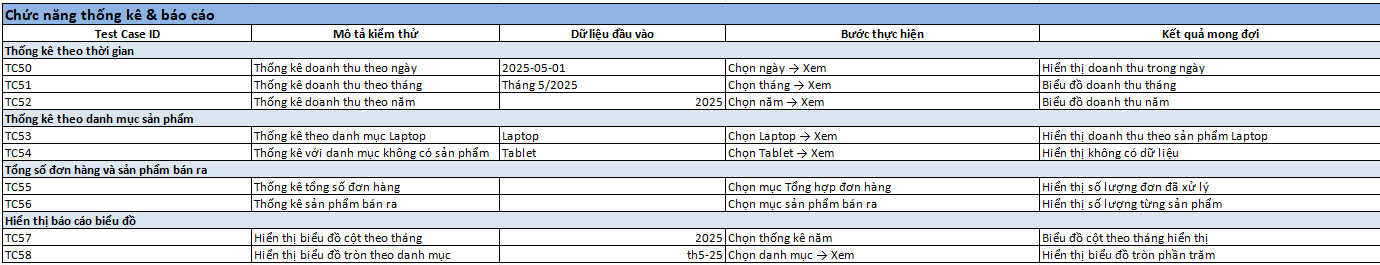
### 5.1.3. Order processing function (admin )



### 5.1.4. Import and warehouse management functions



### 5.1.5. Statistics & reporting functions



### 5.1.6. Customer Email sending function



## 5. 3 . Functional Test

### 5.2. 1. Admin

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-ADM-01 | Valid admin login | username: admin, password: 123 | Login successful |
| TC-ADM-02 | Wrong password | username: admin, password: wrong | Report incorrect information |
| TC-ADM-03 | Account does not exist | username: xyz, password: 123 | Error message does not exist |

### 5.2. 2. Customer

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-CUS-01 | Add valid customers | name: An, phone: 0909999999 | More success |
| TC-CUS-02 | Duplicate phone number | phone: 0909999999 | Report duplicate number error |
| TC-CUS-03 | Missing customer name | phone: 0908888888 | Report missing name error |
| TC-CUS-04 | Phone number is in wrong format | phone: abcxyz | Format error report |
| TC-CUS-05 | Customer Update | ID: 1, new phone: 0999999999 | Update successful |

### 5.2. 3. Product

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-PRD-01 | Add new product | name: iPhone 15, category: 1 | More success |
| TC-PRD-02 | Category does not exist | category: 99 | FK category error report |
| TC-PRD-03 | Missing product name | category: 1 | Report missing name error |
| TC-PRD-04 | Duplicate product code | product\_id already exists | Report duplicate key error |
| TC-PRD-05 | Update information | Price change | Update successful |

### 5.2. 4. Orders

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-ORD-01 | Create new order | customer\_id: 1, staff\_id: 2 | Order created successfully |
| TC-ORD-02 | Missing staff information | customer\_id: 1 | Report missing staff error |
| TC-ORD-03 | Customer does not exist | customer\_id: 999 | Customer FK error report |

### 5.2. 5. Order\_Details

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-ODT-01 | Add order details | order\_id: 1, product\_id: 3, qty: 2 | More success |
| TC-ODT-02 | Product does not exist | product\_id: 999 | Report product FK error |
| TC-ODT-03 | Negative number | qty: -1 | Data error report |

### 5.2. 6. Bill\_Exported

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-BEX-01 | New invoice | order\_id: 1, date: 2024-01-01 | Export successful |
| TC-BEX-02 | Order does not exist | order\_id: 999 | FK order error report |

### 5.2. 7. Bill\_Exported\_Details

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-BEXD-01 | Add invoice details | bill\_id: 1, product\_id: 3, qty: 1 | Add OK |
| TC-BEXD-02 | Lack of quantity | qty: null | Report missing data error |

### 5.2. 8. Bill\_Imported

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-BIM-01 | Import goods from suppliers | supplier\_id: 2, date: 2024-01-02 | Import successful |
| TC-BIM-02 | Supplier does not exist | supplier\_id: 999 | FK supplier error report |

### 5.2. 9. Bill\_Imported\_Details

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-BIMD-01 | Add import invoice details | bill\_id: 1, product\_id: 3, qty: 5 | Add OK |
| TC-BIMD-02 | Negative number | qty: -3 | Data error report |

### 5.2.10. Supplier

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-SUP-01 | Add valid supplier | name: ABC Co., phone: 0908... | Add OK |
| TC-SUP-02 | Missing supplier name | phone: 0908... | Report missing name error |

### 5.2. 11. IMei\_Product

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-IME-01 | Add product IMEI | imei: 111..., product\_id: 3 | Add OK |
| TC-IME-02 | Duplicate IMEI | imei: 111... already exists | Report duplicate code |

### 5.2. 12. IMei\_Product\_Stock

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-IMES-01 | Update IMEI status | imei: 111..., status: sold | Update OK |
| TC-IMES-02 | IMEI does not exist | imei: 999... | Error message not available |

### 5.2. 13. Insurance

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-INS-01 | Create product warranty card | imei: 111..., start: 01/01/24 | Create OK |
| TC-INS-02 | IMEI not sold | imei: 123..., status: stock | Invalid error message |

### 5.2. 14. Discount

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-DSC-01 | Add discount code | code: SALE50, percent: 50 | Add OK |
| TC-DSC-02 | Percentage error | percent: 150 | Data error report |

### 5.2. 15. Stock

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-STK-01 | Update warehouse | product\_id: 3, qty: +10 | Update OK |
| TC-STK-02 | Wrong product | product\_id: 999 | Error message not available |

### 5.2. 16. Category

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-CAT-01 | Add new category | name: Laptop | Add OK |
| TC-CAT-02 | Duplicate category name | name: Laptop | Report duplicate error |

### 5.2. 17. Staff

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Describe** | **Input data** | **Expected results** |
| TC-STF-01 | Add new employee | [name: B, email: b@mail.com](mailto:b@mail.com) | Add OK |
| TC-STF-02 | Invalid email | name: C, email: notanemail | Format error report |
| TC-STF-03 | Duplicate email | [name: D, email: b@mail.com](mailto:b@mail.com) | Report duplicate email error |

# Chapter VI: Conclusion

In the context of strong digital transformation, businesses are increasingly focusing on applying information technology to improve business performance. In particular, the sales sector - characterized by the continuous circulation of goods, customer information, orders and warehouse data - requires a tight, accurate and flexible management system. On that basis, the topic "Building a sales management system" has been deployed with the goal of designing a standardized database, building a set of test cases to test business functions, thereby ensuring the system operates accurately, effectively supporting departments in the business.

## 6.1 . Results achieved

After the process of research, analysis and system construction, the research team achieved the following specific results:

The database design includes 17 tables, covering all core operations of the sales process: from importing goods, storing, managing products, tracking customers, invoicing, to tracking warranties and applying discount policies. The database structure includes main tables such as: Admin, Staff, Customer, Product, Orders, Order\_Details, Bill\_Exported, Bill\_Exported\_Details, Bill\_Imported, Bill\_Imported\_Details, Category, Discount, Insurance, IMei\_Product, IMei\_Product\_Stock, Stock, and Supplier. The tables are linked together through primary keys - foreign keys, ensuring data integrity and consistency.

Build a detailed **test case set in Excel format,** clearly describing:

* Table name and operation to be tested.
* Describe the test case (real-life situation).
* Input conditions.
* Expected Output.
* Status (Passed/Failed).

These test cases accurately reflect real-world behaviors in the sales system, including both valid functional tests ( **positive test cases** ) and abnormal or erroneous data cases ( **negative test cases** ).

Testing is done manually on **Excel files,** focusing on common operations such as:

* Add new customers and check for duplicate phone numbers.
* Check duplicate order code.
* Inventory management when exporting/importing goods.
* Link orders with detailed warranties and invoices.
* Check conditions when adding products that are not in stock,…

**closed-loop** sales management support , while providing a solid foundation for expanding or integrating smart features in the future (such as data analysis, sales consulting chatbots, warehouse management automation, etc.).

**Academically,** the process of implementing the project helped the group improve their skills:

* Practical database design.
* Analyze relationships between data tables.
* Build a software testing process.
* Combine functional testing with business thinking.

## 6.2 . System limitations

Besides the achieved results, the system still has certain limitations:

* **No automated testing yet:** Current test cases are still performed manually, which is time-consuming and difficult to scale to test multiple cases at the same time.
* **Lack of non-functional testing:** Not yet evaluating data processing performance, security or load capacity when many users access at the same time.
* **The system has not been integrated into a real deployment environment (CI/CD, Docker, Cloud),** so it is difficult to assess the level of compatibility with production environments.

## 6.3 . Future development direction

To further improve the sales management system and move towards practical application, the group plans to develop the following additional content:

* **Build user interfaces (UI)** using web technologies (React, HTML/CSS, or Streamlit for prototypes) that make it easy for users to: add orders, create invoices, check inventory, track warranties,...
* **Apply automated testing** using tools like Selenium, Pytest, or integrate into CI/CD testing pipeline.
* **Develop a reporting system for statistics on revenue, inventory, return rates, etc.** using Power BI or visualization libraries.
* **Integrate internal AI models or chatbots** , using LLM to support users in querying information (e.g. “Monthly sales statistics”, “Product X inventory status”).
* **Bring the system to a production deployment platform,** which can be Docker, a real SQL Server, or deployed on the cloud like Azure or AWS.

## 6.4 . General conclusion

In conclusion, the topic “Sales Management” is not only a technological problem but also a practical application model, fully reflecting the common business processes in enterprises. The tight database design combined with a detailed test case system helps ensure the correctness and efficiency of the system. Through this topic, the group not only consolidates professional knowledge in database design and software testing, but also learns how to approach practical problems from the perspective of users and system administrators. This will be an important premise for developing more professional management systems in the future.

# References

[1] Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education Limited. Retrieved from <https://www.amazon.com/Software-Engineering-10th-Ian-Sommerville/dp/0133943038>

[2] Sommerville, I. (2019). Engineering Software Products: An Introduction to Modern Software Engineering. Pearson Education Limited. Retrieved from <https://www.pearson.com/en-us/subject-catalog/p/engineering-software-products-an-introduction-to-modern-software-engineering/P200000003578/9780135210642>

[3] Bruegge, B., & Dutoit, A.H. (2013). Object-Oriented Software Engineering Using UML, Patterns, and Java (3rd ed.). Pearson Education. Retrieved from <https://www.pearson.com/en-us/subject-catalog/p/object-oriented-software-engineering-using-uml-patterns-and-java/P200000003574/9780136061250>

[4 ] Cervantes, H., & Kazman, R. (2016). Designing Software Architectures: A Practical Approach. Pearson Education. Retrieved from <https://www.pearson.com/en-us/subject-catalog/p/designing-software-architectures-a-practical-approach/P200000003573/9780134390789>