Ảnh có chứa văn bản

Mô tả được tạo tự động

**Fixed Asset Management System**

Version 1.1

Lecturer:

Le Thanh Trong

Students:

Pham Hieu Vy – 19520358

Le Thanh Tuan – 19522467

Phung Thanh Tu – 19522455

TPHCM, January - 2022

PREFACE

**Application development environment (expected): Desktop app as client using WPF .NET and Webserver as server using Nodejs and Database using Microsoft SQL Server hosted by smarterasp.net**

**Group’s information:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| .No | Stuent ID | Full name | Phone | Email |
| 1 | 19522455 | Phung Thanh Tu | 0352446482 | 19522455@gm.uit.edu.vn |
| 2 | 19520358 | Pham Hieu Vy | 0838568041 | 19520358@gm.uit.edu.vn |
| 3 | 19522467 | Le Thanh Tuan | 0382362384 | 19522467@gm.uit.edu.vn |

Table of contents

[PREFACE 2](#_Toc92578742)

[Statement of the Project 5](#_Toc92578743)

[1 Purpose of the system: 5](#_Toc92578744)

[2 Description of users/forms and related entities: 6](#_Toc92578745)

[3 Description of processes (business operations): 6](#_Toc92578746)

[4 Requirements (functional): 6](#_Toc92578747)

[5 Requirements (non-functional) and constraints (if any): 7](#_Toc92578748)

[Use Case model 7](#_Toc92578749)

[1 Actor List 7](#_Toc92578750)

[2 Use-case List 8](#_Toc92578751)

[3 Use-case Specification 8](#_Toc92578752)

[Overview 13](#_Toc92578753)

[1 Overview of the system 13](#_Toc92578754)

[2 Database 14](#_Toc92578755)

[2.1 Database overview 14](#_Toc92578756)

[2.2 Database environment 14](#_Toc92578757)

[2.3 Database design 14](#_Toc92578758)

[3 Webserver 15](#_Toc92578759)

[3.1 Webserver overview 15](#_Toc92578760)

[3.2 Webserver environment 15](#_Toc92578761)

[3.3 Webserver design 16](#_Toc92578762)

[3.4 File and folder structure 18](#_Toc92578763)

[3.5 API structure 20](#_Toc92578764)

[Conceptual model 22](#_Toc92578765)

[1 Class diagram 23](#_Toc92578766)

[1.1 Data 23](#_Toc92578767)

[1.2 WPF Client 24](#_Toc92578768)

[**1.3** List of classes and relationships in Database Module 24](#_Toc92578769)

[**1.4** List of classes and relationships in UI Module 25](#_Toc92578770)

[2 Cohesion and coupling 26](#_Toc92578771)

[3 Sequence diagram*.* 27](#_Toc92578772)

[3.1 Login 27](#_Toc92578773)

[3.2 Device List 27](#_Toc92578774)

[3.3 Device Detail 28](#_Toc92578775)

[3.4 Check Device 28](#_Toc92578776)

[3.5 Inventory 29](#_Toc92578777)

[3.6 Liquidation 29](#_Toc92578778)

[3.7 Repair 30](#_Toc92578779)

[3.8 Statistics 30](#_Toc92578780)

[Screen Flow Diagram 31](#_Toc92578781)

[Screen Detail 31](#_Toc92578782)

[1 Login Window 31](#_Toc92578783)

[1.1 Main Flow 32](#_Toc92578784)

[**1.2** UI 32](#_Toc92578785)

[1.3 Data Mapping / Validation Rule 32](#_Toc92578786)

[2 Main Window 31](#_Toc92578787)

[2.1 Main Flow 32](#_Toc92578788)

[**2.2** UI 32](#_Toc92578789)

[2.3 Data Mapping / Validation Rule 32](#_Toc92578790)

[3 Inventory Page 31](#_Toc92578791)

[3.1 Main Flow 32](#_Toc92578792)

[**3.2** UI 32](#_Toc92578793)

[3.3 Data Mapping / Validation Rule 32](#_Toc92578794)

[4 Liquidation Page 31](#_Toc92578795)

[4.1 Main Flow 32](#_Toc92578796)

[**4.2** UI 32](#_Toc92578797)

[4.3 Data Mapping / Validation Rule 32](#_Toc92578798)

[5 Repair Page 32](#_Toc92578799)

[5.1 Main Flow 33](#_Toc92578800)

[**5.2** UI 33](#_Toc92578801)

[5.3 Data Mapping / Validation Rule 33](#_Toc92578802)

[6 Statistic Page 32](#_Toc92578803)

[6.1 Main Flow 33](#_Toc92578804)

[**6.2** UI 33](#_Toc92578805)

[6.3 Data Mapping / Validation Rule 33](#_Toc92578806)

[7 Add Personnel Window 32](#_Toc92578807)

[7.1 Main Flow 33](#_Toc92578808)

[**7.2** UI 33](#_Toc92578809)

[7.3 Data Mapping / Validation Rule 33](#_Toc92578810)

[8 New Personnel Window 32](#_Toc92578811)

[8.1 Main Flow 33](#_Toc92578812)

[**8.2** UI 33](#_Toc92578813)

[8.3 Data Mapping / Validation Rule 33](#_Toc92578814)

[9 Edit Status Window 32](#_Toc92578815)

[9.1 Main Flow 33](#_Toc92578816)

[**9.2** UI 33](#_Toc92578817)

[9.3 Data Mapping / Validation Rule 33](#_Toc92578818)

[10 Inventory Detail Window 32](#_Toc92578819)

[10.1 Main Flow 33](#_Toc92578820)

[**10.2** UI 33](#_Toc92578821)

[10.3 Data Mapping / Validation Rule 33](#_Toc92578822)

[11 Establish Inventory Council Window 32](#_Toc92578823)

[11.1 Main Flow 33](#_Toc92578824)

[**11.2** UI 33](#_Toc92578825)

[11.3 Data Mapping / Validation Rule 33](#_Toc92578826)

[12 Inventory Device Window 32](#_Toc92578827)

[12.1 Main Flow 33](#_Toc92578828)

[**12.2** UI 33](#_Toc92578829)

[12.3 Data Mapping / Validation Rule 33](#_Toc92578830)

[12.4 Data Mapping / Validation Rule 33](#_Toc92578831)

[Installation 32](#_Toc92578832)

[1 System Requirement 32](#_Toc92578833)

[2 How to Install 32](#_Toc92578834)

[Demo 32](#_Toc92578835)

[Future Update 32](#_Toc92578836)

[Team member assignment 33](#_Toc92578837)

[Lesson Learned 33](#_Toc92578838)

Statement of the Project

**Document change record sheet**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Authors** |
| 03/10/2021 | 1.0 | First product release | Group 15 |
| 03/12/2021 | 1.1 | Update software | Group 15 |
| 17/12/2021 | 2.0 | Create Nodejs webserver | Group 15 |

*Describe topic:*

The software is used to manage fixed assets in an enterprise, thereby easily checking and handling when there is a problem and convenient for inventory, routine maintenance for equipment and items needed.

# Purpose of the system:

To re-evaluate the quality of each type of equipment so as to have an appropriate equipment distribution plan; the information needed through an asset inventory is: how much is each piece of equipment still worth? Date of the census, the individuals who participated in the inventory.

# Description of users/forms and related entities:

* Kindergartens, primary, high schools and universities in the country.
* Kindergarten, primary and high schools abroad.

# Description of processes (business operations):

|  |  |
| --- | --- |
| **Milestones** | **Date** |
| Build project idea | *25/09/2021* |
| Analyze national and international schools | *01/10/2021* |
| Complete the development of the project's approach and charter | *18/10/2021* |
| Complete database design | *03/11/2021* |
| Complete interface design | *05/11/2021* |
| Complete database construction | *10/11/2021* |
| Complete interface construction, connect database | *12/12/2021* |
| Software testing | *3/1/2022* |
| Software release | *8/1/2022* |

# Requirements (functional):

System must have these functions below:

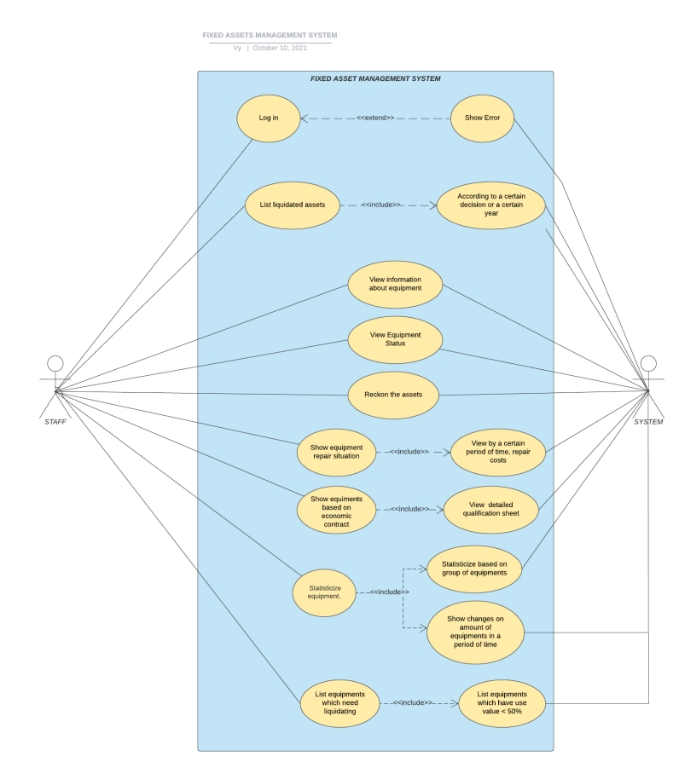
* List of devices is managed by each unit.
* Equipment repair situation of all units in a certain period of time, repair costs.
* Asset inventory of each unit (qualification sheet for each device).
* List of equipment to be liquidated according to a certain decision or to a certain year.
* Information about equipment in a certain economic contract.
* Status of devices.
* Statistics of the number of assets by group of the whole school, the increase or decrease in the number of devices over a certain period of time.
* List assets subject to liquidation (remaining use value is less than 50%).

# Requirements (non-functional) and constraints (if any):

* Software must be released on October 3, 2021 by the customers’ requirements and sent to them for checking.
* Edit, supplement and complete customer requirements after submitting the demo. Release the official version on February 3, 2022.
* All security services must be carefully guaranteed.
* The system needs to be deployed in the corporate intranet.
* User Interface must meet expectation about UI Design (5 principle)

Use Case model

Use-case diagram



# Actor List

|  |  |  |
| --- | --- | --- |
| .No | Name of Actor | Short description/Notes |
| 1 | Staff | Staff will have permission to all functions that are in the system. |
| 2 | System | System will response all the requests from Staff. |

# Use-case List

|  |  |  |
| --- | --- | --- |
| .No | Name of Use-case | Short desctiption/Notes |
| 1 | Login | User enter username and password, then login to the system |
| 2 | List liquidated assets | Show list of liquidated assets based on year or liquidating decision |
| 3 | **View information about assets** | Show list of assets and their detailed information |
| 4 | **Show equipment repair situation** | Show equipment repair situation |
| 5 | **Reckon assets** | Show assets inventory |
| 6 | **Show equipment based on economic contract** | The use-case begins when staff want to see information about equipment in a certain economic contract. |
| 7 | **View equipment status** | View use value, detailed status,… of assets |
| 8 | **Statisticize equipment** | Make statistic about assets |
| 9 | **List equipments which need liquidating** | List some assets which has use value is less than 50% and thus need liquidating |

# Use-case Specification

***Note: For each Use-case, specify the content of the Use-case according to the following template:***

|  |  |
| --- | --- |
| Name: | **Login** |
| Description: | User enter username and password, then login to the system. |
| Special Requirement: |  |
| Preconditions: | System connected to the Internet  Database is available to verify user’s account |
| Postconditions: |  |
| Normal Flow: | User enter username and password in the corresponding field.  System verify user’s login information.  System grant access to valid user. |
| Alternative Flows: | User enter wrong password or username : Show error and require user to enter again. |
| Extend Points: | When user failed to insert valid information. |

|  |  |
| --- | --- |
| Name: | **View information about assets** |
| Description: | Show list of assets and their detailed information |
| Special Requirement: | User has logged into system. |
| Preconditions: | Database connected. |
| Postconditions: | User can click to menu to use another features. |
| Normal Flow: | System show list of assets which is within management of user’s faculty. |
| Alternative Flows: | If user is in another feature, user have to click on Show List on standard menu so that system can show list of assets. |
| Extend Points: | none |

|  |  |
| --- | --- |
| Name: | **Show equipment repair situation** |
| Description: | Show equipment repair situation |
| Special Requirement: | User has logged into system. |
| Preconditions: | Database connected. |
| Postconditions: | User can click to menu to use another features. |
| Normal Flow: | User click on “Show repair situation” on menu  System show a list of units with their repair situation and fixing bills. |
| Alternative Flows: | none. |
| Extend Points: | none |

|  |  |
| --- | --- |
| Name: | **Reckon assets** |
| Description: | Show assets inventory |
| Special Requirement: | User has logged into system. |
| Preconditions: | Database connected. |
| Postconditions: | User can click to menu to use another features. |
| Normal Flow: | User click on “Show assets reckoning information”  User choose units of which they want to reckon assets.  System show assets inventory of chosen units. |
| Alternative Flows: | none |
| Extend Points: | none |

|  |  |
| --- | --- |
| Name: | **List liquidated assets** |
| Description: | Show list of liquidated assets based on year or liquidating decision |
| Special Requirement: | User has logged into system. |
| Preconditions: | Database connected. |
| Postconditions: | User can click to menu to use another features. |
| Normal Flow: | User click on “Show liquid assets based on” in menu  User choose year/decision  System show information of liquidated assets |
| Alternative Flows: | none |
| Extend Points: | none |

|  |  |
| --- | --- |
| Name: | **Show equipment based on economic contract** |
| Description: | The use-case begins when staff want to see information about equipment in a certain economic contract. |
| Special Requirement: | none |
| Preconditions: | Staff must successfully login to the system.  The economic contract needs to be found in the list of contracts available. |
| Postconditions: | .If the contract needs to be displayed on the list, it will shows the information of the disadvantages in that economic contract |
| Normal Flow: | Search for economic contracts: staff will find a contract to see in the list of existing contracts  Display information about equipment in the economic contract |
| Alternative Flows: | Staff can continue to choose other economic contracts: continue showing information about equipment. |
| Extend Points: |  |

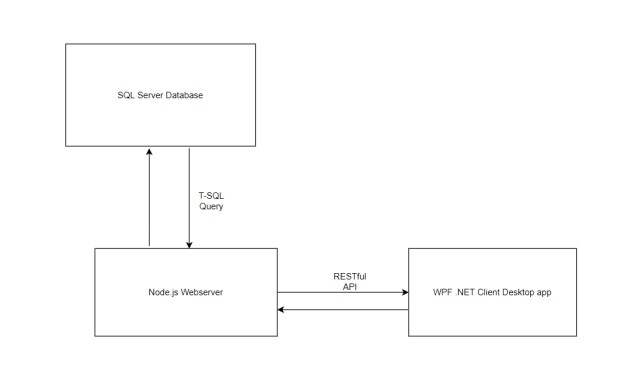
|  |  |
| --- | --- |
| Name: | **View equipment status** |
| Description: | The Use-case begins when staff want to see information about usage status of equipment (New equipment or used, the device has been liquidated or not). |
| Special Requirement: | none |
| Preconditions: | Staff must successfully login to the system  Need to have full information about the usage status of all equipment. |
| Postconditions: | .After the selected equipment will display information about their usage status |
| Normal Flow: | Displays a list of available devices: Staff will choose the equipment information section to display them.  Displays equipment status: select the equipment to display their information |
| Alternative Flows: | Staff can continue to choose other equipment: continue showing information about usage status of equipment. |
| Extend Points: | none |

|  |  |
| --- | --- |
| Name: | **Statisticize equipment** |
| Description: | The Use-case begins when staff want to list the number of assets in the whole school, the increase in the number of devices at a certain period of time |
| Special Requirement: |  |
| Preconditions: | Staff must successfully login to the system  The selected group and time need to be available in the system |
| Postconditions: | If Use-case is successfully executed, system will be displayed as a chart to display information details |
| Normal Flow: | Identify the selected group or time to display information  Display information about group assets in groups  Displays information about the increase in the number of equipment according to the specified period of time |
| Alternative Flows: | Staff can continue to choose group or time to display information |
| Extend Points: | none |

|  |  |
| --- | --- |
| Name: | **List equipments which need liquidating** |
| Description: | The Use-case begins when staff want to list properties that need to be liquidated (the remaining use value is less than 50%). |
| Special Requirement: | none |
| Preconditions: | Staff must successfully login to the system  The system must determine the remaining use value of the equipment |
| Postconditions: | If Use-case is successfully executed, The system be displayed the full information of the device on use time |
| Normal Flow: | Select the list of assets to be liquidated: staff will choose the list of assets to display information.  The system identifies the equipment with the remaining use value of more than 50% and displays them |
| Alternative Flows: | none |
| Extend Points: | none |

Overview

# Overview of the system

This system consist of 3 parts:

* The SQL Server Database
* The Nodejs Webserver
* The WPF.NET Desktop App

They interact with each other by different protocol:

* The client cannot interact directly with the database
* The client interacts with the webserver by RESTful API with JSON datatype of request/response body.
* The webserver interacts with the database by using Transact-SQL queries.

Please note that this document mainly focuses on the WPF .NET Client because that the database and the webserver are not developed using object oriented programming method so we only specify about database and webserver in just 2 small part of this document.

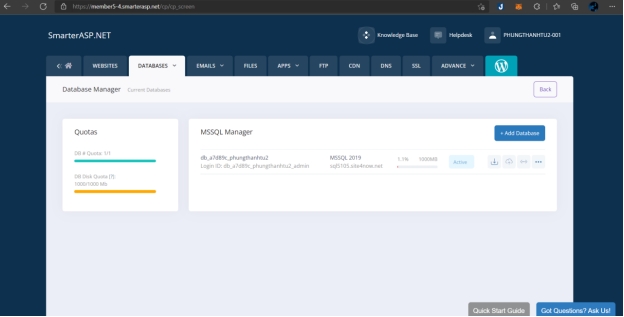
# Database

## Database overview

The database of the system is a relational database. It is to store data for the system. Webserver get data from database and send it to the client when required, also update data in the database when client send update information.

## Database environment

The database is hosted online, using service of smarterasp.net, with 60day existence. After 60 day, the database will be terminated.



This is a MS SQL Server database, to manage the database directly, we use SQL Server Management Studio.

## Database design

This is the design prototype of the database, described by an ERD Diagram



# Webserver

## Webserver overview

The webserver is the middle part of the system. It’s a bridge to make an interaction between the client and the database. It helps the client to get and send data to the SQL database, using JSON and SQL Query.

It communicates with the client by RESTful web API. Desktop app can using HTTP Request to send and request data from the webserver and then the webserver manipulate request and data in request, then using model to contact with the database to get corresponding data to send back to the client.

The webserver consists of 3 part:

* Model contain module which connect and contact directly with the database through SQL queries
* The Controller handle and manipulate the data and it’s also responsible for sending data to the client.
* The routes define route URIs to specify URI for the client to use.

There’s also an entry module (index.js) which is responsible for setting up the main routes, also the façade of the webserver (all data request from client will be received by this module).

## Webserver environment

* Programming language: JavaScript
* Programming framework: Node.js
* Cloud webserver host: Heroku Cloud Platform
* Host domain: https://asset-management-webserver.herokuapp.com/

## Webserver design

This is the webserver package diagram:

Ảnh có chứa văn bản, máy tính

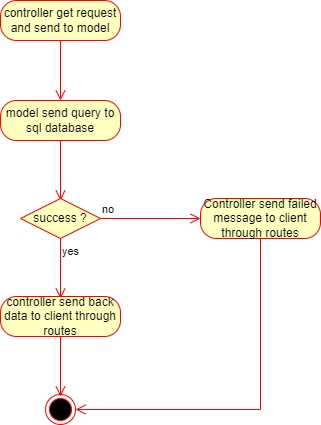
Mô tả được tạo tự động

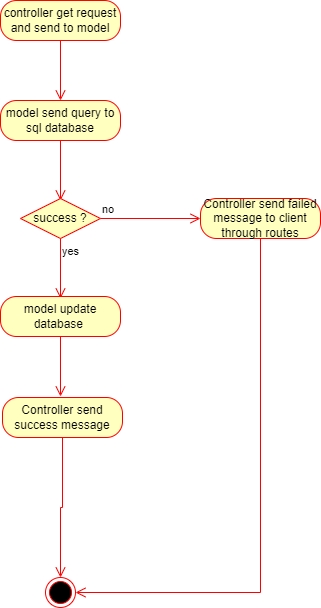
Dependency in this package diagram is simplified. In fact every package routes module uses dependency of its same name controller package, and every controller package uses dependency of its same name model package.

This diagram specify how data is transferred and manipulated in the webserver.

Note that this is the universal for all function of the webserver. We don’t focus on backend in this document so we won’t specify all function, moreover all function uses the same data flow structures.

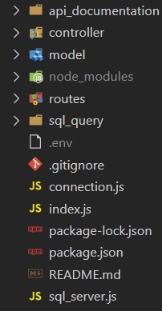
Data flow for user’s GET request:



Data flow for user’s POST, PATCH and DELETE request:

## File and folder structure

This is the source folder structure of the webserver



The connection.js is responsible for config database connection.

File structure of controller:

Ảnh có chứa văn bản

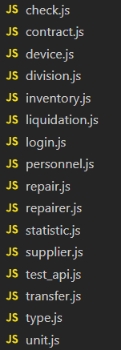
Mô tả được tạo tự động

File structure of model:

Ảnh có chứa văn bản

Mô tả được tạo tự động

File structure of routes:



## API structure

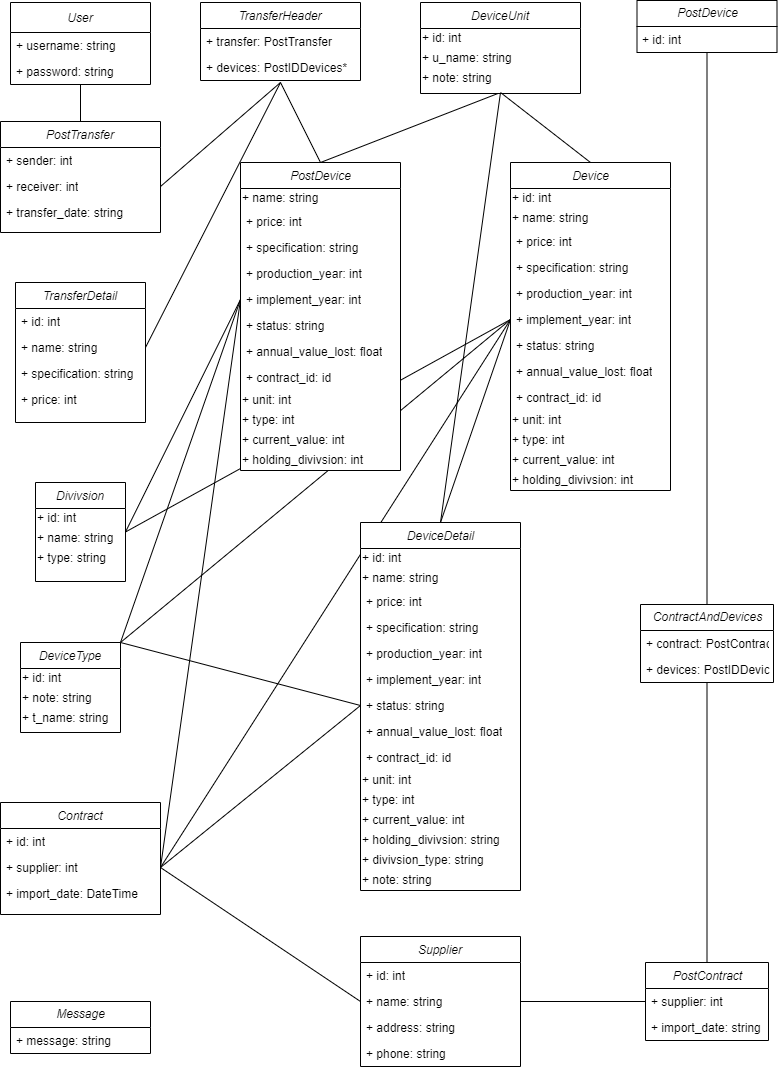
|  |  |  |
| --- | --- | --- |
| **HTTP method** | **API Relative URI** | **Function** |
| GET | /login/:username/:password | Return success message if login success and fail message if login fail |
| GET | /device/query?division|contract | Get device list by division or contract using URL query |
| GET | /contract | Get contract list |
| POST | /device/add | Create new contract with devices |
| GET | /supplier | Get supplier list |
| POST | /transfer/add | Transfer devices from division to other division |
| GET | /transfer/list | Get transfer record list |
| GET | /transfer/detail/:id | Get transfer device detail by id |
| GET | /check/list | Get check records |
| POST | /check/add | Update device status and store result to database |
| GET | /personnel | Get personnel list |
| POST | /personnel | Create new personnel |
| POST | /inventory/add | Create a new inventory |
| GET | /inventory/list | List inventory records |
| GET | /inventory/device/:id | List devices in an inventory |
| GET | /inventory/personnel/:id | List personnels in an inventory |
| GET | /liquidation/query?division | Get devices which need liquidating in a division |
| POST | /liquidation/add | Liquidate devices |
| GET | /liquidation/list | Get liquidation list |
| GET | /liquidation/devices/:id | Get liquidation device detail by id |
| GET | /liquidation/personnel/:id | Get liquidation personnel detail by id |
| GET | /repairer | Get list of repairer company |
| POST | /repairer | Create new repairer company |
| GET | /repair/spoiled | Get spoiled devices (need repairing) |
| GET | /repair/bill | Get repair bills |
| GET | /repair/detail/:id | Get bill detail by id |
| POST | /repair/add | Create repair bill and update devices status |
| GET | /statistic/year\_price | Get device repair money by year |

Conceptual model

Class diagram

# Class diagram

## Data

**

## WPF Client

Ảnh có chứa văn bản

Mô tả được tạo tự động

## List of classes and relationships in Database Module

|  |  |  |  |
| --- | --- | --- | --- |
| .No | Class/Relationship | Type | Note |
| 1 | User/Association |  | PostTransfer |
| 2 | TransferHeader/Association |  | PostTransfer  TransferDetail  PostDevice |
| 3 | PostTransfer/Association |  |  |
| 4 | TransferDetail/Association |  |  |
| 5 | PostDevice/Association |  | DeviceUnit  Divivsion  DeviceType  Contract |
| 6 | DeviceUnit/Association |  | Device  DeviceDetail |
| 7 | Device/Association |  | Divivsion  DeviceType  Contract  DeviceDetail |
| 8 | DeviceDetail/Association |  | DeviceType  Contract |
| 9 | Divivsion/Association |  |  |
| 10 | DeviceType/Association |  |  |
| 11 | Contract/Association |  | Supplier |
| 12 | Supplier/Association |  | PostContract |
| 13 | PostContract/Association |  | ContractAndDevices |
| 14 | ContractAndDevices/Association |  | PostDevice |
| 15 | PostDevice/Association |  |  |
| 16 | Message |  |  |

## List of classes and relationships in UI Module

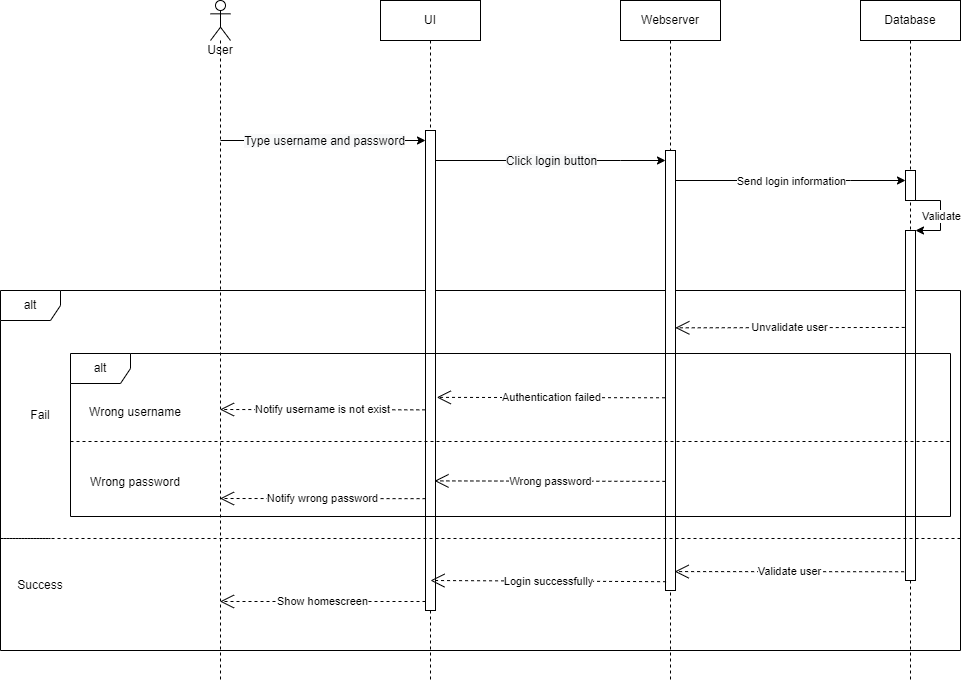
|  |  |  |  |
| --- | --- | --- | --- |
| .No | Class/Relationship | Type | Note |
| 1 | EditStatusWindow/Association |  | MainWindow |
| 2 | NewPersonnelWindow/Association |  | AddPersonnelWindow |
| 3 | LoginWindow/Association |  | MainWindow |
| 4 | MainWindow/Association |  | StatisticPage  LiquidationPage  InventoryPage  RepairPage  Devicepage |
| 5 | AddPersonnelWindow/Association |  | Establish\_Inventory\_CouncilWindow  Establish\_Liquidation\_CouncilWindow  InventoryDeviceWindow |
| 6 | StatisticPage/Association |  | StatictisDeviceCount  StatictisRepairPrice |
| 7 | LiquidationPage/Association |  | Establish\_Liquidation\_CouncilWindow  LiquidationHistoryWindow  LiquidationDetailWindow |
| 8 | Establish\_Liquidation\_CouncilWindow/Association |  | LiquidationDetailWindow |
| 9 | LiquidationDetailWindow/Association |  | LiquidationHistoryWindow |
| 10 | LiquidationHistoryWindow/Association |  |  |
| 11 | InventoryPage/Association |  | Establish\_Inventory\_CouncilWindow  InventoryDetailWindow |
| 12 | Establish\_Inventory\_CouncilWindow/Association |  | InventoryDeviceWindow |
| 13 | InventoryDetailWindow/Association |  |  |
| 14 | InventoryDeviceWindow/Association |  | EditStatusInventoryWindow |
| 15 | EditStatusInventoryWindow/Association |  |  |
| 16 | RepairPage/Association |  | RepairListWindow  Choose\_Repairer |
| 17 | RepairListWindow/Association |  | RepairListDetail |
| 18 | RepairListDetail/Association |  |  |
| 19 | Choose\_Repairer/Association |  | InsertRepairWindow  EnterRepairPriceWindow |
| 20 | InsertRepairWindow/Association |  |  |
| 21 | EnterRepairPriceWindow/Association |  |  |
| 22 | Devicepage/Association |  | DeviceDetailWindow  CheckWindow  DeviceEditingWindow  NewContractWindow  TransferWindow |
| 23 | DeviceDetailWindow/Association |  |  |
| 24 | CheckWindow/Association |  | CheckListWindow  StartCheckingWindow |
| 25 | CheckListWindow/Association |  | CheckListDetail |
| 26 | CheckListDetail/Association |  |  |
| 27 | StartCheckingWindow/Association |  |  |
| 28 | DeviceEditingWindow/Association |  |  |
| 29 | NewContractWindow/Association |  | AddDevicesWindow  ContractWindow |
| 30 | AddDevicesWindow/Association |  |  |
| 31 | ContractWindow/Association |  |  |
| 32 | TransferWindow/Association |  | ListTransferWindow |
| 33 | ListTransferWindow/Association |  | TransferDetailWindow |
| 34 | TransferDetailWindow/Association |  |  |
| 35 | StatictisDeviceCount/Association |  |  |
| 36 | StatictisRepairPrice/Association |  |  |

# Cohesion and coupling

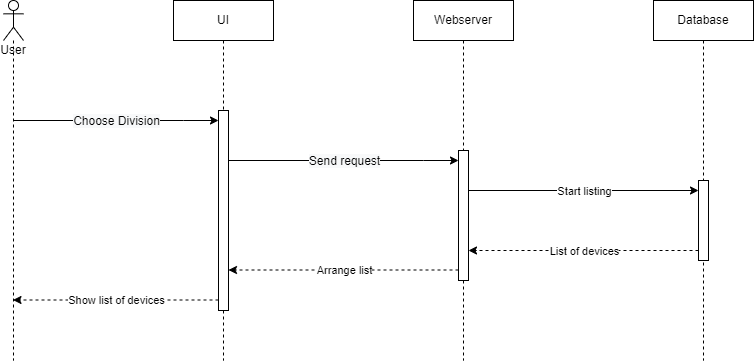
* Coupling in this diagram: Data Coupling because the three modules communicate with each other by transferring data only.
* Cohesion in this diagram: Sequential Cohesion because data in database module flow between elements.

# Sequence diagram*.*

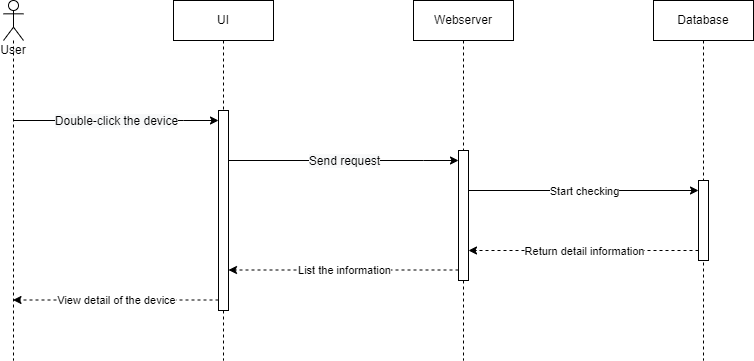
## Login

**

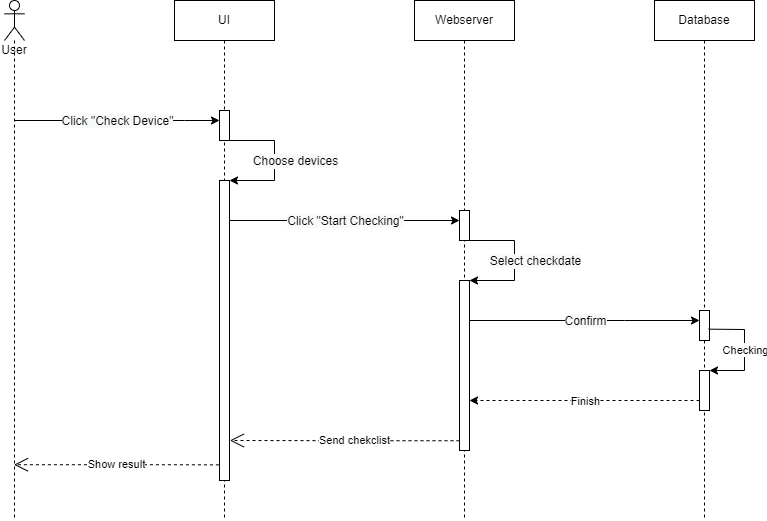
## Device List



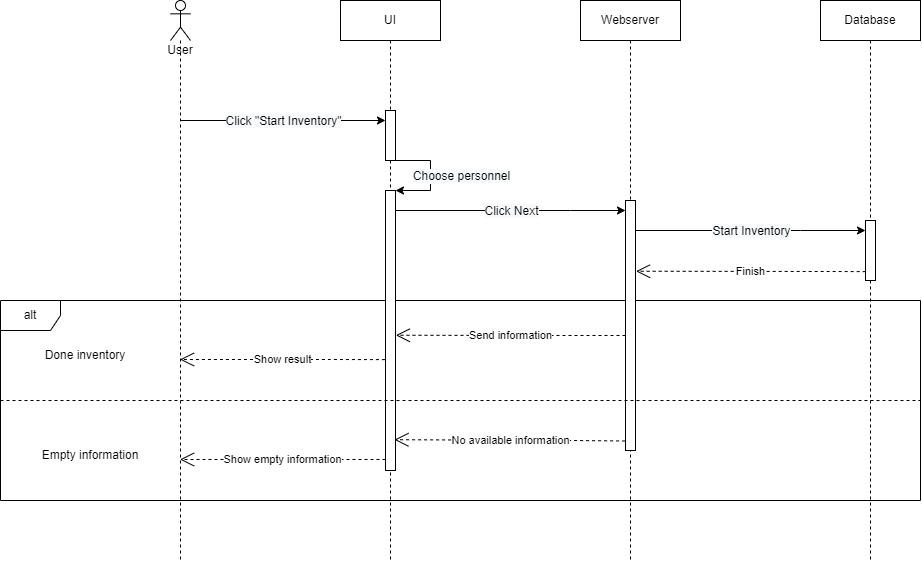
## Device Detail



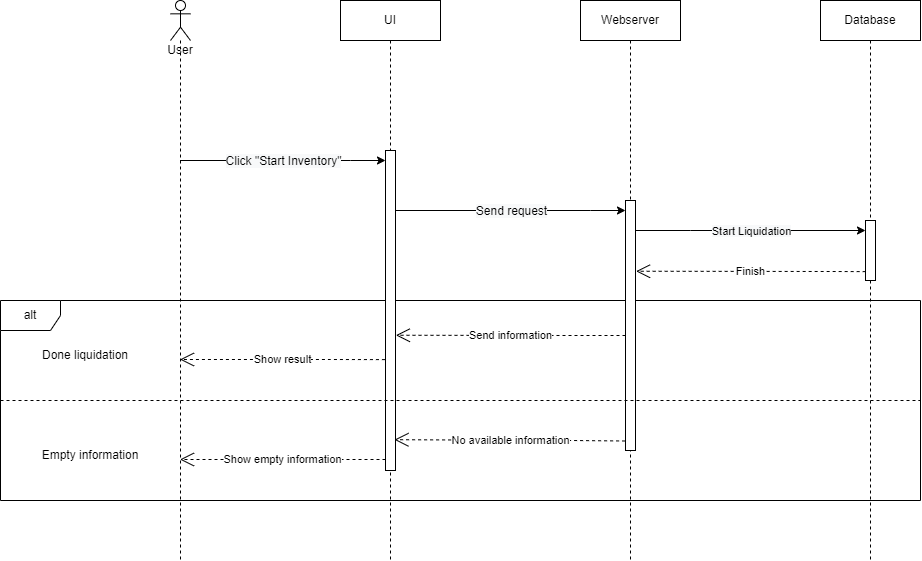
## Check Device



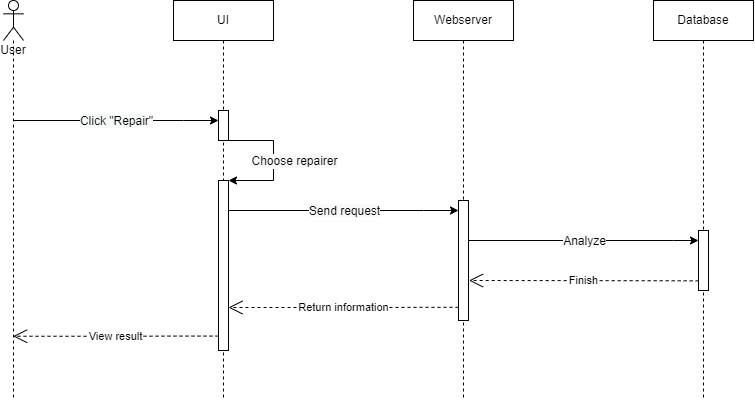
## Inventory



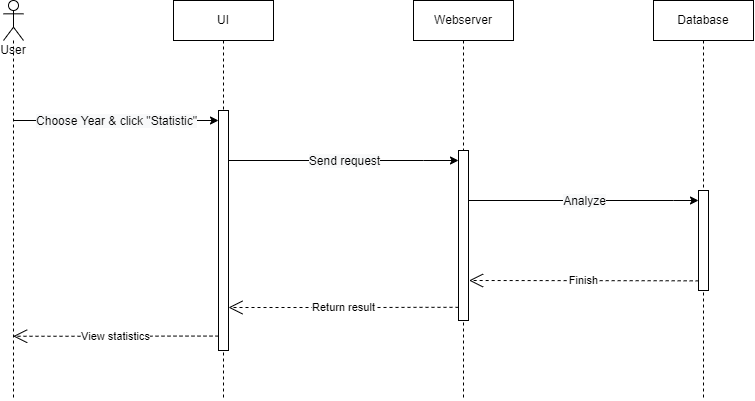
## Liquidation



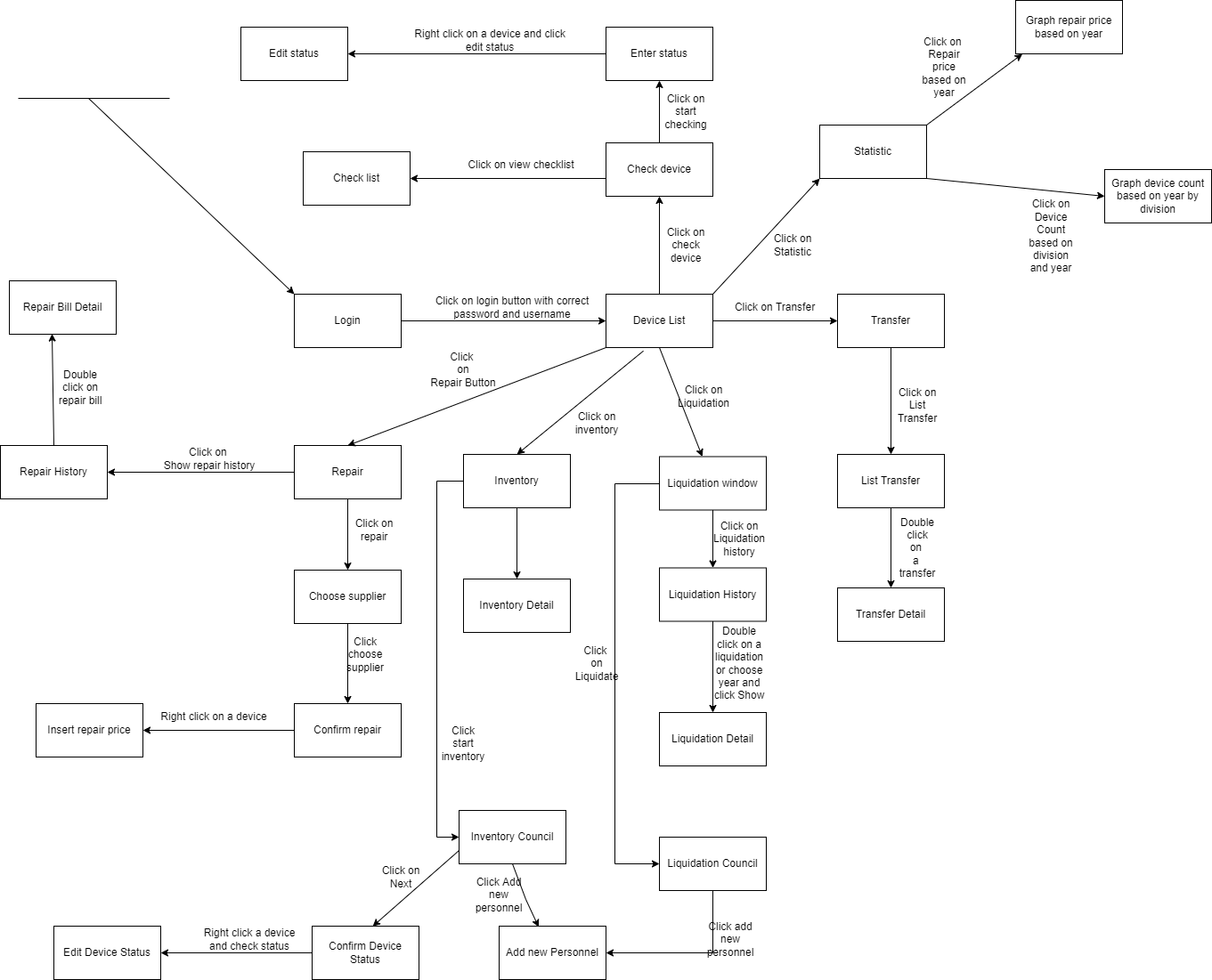
## Repair



## Statistics



Screen Flow Diagram



Screen Detail

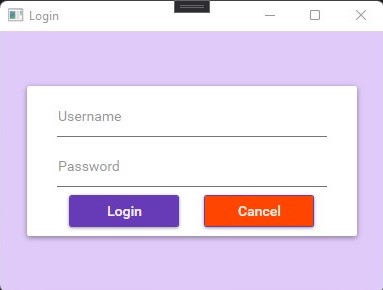
Due to the limited length of this document, we don’t mention all screen listed in the screen flow diagram above.

# Login Window

## Main Flow

|  |  |
| --- | --- |
| Screen | Screen for login |
| Description | Login |
| Screen Access | On Application Opening On clicking Log out button |

## UI



## Data Mapping / Validation Rule

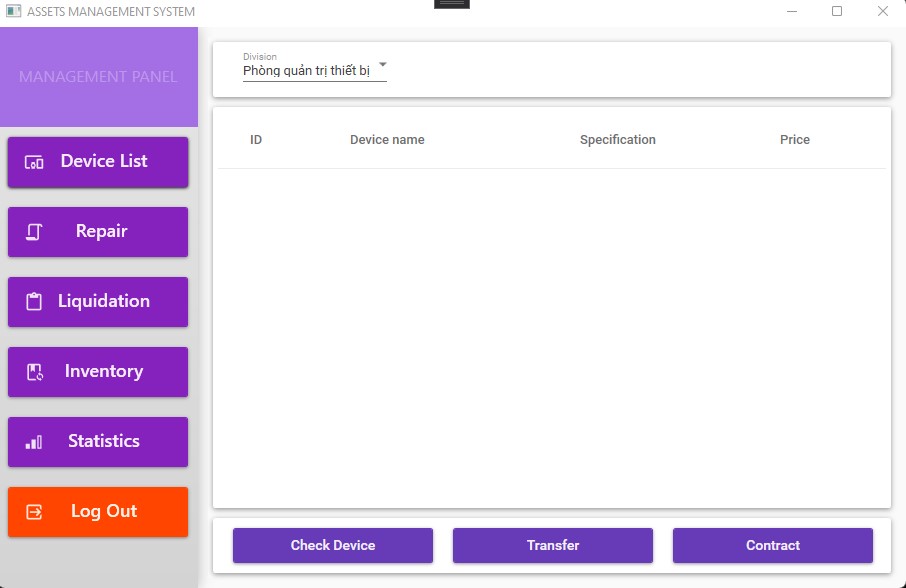
|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Username | Textbox | Input Username |
| Password | PasswordBox | Input Password |
| Login | Button | Click to Login into system  Use GET /login to check whether validation is valid |
| Cancel | Button | Exit the program |

# Main Window

## Main Flow

|  |  |
| --- | --- |
| Screen | Home Screen after log in successfully |
| Description | Device Page + Main Screen  Contain all devices’ information |
| Screen Access | On logging in On clicking Device List button from Main Window |

## UI



## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Device List | Button | Refresh the page to get data updated |
| Repair | Button | Switch to Repair page |
| Liquidation | Button | Switch to Liquidation page |
| Inventory | Button | Switch to Inventory page |
| Statistics | Button | Switch to Statistics page |
| Log Out | Button | Return to Login page |
| Division | Combobox | Choose division want to view devices from |
| Device List View | ListView | Contains data of remaining division’s devices |
| Check Device | Button | Navigate to Check Device Window |
| Transfer | Button | Navigate to Transfer Window |
| Contract | Button | Navigate to Contract Window |

# Inventory Page

## Main Flow

|  |  |
| --- | --- |
| Screen | Inventory Page |
| Description | Contain inventories’ information |
| Screen Access | On clicking Inventory button |

## UI

Ảnh có chứa văn bản

Mô tả được tạo tự động

## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Device List | Button | Switch to Device List page |
| Repair | Button | Switch to Repair page |
| Liquidation | Button | Switch to Liquidation page |
| Inventory | Button | Refresh the page to get data updated |
| Statistics | Button | Switch to Statistics page |
| Log Out | Button | Return to Login page |
| Division | Combobox | Choose division want to view devices from |
| Inventory List View | ListView | Contains data of the whole inventory information |
| Start Inventory | Button | Start checking the inventory |

# Liquidation Page

## Main Flow

|  |  |
| --- | --- |
| Screen | Liquidation Page |
| Description | Contain liquidations’ information |
| Screen Access | On clicking Liquidation button |

## UI

Ảnh có chứa văn bản

Mô tả được tạo tự động

## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Device List | Button | Switch to Device List page |
| Repair | Button | Switch to Repair page |
| Liquidation | Button | Refresh the page to get data updated |
| Inventory | Button | Switch to Inventory page |
| Statistics | Button | Switch to Statistics page |
| Log Out | Button | Return to Login page |
| Choose Division | Combobox | Choose division want to view devices from |
| Liquidation List View | ListView | Contains data of the whole liquidation information |
| Show Liquidation History | Button | View history of liquidations |
| Start Liquidation | Button | Start checking the inventory |

# Repair Page

## Main Flow

|  |  |
| --- | --- |
| Screen | Repair Page |
| Description | Contain repair information |
| Screen Access | On clicking Repair button |

## UI

Ảnh có chứa văn bản

Mô tả được tạo tự động

## Data Mapping / Validation Rule

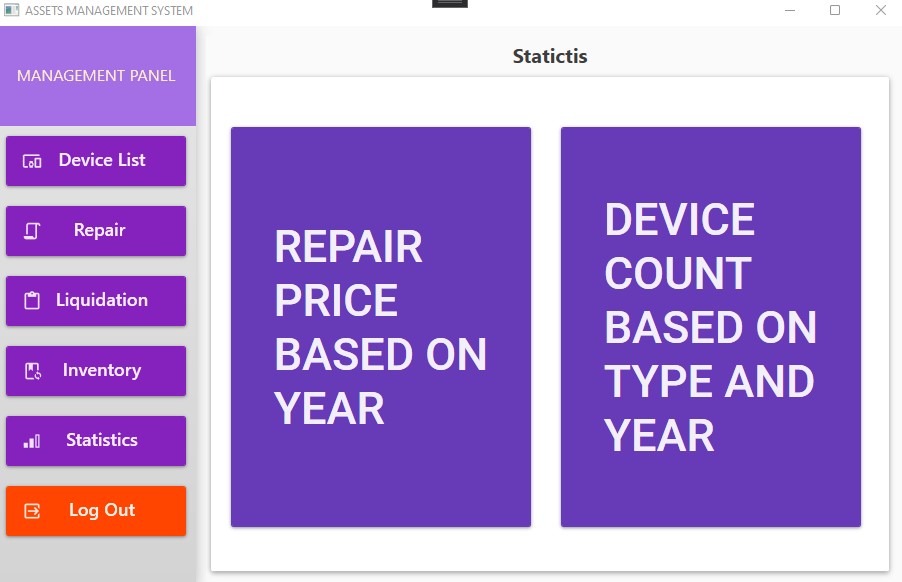
|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Device List | Button | Switch to Device List page |
| Repair | Button | Refresh the page to get data updated |
| Liquidation | Button | Switch to Liquidation page |
| Inventory | Button | Switch to Inventory page |
| Statistics | Button | Switch to Statistics page |
| Log Out | Button | Return to Login page |
| Repair List View | ListView | Contains data of the whole repair information |
| Show Repair History | Button | View history of repairments |
| Repair | Button | Start checking the repairment |

# Statistic Page

## Main Flow

|  |  |
| --- | --- |
| Screen | Statistic Page |
| Description | Contain statistics |
| Screen Access | On clicking Statistics button |

## UI



## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Device List | Button | Switch to Device List page |
| Repair | Button | Switch to Repair page |
| Liquidation | Button | Switch to Liquidation page |
| Inventory | Button | Switch to Inventory page |
| Statistics | Button | Refresh the page to get data updated |
| Log Out | Button | Return to Login page |
| Repair Price | Button | View statistics sort by year |
| Device Count | Button | View statistics sort by type and year |

# Add Personnel Window

## Main Flow

|  |  |
| --- | --- |
| Screen | Add Personnel Window |
| Description | Choose another personnel in the system |
| Screen Access | On clicking Add Personnel button |

## UI

Ảnh có chứa bàn

Mô tả được tạo tự động

## Data Mapping / Validation Rule

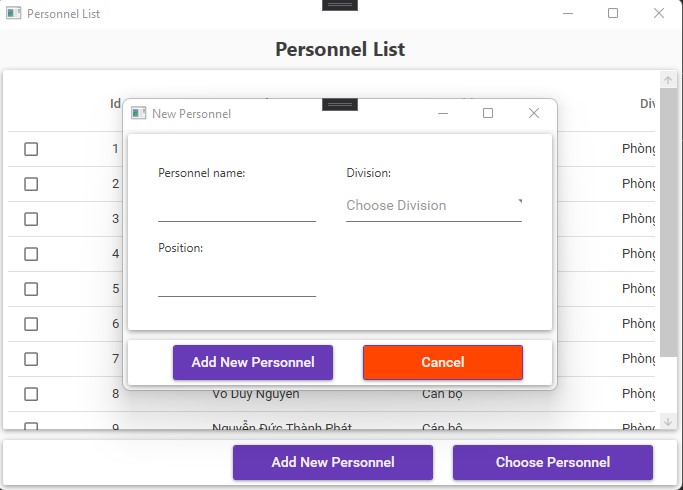
|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Personnel List View | ListView | View list of personnel |
| Add New Personnel | Button | Navigate to New Personnel Window |
| Choose Personnel | Button | Confirm choosing the personnel |

# New Personnel Window

## Main Flow

|  |  |
| --- | --- |
| Screen | New Personnel Window |
| Description | Add new personnel to the system |
| Screen Access | On clicking Add New Personnel button |

## UI



## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Personnel name | TextBox | Type the personnel’s name |
| Position | TextBox | Type the personnel’s position |
| Division | ComboBox | Choose the personnel’s division |
| Add New Personnel | Button | Add the personnel |
| Cancel | Button | Cancel the action |

# Edit Status Window

## Main Flow

|  |  |
| --- | --- |
| Screen | Edit Status Window |
| Description | Change device status |
| Screen Access | Right-click the device want to edit |

## UI

Ảnh có chứa văn bản

Mô tả được tạo tự động

## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Choose Status | ComboBox | Choose the personnel’s division |
| Edit Status | Button | Confirm edition |

# Inventory Detail Window

## Main Flow

|  |  |
| --- | --- |
| Screen | Inventory Detail Window |
| Description | View detail of each inventory |
| Screen Access | Double-click the inventory |

## UI

Ảnh có chứa bàn

Mô tả được tạo tự động

## Data Mapping / Validation Rule

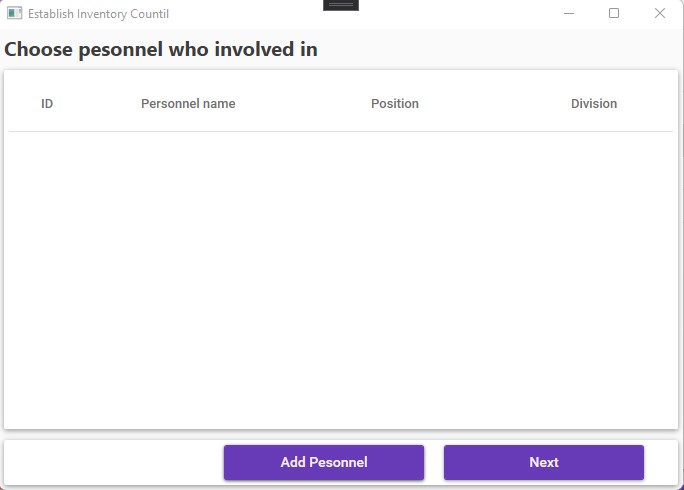
|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Inventory List View | ListView | View detail of inventory |
| Close | Button | Close the window |

# Establish Inventory Council Window

## Main Flow

|  |  |
| --- | --- |
| Screen | Establish Inventory Council Window |
| Description | Establish Inventory |
| Screen Access | On clicking Start Inventory button |

## UI



## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Personnel List View | ListView | View list of personnel |
| Add Personnel | Button | Navigate to New Personnel Window |
| Next | Button | Navigate to Inventory Device Window |

# Inventory Device Window

## Main Flow

|  |  |
| --- | --- |
| Screen | Inventory Device Window |
| Description | List Inventory Devices |
| Screen Access | On clicking Next button |

## UI

Ảnh có chứa văn bản

Mô tả được tạo tự động

## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Select Checkdate | DatePicker | Choose checkdate |
| Device List View | ListView | View list of inventory devices |
| Finish Inventory | Button | Start the inventory |

Ảnh có chứa văn bản

Mô tả được tạo tự động

## Data Mapping / Validation Rule

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| Choose Status | ComboBox | Change the device status |
| Edit Status | Button | Confirm edition |

Installation

# System Requirement

* .NET Framework v5.0
* Window 7/10/11 64bit
* At least 1GB or RAM, 5GB of memory
* Internet connection

# How to Install

* Download Installer in this [link](https://drive.google.com/file/d/1K7SfnHui-u5wLYU8zgfXyRzhg5sKyHew/view?usp=sharing)
* Open the installer
* Follow Installer’s instruction

Demo

Demo will be operated in the last day.

Future Update

The software need to be updated in some point:

* Security: We haven’t use any security method to protect our API data. In the future we will use Session Token login and Open Authentication to ensure the security of the application
* User Interface: Some point of the application the UI hasn’t been optimized. We will update the UI of the application to increase the user experience
* More feature: In the future’s update, we will add more feature such as : account role, real-time statistic, PDF printing, ….
* Refactor and Optimize current source code: The source code is not well structure, so in the future we will optimize and refactor the code so that the software is easy to maintain and operate.
* Fix current existing bugs.

Team member assignment

|  |  |  |
| --- | --- | --- |
| **Name** | **Code** | **Responsibility** |
| Phung Thanh Tu | 19522455 | Project Manager, Team Leader, Backend Developer, Write Backend Documentation, Support Team member to fix bug or to accomplish the features |
| Le Thanh Tuan | 19522467 | Frontend Developer, Design UI, Frontend Tester |
| Pham Hieu Vy | 19520358 | Write Documentation, Design and Develop Database, Test API and Database. |

Lesson Learned

Through this project and this course, we have learned:

* Design the architecture is important: It’s very important for us to design the software architecture before start coding. It saves time to develop and we can spot lethal bugs before developing the application – which phase fixing bugs cost more.
* Time management is important: Time and deadlines management is the key to make the project finished. Member follow the timeline and the project will be accomplished effectively.
* Meeting daily is an approach to ensure the project’s success: We figured out that meeting daily keep the track of the project and make project finish sooner than when we don’t have a meeting every 9.00 AM
* How to working online: In the pandemic condition, we can’t group together at a coffee to start meeting offline and work together offline, so we must do everything online and therefore we learned how to work effectively even online in the pandemic condition. This is one of the important thing which helps us a lot in the future.