CMP307 – SOFTWARE ENGINEERING PRACTISE

REPORT

EWAN STEWART

1900598

Contents

[Design 4](#_Toc90055871)

[Requirements 4](#_Toc90055872)

[Functional 4](#_Toc90055873)

[Non-Functional 4](#_Toc90055874)

[UML 4](#_Toc90055875)

[Use Case 4](#_Toc90055876)

[Class Diagram 7](#_Toc90055877)

[Sequence Diagram 7](#_Toc90055878)

[Implementation 9](#_Toc90055879)

[Development Methodology 9](#_Toc90055880)

[Testing 10](#_Toc90055881)

[Unit Testing 10](#_Toc90055882)

[Asset Display 10](#_Toc90055883)

[Asset Create 10](#_Toc90055884)

[Asset Update 10](#_Toc90055885)

[Asset Delete 11](#_Toc90055886)

[Asset Link 11](#_Toc90055887)

[Vulnerability API Call 11](#_Toc90055888)

[Database Backup 11](#_Toc90055889)

[Usability Testing 12](#_Toc90055890)

[Tester 1 12](#_Toc90055891)

[Tester 2 12](#_Toc90055892)

[Reliability & Security 13](#_Toc90055893)

[Fault Avoidance 13](#_Toc90055894)

[Fault Tolerance 13](#_Toc90055895)

[Security Measures 13](#_Toc90055896)

[Account Credentials 14](#_Toc90055897)

[Report only user: **username: report, password: HX{D9h8\** 14](#_Toc90055898)

[Access all user: **username: admin, password: FD2-au\*+** 14](#_Toc90055899)

[Software Guide 15](#_Toc90055900)

[Login 15](#_Toc90055901)

[Control Panel 15](#_Toc90055902)

[15](#_Toc90055903)

[15](#_Toc90055904)

[Logout 15](#_Toc90055905)

[15](#_Toc90055906)

[Asset 16](#_Toc90055907)

[Create 16](#_Toc90055908)

[16](#_Toc90055909)

[Delete 16](#_Toc90055910)

[Update 16](#_Toc90055911)

[Update 17](#_Toc90055912)

[Display Assets 17](#_Toc90055913)

[NIST API Call 18](#_Toc90055914)

[18](#_Toc90055915)

[Backup Database 18](#_Toc90055916)

[Evaluation 19](#_Toc90055917)

# Design

## Requirements

### Functional

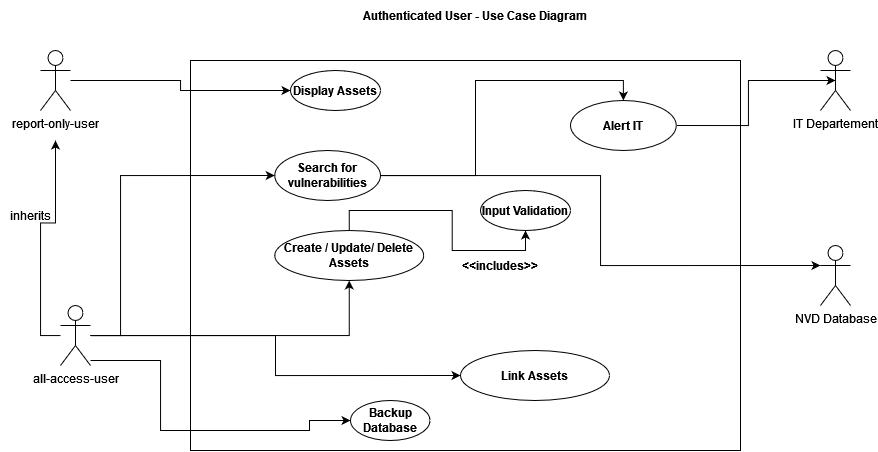
1. The system must allow the user to create / update / delete both software and hardware assets.
2. The system must allow the user to link a hardware asset to a software asset.
3. The system must prevent invalid data entry at the point of entry.
4. The system must have a clear instruction available for how to correctly enter data.
5. The system must run on Windows.
6. The system must authenticate users before they allowed access to the system.
7. The system must allow functionality for a report only account where this user can only access the asset listing feature.
8. The system must be able to provide local backups for the asset data.
9. The system must allow vulnerability reporting on the assets.

### Non-Functional

1. The system must make data entry for the user easy.
2. The system must be reliable and secure.
3. The system must make forwarding vulnerability data to the IT department easy.

## UML

### Use Case



Search for vulnerabilities

Main Success Scenario:

1. The user is authenticated by the system.
2. The user selects the search for vulnerabilities button
3. The user is presented with a list of all known vulnerabilities related to the data stored in the database.

Extensions:

3a. There is no vulnerabilities found.

1. The user is prompted that there is no known vulnerabilities found.

Display Assets

Main Success Scenario:

1. The user is authenticated by the system.
2. The user selects the display assets button.
3. The user is presented with tables of software and hardware assets.

Extensions:

3a. There is no data in the software and hardware asset tables.

1.The user is prompted that there is no data stored on software and hardware assets.

3b. There is data in only one of the asset tables. **(e.g., there is data in the hardware table but not software)**

1.The user is shown only the table with data.

### 

Delete Asset

Main Success Scenario:

1. The user is authenticated by the system.
2. The user selects the delete asset button.
3. The user selects either hardware or software from a drop-down box.
4. The user enters an asset ID into either the hardware or software form.
5. The user selects delete asset.

Extensions:

4a. The user enters an invalid asset ID

1. The user is prompted the asset ID is invalid and no data is shown on the screen.

4b. The user enters an invalid character in an input box

1.The last character entered is removed

4c. The user’s input is too long

1.The last character entered is removed

Create Asset

Main Success Scenario:

1. The user is authenticated by the system.
2. The user selects the create asset button.
3. The user selects either hardware or software from a drop-down box.
4. The user enters data in either the hardware or software form.
5. The user selects create asset.

Extensions:

4a. The user enters an invalid character in an input box

1. The last character entered is removed

4b. The user’s input is too long

1. The last character entered is removed

5a. The user enters an invalid MAC address

1. The user is prompted that their MAC is invalid and the create request is not processed.

5b. The user enters an invalid IP address

1. The user is prompted that their IP is invalid and the create request is not processed.

Link Asset

Main Success Scenario:

1. The user is authenticated by the system.
2. The user selects the link assets button.
3. The user enters the asset IDs to link.
4. The user selects link assets.

Extensions:

3a. The user enters an invalid asset ID.

1.The user is prompted that they entered an invalid ID.

3b. The user enters an invalid character in an input box.

1.The last character entered is removed

4a. The assets are already linked.

1.The user is prompted that the assets are already linked.

Backup

Main Success Scenario:

1. The user is authenticated by the system.
2. The user selects the backup asset button.
3. The user is prompted that the backup was successful.

Extensions:

3a. The backup failed.

1.The user is prompted that the backup failed

Update Asset

Main Success Scenario:

1. The user is authenticated by the system.
2. The user selects the update asset button.
3. The user selects either software or hardware from a drop-down box.
4. The user enters an asset ID into either the hardware or software form.
5. The relevant data to the asset ID is displayed.
6. The user can now modify the displayed data.
7. The user selects update asset.

Extensions:

4a. The user enters an invalid asset ID

1. The user is prompted the asset ID is invalid and no data is shown on the screen.

4b, 6a. The user enters an invalid character in an input box

1.The last character entered is removed

6b. The user’s input is too long

1.The last character entered is removed

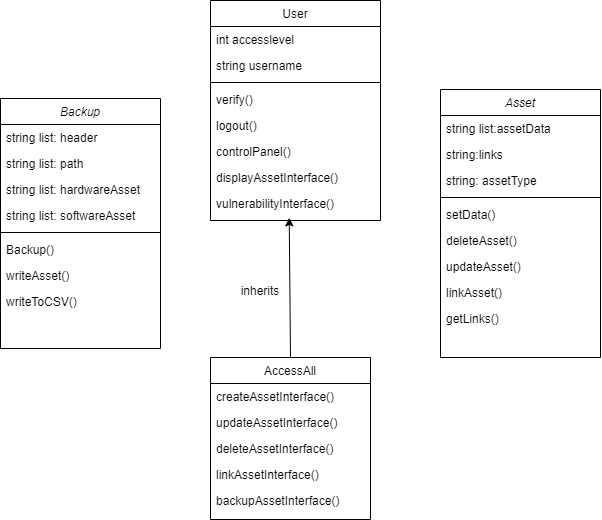
7a. The user enters an invalid MAC address

1.The user is prompted that their MAC is invalid and the create request is not processed.

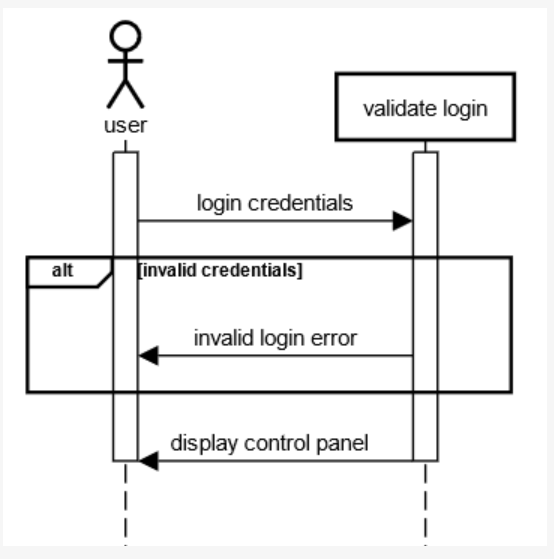
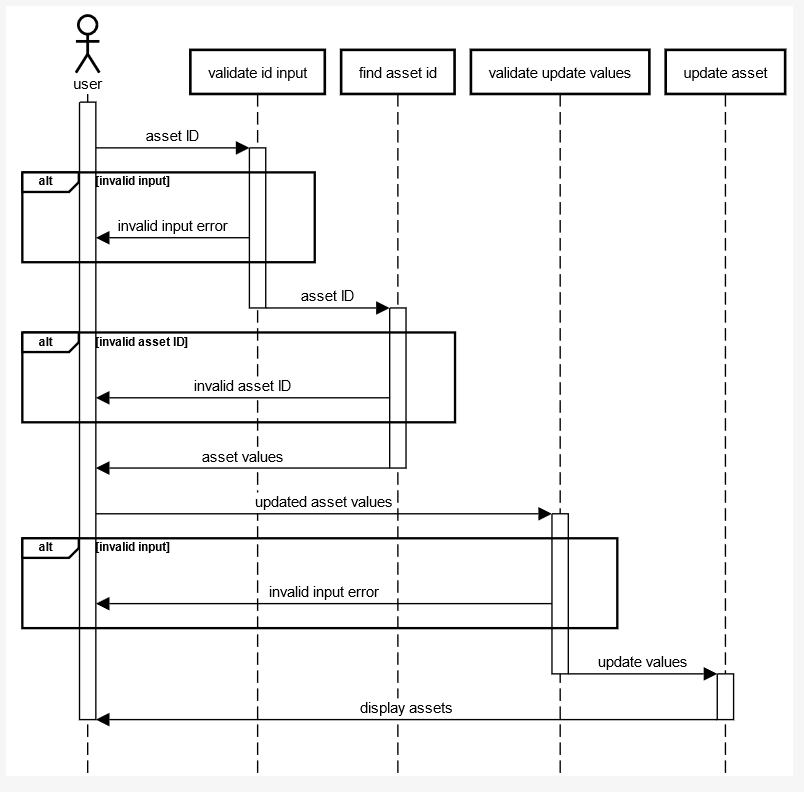
7b. The user enters an invalid IP address

1.The user is prompted that their IP is invalid and the create request is not processed.

### Class Diagram

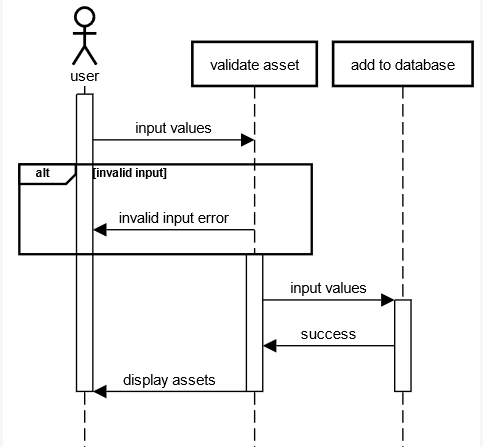


### Sequence Diagram

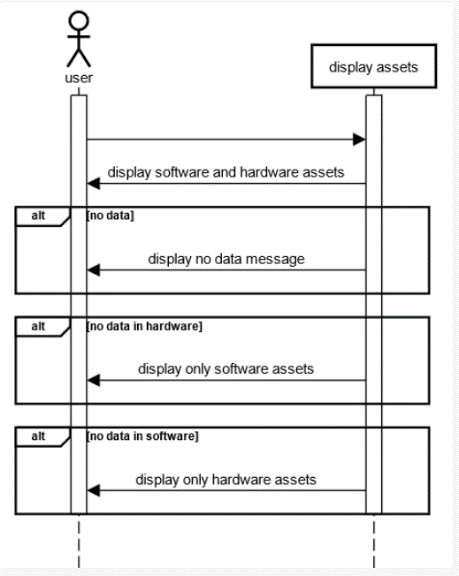


Login

Update asset

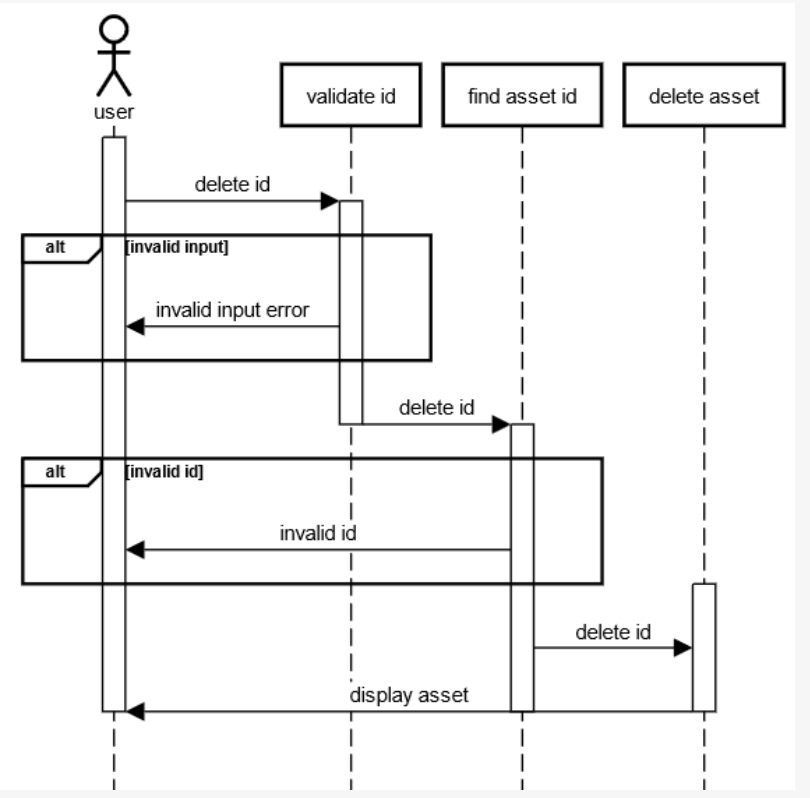


Create asset

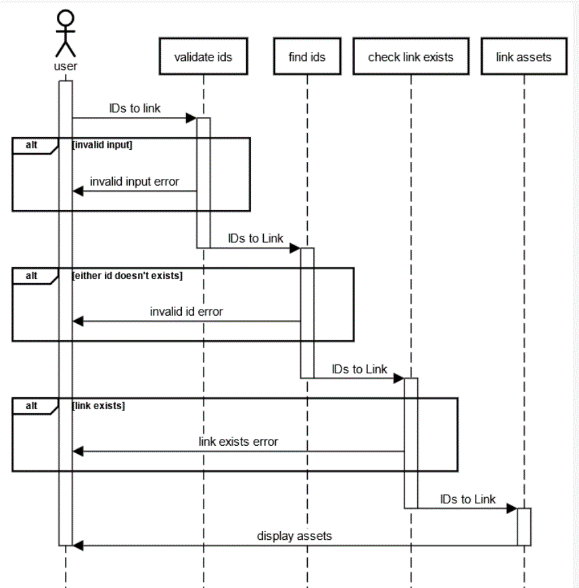


Display asset

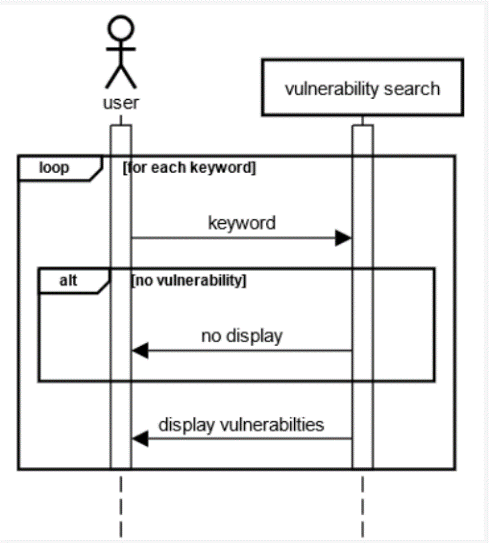
Delete asset



Link assets



Vulnerability search



# Implementation

## Development Methodology

To develop this application, I decided to follow an agile Kanban methodology as it allowed me to keep track of my tasks in an efficient and clean manner.

I started my journey of developing this application for Scottish Glen by breaking down the initial brief into individual work packages and plotting out the timeline of what order these packages needed to be completed in. Once I had my conceptual chronology, I mapped out the work packages into a simple Gannt Chart. I took each work package and refined it into simpler steps and began constructing my Kanban board. I opted to keep a physical Kanban board as I felt it gave me greater visualisation of tasks to be completed over a virtual board and as I was the only participant in the project. I approached this project by creating independent prototypes for each requirement and then piecing them together after all the prototypes were considered complete.

Initial first round Kanban board for creating login functionality:

Shape

Description automatically generated

Throughout development version control was present using GIT. Version control was proven as a valuable technology as it allowed the synchronisation and backup of the source code of this project.

This project is available at[**https://github.com/cmp307/project-EwanStewart.git**](https://github.com/cmp307/project-EwanStewart.git)

Commit history is available in the root of the project as log.txt.

# Testing

## Unit Testing

|  |
| --- |
| Asset Display |
| |  |  |  | | --- | --- | --- | | Input | Expected output | passed (y/n) | | no assets | ‘no data’ | y | | only software assets | display only software table | y | | only hardware assets | display only hardware table | y | | hardware and software assets | display both tables | y | | click search for vulnerabilities button | call vulnerability API | y | |

|  |
| --- |
| Asset Create |
| |  |  |  | | --- | --- | --- | | Input | Expected output | passed (y/n) | | Set asset type to hardware | hardware creation fields enabled | y | | set asset type to software | software creation fields enabled | y | | enter a banned character in any input | banned character removed from input | y | | enter a length greater than input expected | last character removed from input | y | | enter an invalid MAC address | Invalid MAC error appears | y | | enter an invalid IP address | Invalid IP error appears | y | | click create asset with all valid inputs | asset added to database and taken to display assets | y | |

|  |
| --- |
| Asset Update |
| |  |  |  | | --- | --- | --- | | Input | Expected output | passed (y/n) | | Set asset type to hardware | hardware update ID field enabled | y | | set asset type to software | software update ID field enabled | y | | enter a banned character in ID input | banned character removed from input | y | | click find asset with invalid asset ID | Invalid ID error appears | y | | click find asset with valid asset ID | hardware or software fields enabled, and data associated to ID fills the input boxes | y | | enter a banned character in any input | banned character removed from input | y | | enter a length greater than text box expected | last character removed from input | y | | enter an invalid MAC address | Invalid MAC error | y | | enter an invalid IP address | Invalid IP error | y | | click update asset | asset updated to database and taken to display assets | y | |

|  |
| --- |
| Asset Delete |
| |  |  |  | | --- | --- | --- | | Input | Expected output | passed (y/n) | | Set asset type to hardware | hardware delete ID field enabled | y | | set asset type to software | software delete ID field enabled | y | | enter a banned character in ID input | banned character removed from input | y | | click delete asset with invalid asset ID | Invalid ID error appears | y | | click delete asset with valid asset ID | asset deleted from database and taken to display assets | y | |

|  |
| --- |
| Asset Link |
| |  |  |  | | --- | --- | --- | | Input | Expected output | passed (y/n) | | enter a banned character in ID inputs | banned character removed from input | y | | click link asset with invalid hardware ID | Invalid hardware ID error appears | y | | click link asset with invalid software ID | Invalid software ID error appears | y | | click link asset with valid IDs | Asset link added to database taken to display assets | y | | click link assets when already linked | Already linked error appears | y | |

|  |
| --- |
| Vulnerability API Call |
| |  |  |  | | --- | --- | --- | | Inputs | Expected output | passed (y/n) | | No keywords in database | ‘no data found’ | y | | 1 keyword in database | displays list box with relevant vulnerabilities and number of them. Vulnerabilities are dumped in text file in ../nist/NIST DUMP | y | | >5 keywords in database | same output as 1 keyword provided however screen becomes scrollable. | y | |

|  |
| --- |
| Database Backup |
| |  |  |  | | --- | --- | --- | | Inputs | Expected output | passed (y/n) | | select backup button | software and hardware files created in ../backups/ | y | |

## Usability Testing

I had my testing group complete the System Usability Scale Survey to gather data on the usability of the application. All testers were first time users of the application and are not considered to be from a technical background.

### Tester 1

Tester 1 scored a SUS score of 80: which is considered good on the SUS scale. Tester 1 critiqued the colour of the error messages and recommended to make them standout more. I implemented this by changing the error messages from white to red.

### Tester 2

Tester 2 scored a SUS score of 68: which is considered average on the SUS scale. Tester 2 critiqued that it was not clear where the backups of the assets were being stored. I decided to make a pop-up box of the path of the backups after backup completion to increase usability.

Overall, I was satisfied with the results of my usability survey. There is a disparity between the SUS scores however, tester 2 commented that they feel with a little training they would be more confident in operating the system.

# Reliability & Security

### Fault Avoidance

Avoidance features have been implemented within this application to prevent the user causing a system breaking error.

|  |  |
| --- | --- |
| Fault | Avoidance Measure |
| The user inputs a value into an input field that is larger than anticipated. | Each input box has a set size declared and when the user enters a character a check is made to determine whether the combined input is over the character limit. If so, the last character is removed. |
| The user inputs the wrong type of data into an input field. | Where possible the user is given a pre-set choice of inputs (like seen in device type and calendar field) to make their input decision however, where this isn’t possible each input field has a regex statement attached to it which determines whether the user’s last inputted character is permitted or not. If not, the last character is removed. |
| The user refers to an asset ID that does not exist. | When interacting with assets in update and asset link checks are done to ensure that the asset IDs that are being referred to exist before the user can modify them. |

### Fault Tolerance

Several faults could occur when this interacting with this application. Although these faults can’t be directly fixed by the system, the system has measures in place to tolerate them. The following potential faults have been identified and measures have been put into place to ensure the stability of the application.

|  |  |
| --- | --- |
| Fault | Tolerance Measure |
| SQL failure | The system alerts the user that the SQL operation that was requested has failed and that they should attempt the operation again. |
| Database backup failure | The system alerts the user that the backup operation has failed. |
| Input validation failure | The system removes the user’s input entirely if it detects a validation failure |

### Security Measures

To ensure the integrity of Scottish Glen’s asset tracking system security measures have been implemented into the application.

All users of the application must be authenticated before accessing the operations of the system. Scottish Glen’s system has two different account types: report only and access all. The report only account can only access the display asset and search for vulnerabilities functionality where the access all can use all features of the system.

All SQL database operations are structured using prepared statements to prevent user input from directly being placed within a query. This will mitigate the risk of the Scottish Glen’s database being victim to a SQL injection attack.

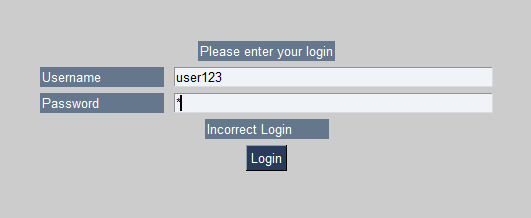
## Account Credentials

### Report only user: **username: report, password: HX{D9h8\**

### Access all user: **username: admin, password: FD2-au\*+**

# Software Guide

## Login



The first screen the user will be presented with is the login screen where they will have to enter their username or password to gain access to the system.

The user should enter their credentials and then click login.

If the user enters incorrect credentials, they will be prompted with an incorrect login error message.

The user should re-enter their login and attempt to login again.

## Control Panel

## 

Once the user has gained access to the system, they will be presented with two different types of control panels dependant on their account access level.

The user will then be taken to the operation they click.

## 

Report Only

Access All

## Logout

## 

Once in the control panel, the user can click log out which will return them to the login screen and destroy their current session.

## Asset

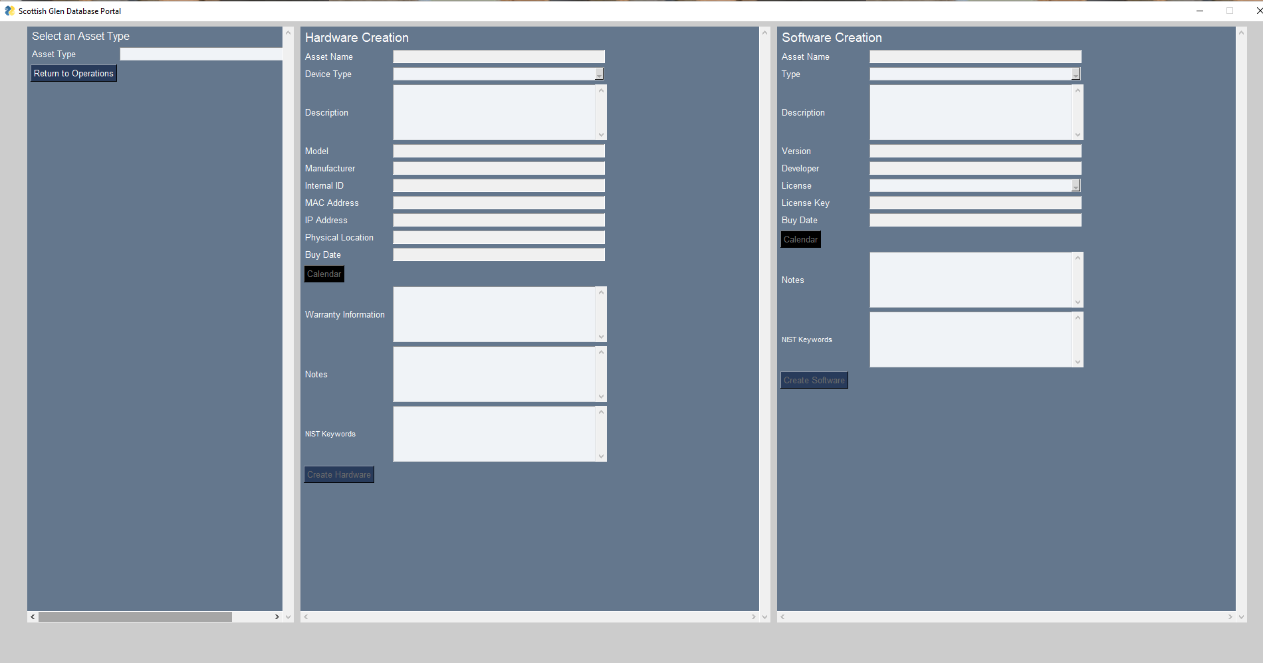
### Create

In the create asset feature the user should firstly select the type of asset they would like to create on the far-left panel.

The decision will then enable the fields for either the hardware or software panels.

The user can then proceed to type in the fields and select create hardware/software at the bottom of the panel.

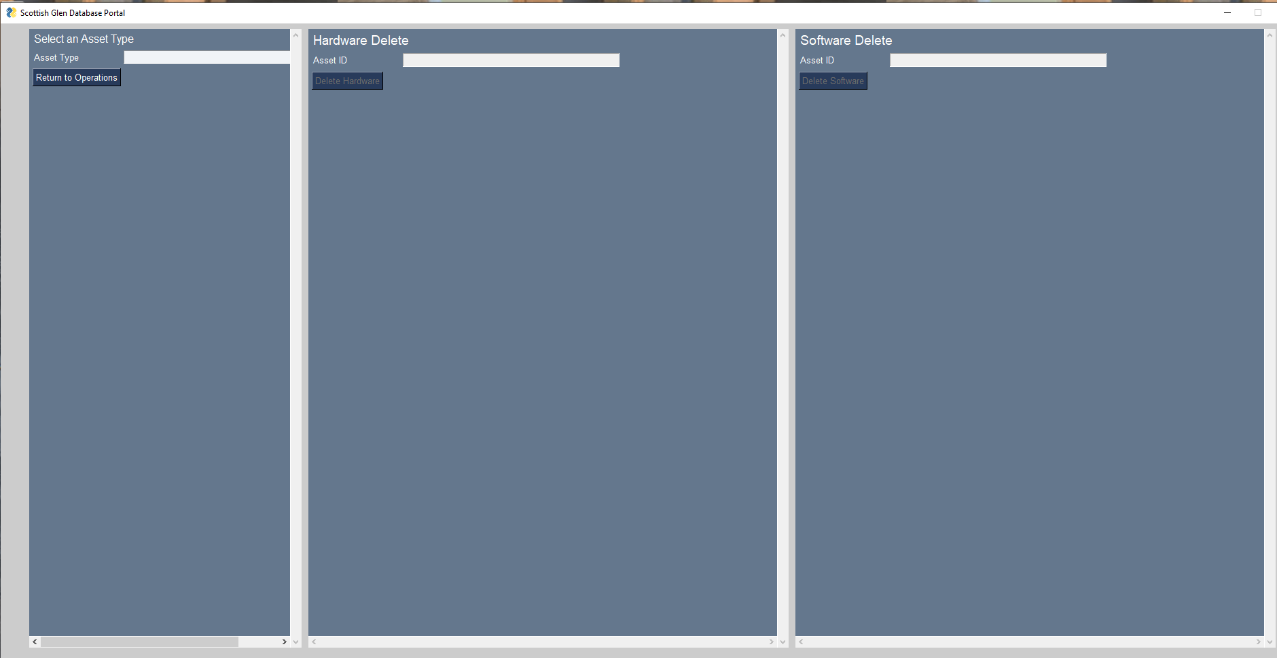
Each input has character restrictions however, the system will deny invalid characters so the user should not have any concern.



When the user selects create hardware if the IP or MAC address input is not correctly structured an error message will appear and the user will have to correct their input to proceed. An IP address is structured (X.X.X.X) where the characters are integers, and a MAC address is structured (XX:XX:XX:XX:XX:XX) where the characters are hexadecimal.







Delete

In the delete asset feature the user should firstly select the type of asset they would like to create on the far-left panel.

The decision will then enable the fields for either the hardware or software panels.

The user can then proceed to type in the id field and select delete hardware/software at the bottom of the panel.

The ID field is restricted to integers. If the user attempts to delete an asset that doesn’t exist, they will be shown the invalid ID error.



### Update

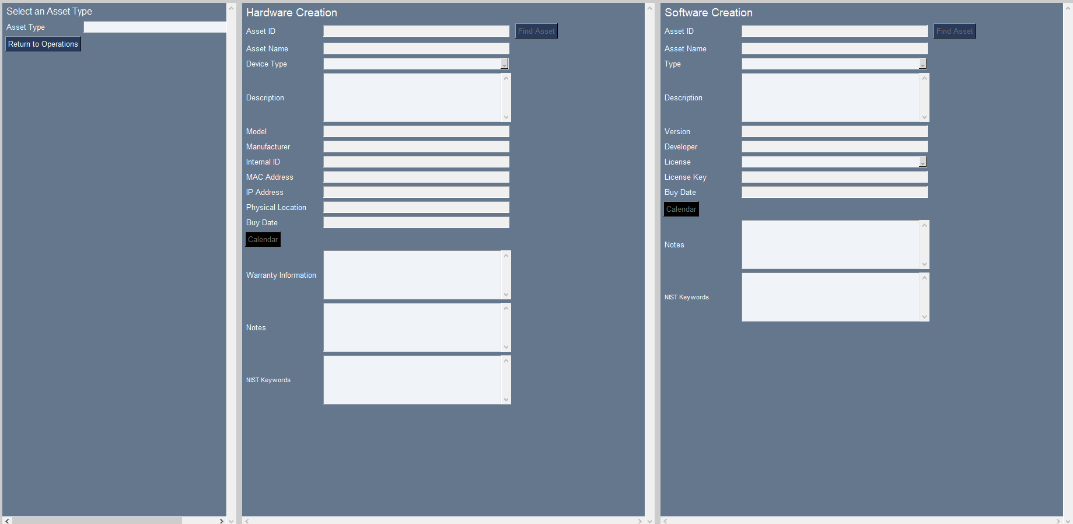
### Update

In the update asset feature the user should firstly select the type of asset they would like to create on the far-left panel.

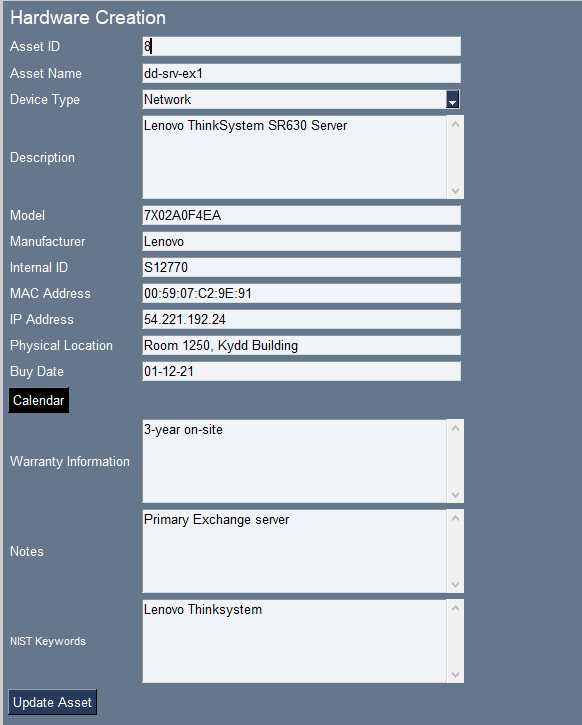
The decision will then enable the fields for either the hardware or software panels.

The user can then proceed to type in the ID field and then select find hardware/software to the right of the ID field.

The ID field is restricted to integers. If the user attempts to find an asset that doesn’t exist, they will be shown the invalid ID error.





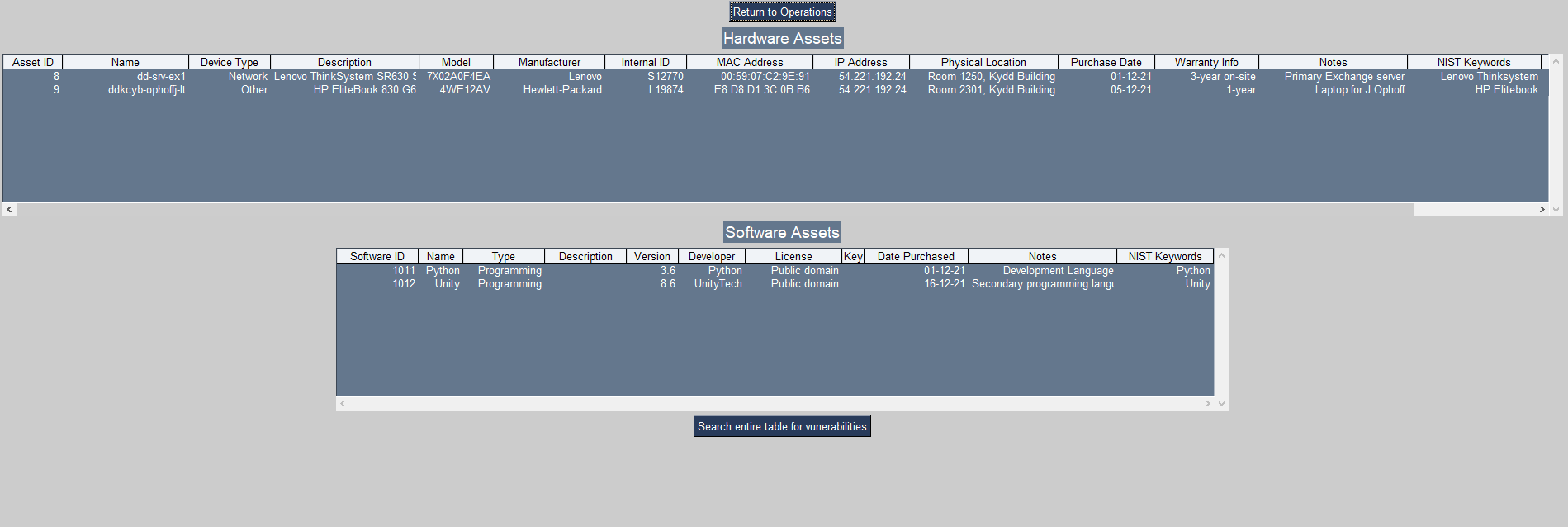


When the user successfully finds an asset, they will be shown an updated panel.

The user can then modify any of the fields and select update asset.

Like in the create asset the MAC address and the IP must conform to the correct structure to be accepted.

## Display Assets



In the display assets feature the user is shown two tables the top table showing hardware assets and the bottom software assets.

The user has the option to search for vulnerabilities using the NIST API at the bottom of the screen.

## NIST API Call

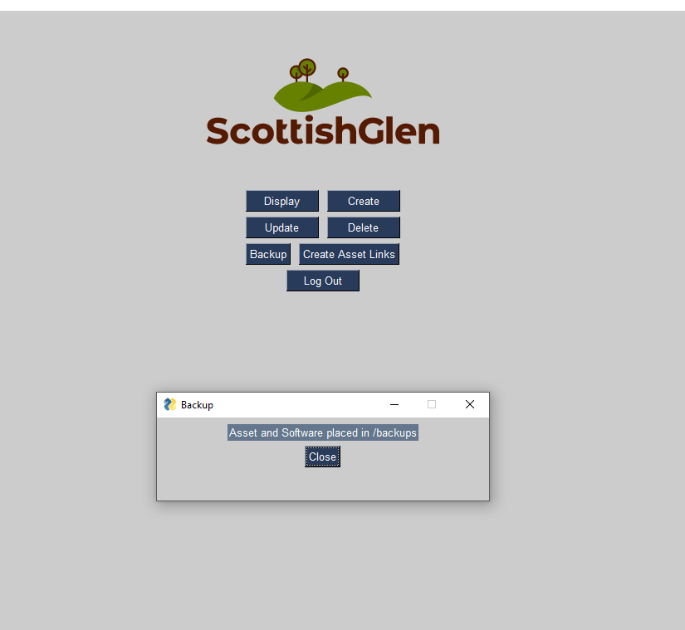
## 

In the vulnerabilities feature all keywords from the database are queried against the NIST API and the results are shown.

Each asset is given its own list box and if multiple assets there is the ability to scroll on far-right hand side of the screen.

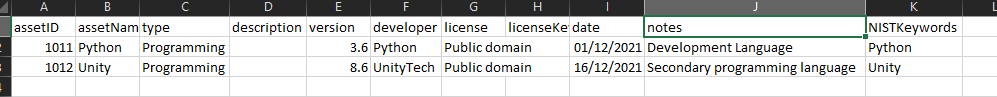
The results for the NIST API call are dumped to a text file located in ../nist/NIST dump.txt which will allow for easy forwarding to the IT department.

## Backup Database



In the backup feature all assets are processed into a CSV file. Assets are broken into hardware and software where they are given their own CSV file.

Backups are located in ../backups/



# Evaluation

Upon reflection of this project, it can be determined that the system is fit for purpose. The system performs all the specified requirements sufficiently and has measures in place to make the system robust and prevent the user from creating errors.

Part 1 of the project created no significant development issues however, a point to consider in the future is to consider using a development language which allows the binding of functions and input validation via the development language’s GUI. Developing with Python provided little help regarding these points and required a lot of surplus code to handle these events in comparison to a language like C#.

The addition of part 2 of the project to the requirements created slight inconvenience as the interfaces had to be re-designed to incorporate the functionality of having software assets in addition to hardware assets. This was overcome by incorporating a drop-down box to allow the user to choose which type of asset they would like to interact with.