IT Asset Tacking – Design Document

last updated: 28-02-2025

Author: Roth Thomas

Overview

The IT Asset Tracking Project is a multi project solution to manage hardware and software assets, assigning them to employees or departments as well as keeping track of IT support tickets and reading various statistics data like asset values, license usage, ticket resolution times and more.

Context

This project should provide options for managing and tracking assets, support tickets and reading various reports.

The project should provide an Interface that is easy to navigate and easy understand created with ASP.Net MVC. It also provides a complete API that allows for implementation of an external front end.

Both interface and API should have proper Security in Place.

Solution

The Solution consist of multiple projects:

Core Layer

Contains interfaces, models and entities that are shared across the solution

Data Layer

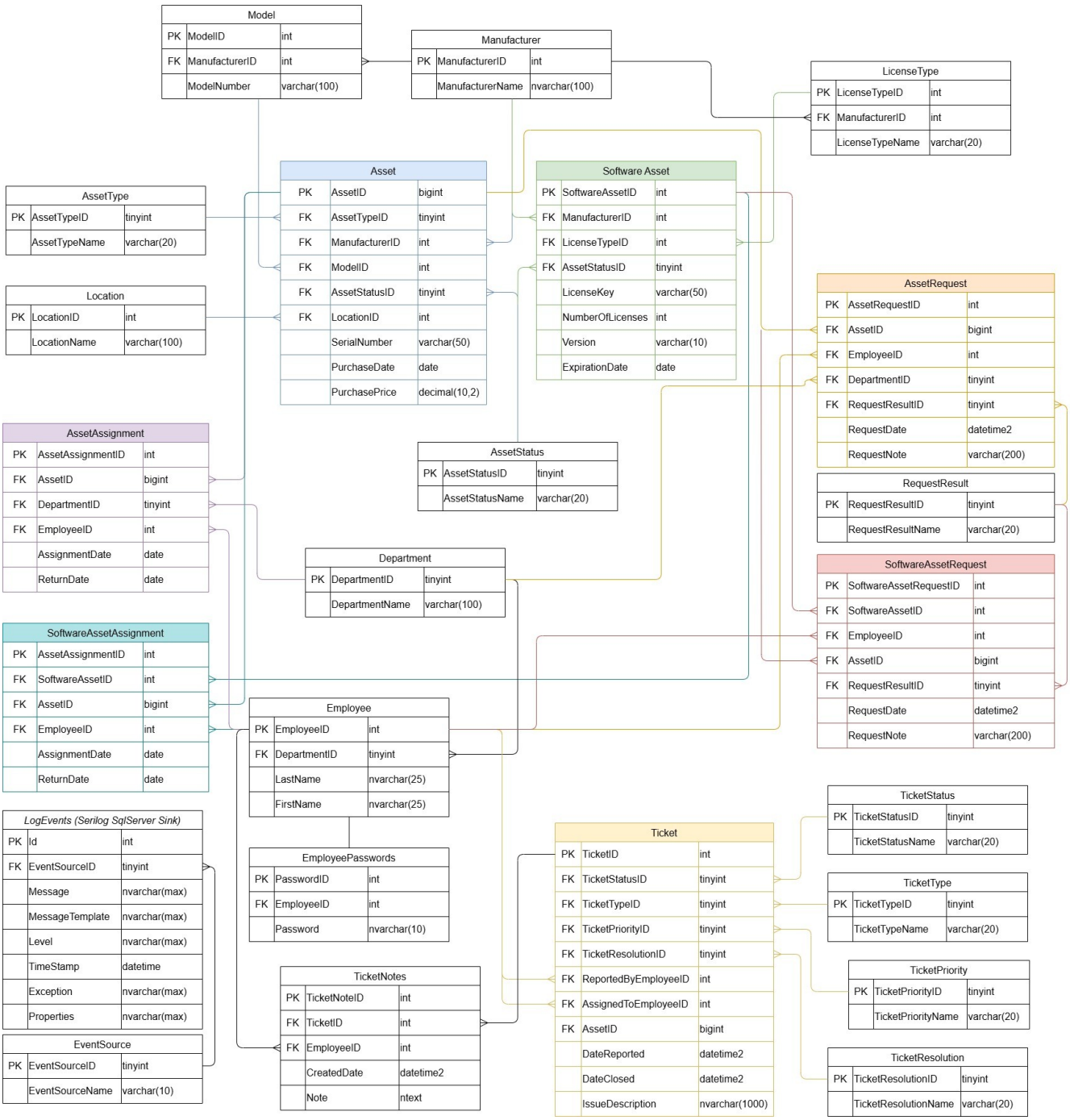
Connects to the Database for adding, retrieving, updating and deleting data.

I choose Entity Framework to query the DB as I expect to write A LOT of queries. A full ORM like EF will speed up creating and updating the repositories for the Data Layer. Also this project represents an application for a possibly large company where other choices like Dapper, as a third party package, could potentially be unreliable.

I made necessary adjustments to the provided ERD to meet the project's requirements.

Tech:

- SqlServer Database
- Entity Framework



IT Asset Tracking Project - ERD

Application Layer

Contains testable business logic between the Data Layer and MVC / API Layer.

The application class methods return “result” types:

{

Ok: bool,

Data?: <T>,

message: string,

Ex?: Exception

}

which are consumed by the API and MVC Layer.

That way data and errors are being passed to the upper layers where they are handled properly.

Testing Layer

Contains unit tests for any testable logic of the Application Layer

Tech:

- Nunit
- Microsoft Test Sdk

Model View Controller

Here the interface of the web application will be created using MVC workflow and Razor syntax.

Users need to authenticate via Username and Password to view any of the sites content.

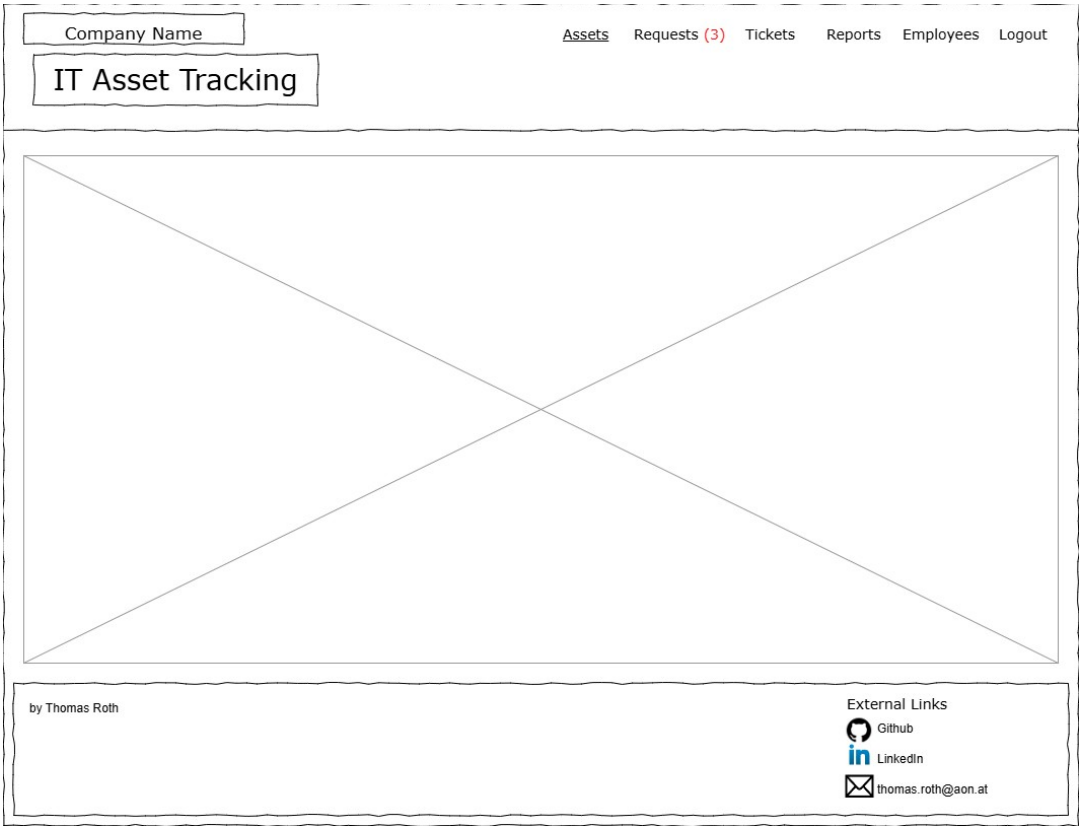
Users can have one of the following roles assigned to them:

- **Admin:** Access to all sites content and functionalities, can add or update users
- **Asset Manager:** Can add, edit and delete assets, assign assets, generate asset reports.
- **Software License Manager:** Manages software assets and their assignment, generate software compliance reports
- **Help Desk Technician:** Manage support tickets. Creating, updating, assigning and adding notes to tickets.
- **Department Manager:** View assets assigned to their department, request new assets or reassignments, view ticket status for their department's assets
- **Employee:** View and create support tickets for their assigned assets.
- **Auditor:** Read-Only access to all asset and ticket information, generate and view any reports.

For the styling I will leverage **bootstrap** for responsive design.

The following pages will showcase the interface wireframes.

The header and footer are static while the main section is dynamic.



The Header contains links to the main sections of the Service.

- **Assets** => leads to the Assets Index Page
- **Requests** => Shows the number of open asset requests. Leads to the asset request page accessible to *Asset Managers* and *Admins* and *read-only access for Department Managers.*
- **Tickets** => Accessible to *Help Desk Technicians*. Leads to the Tickets Index Page
- **Reports** => Accessible to *Auditors*. Leads to the Reports Index Page
- **Employees** => Accessible to Admins only. Leads to Employee List.

The following wireframes represent the main section for each path.

The **Assets Index** Page simply contains three links.

“My Assets”, “Department Assets”, “All Assets“

companyitassets/employee/23/assets

Assets assigned to Doe, John

Assets

Model	Type	Status	
CR-123	Card Reader	Repair	Details
G83-6105	Keyboard	In Use	Details
GZ-2	Stationary Phone	In Use	Details
MSBOM	Mouse	In Use	Details
SE2422HX	Monitor	In Use	Details
Wacom K100986	Graphic Display	In Use	Details

Software Licenses

License	Version	Status	
Zbrush	2025.1.1	In Use	Details

The **‘My Assets’** link redirects to a list of assets assigned to the currently logged in user. The list is ordered alphabetically by Model. As this should be a fairly short list of assets I choose to not implement a filter form here to avoid unnecessary cluttering.

The **‘Department Assets’** link redirects to the page shown below. This page is accessible to **department managers, asset managers and admins** and shows all assets assigned to the selected department. For department managers the *‘Department Dropdown’* is greyed out and defaults to the department of the currently logged in user.

companyitassets/assets/department/2/hardware

Department: Dep. 1B

Order: Status

Type: Any

Manufacturer: Any

Status: Any

Search

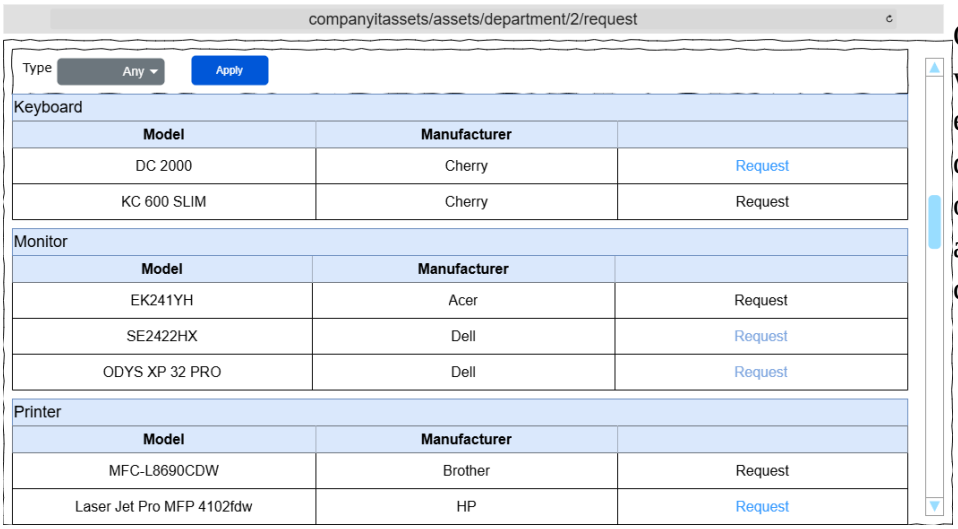
Apply

Hardware Assets | Software Assets | Request New Asset

Dep. 1B Assets

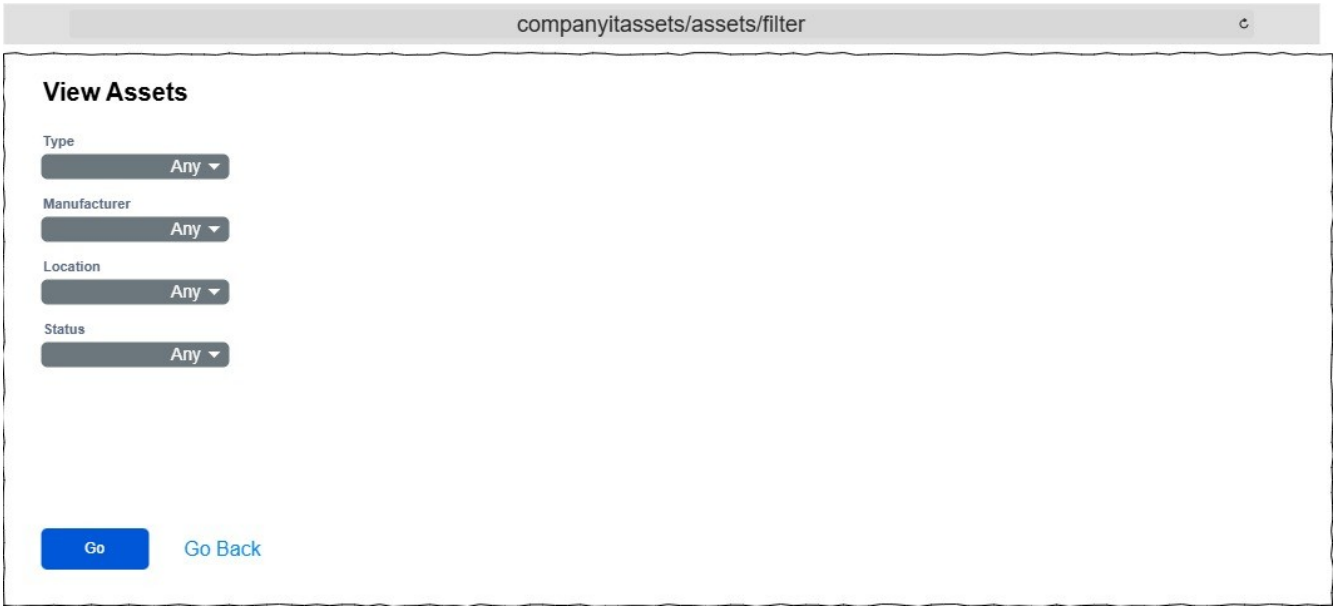
Serial Nr.	Model	Type	Manufacturer	Status	
DM-1B-01	SE2422HX	Monitor	Dell	In Use	Details
DM-1B-02	SE2422HX	Monitor	Dell	In Use	Details
MS-1B-02	MSBOM	Mouse	Microsoft	In Use	Details
KB-1B-01	G83-6105	Keyboard	Cherry	In Use	Details
KB-1B-01	G83-6105	Keyboard	Cherry	In Use	Details
DM-1B-01	SE2422HX	Monitor	Dell	In Use	Details
DM-1B-02	SE2422HX	Monitor	Dell	In Use	Details
KB-1B-01	G83-6105	Keyboard	Cherry	In Use	Details
KB-1B-08	G83-6105	Keyboard	Cherry	Repair	Details
KB-1B-01	G83-6105	Keyboard	Cherry	Repair	Details
MS-1B-01	MSBOM	Mouse	Microsoft	Storage	Details

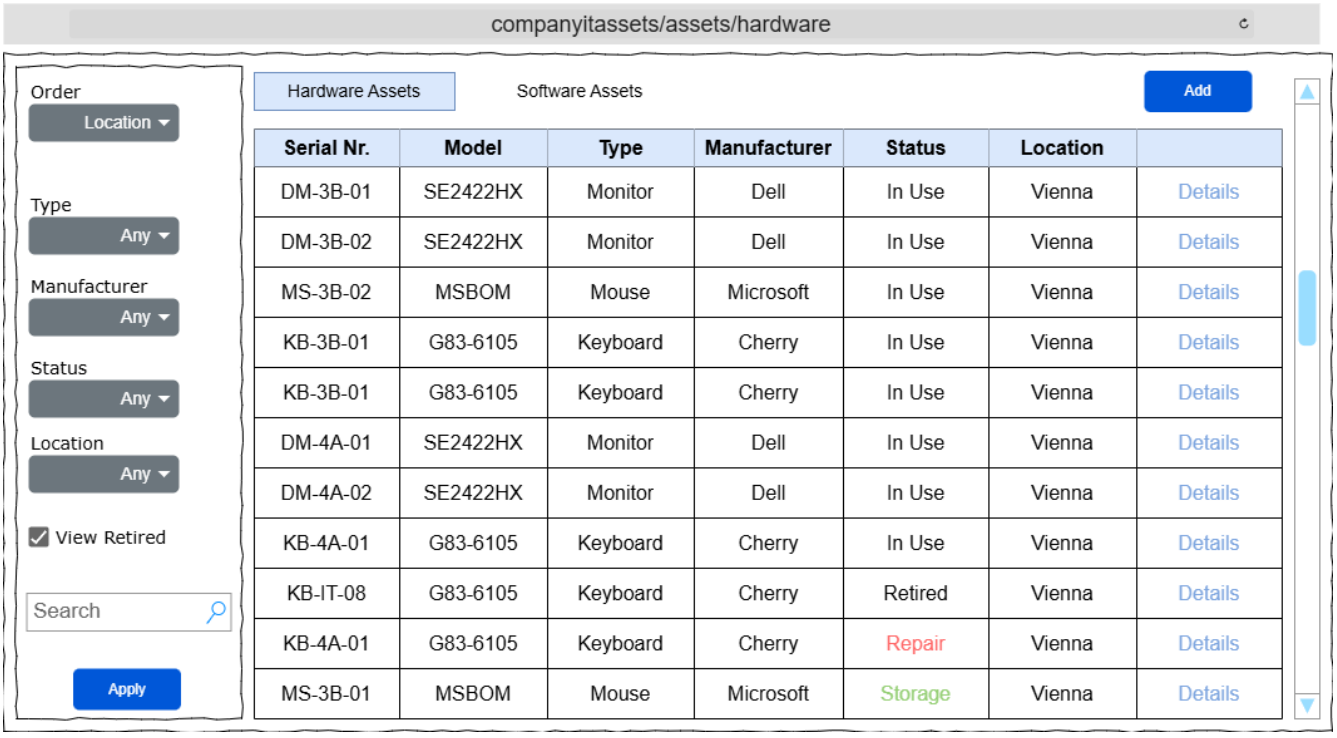
The ‘Request New Asset’ button redirects to a list of available or unassigned assets.



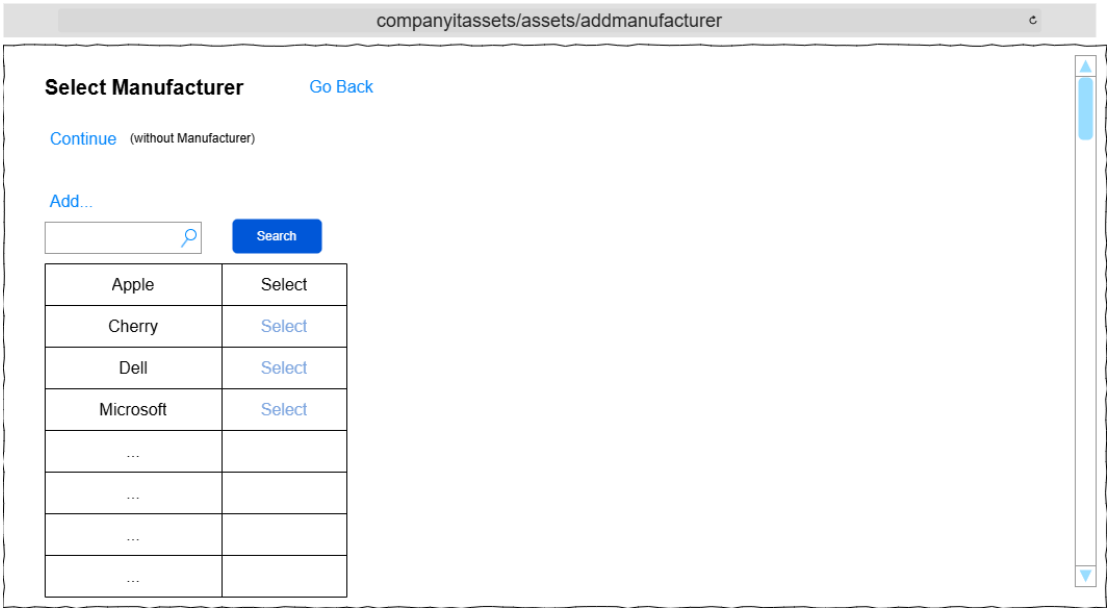
Clicking the Request Link will then redirect to a list of employees of the current department where optionally an employee to assign the requested asset to can be selected.

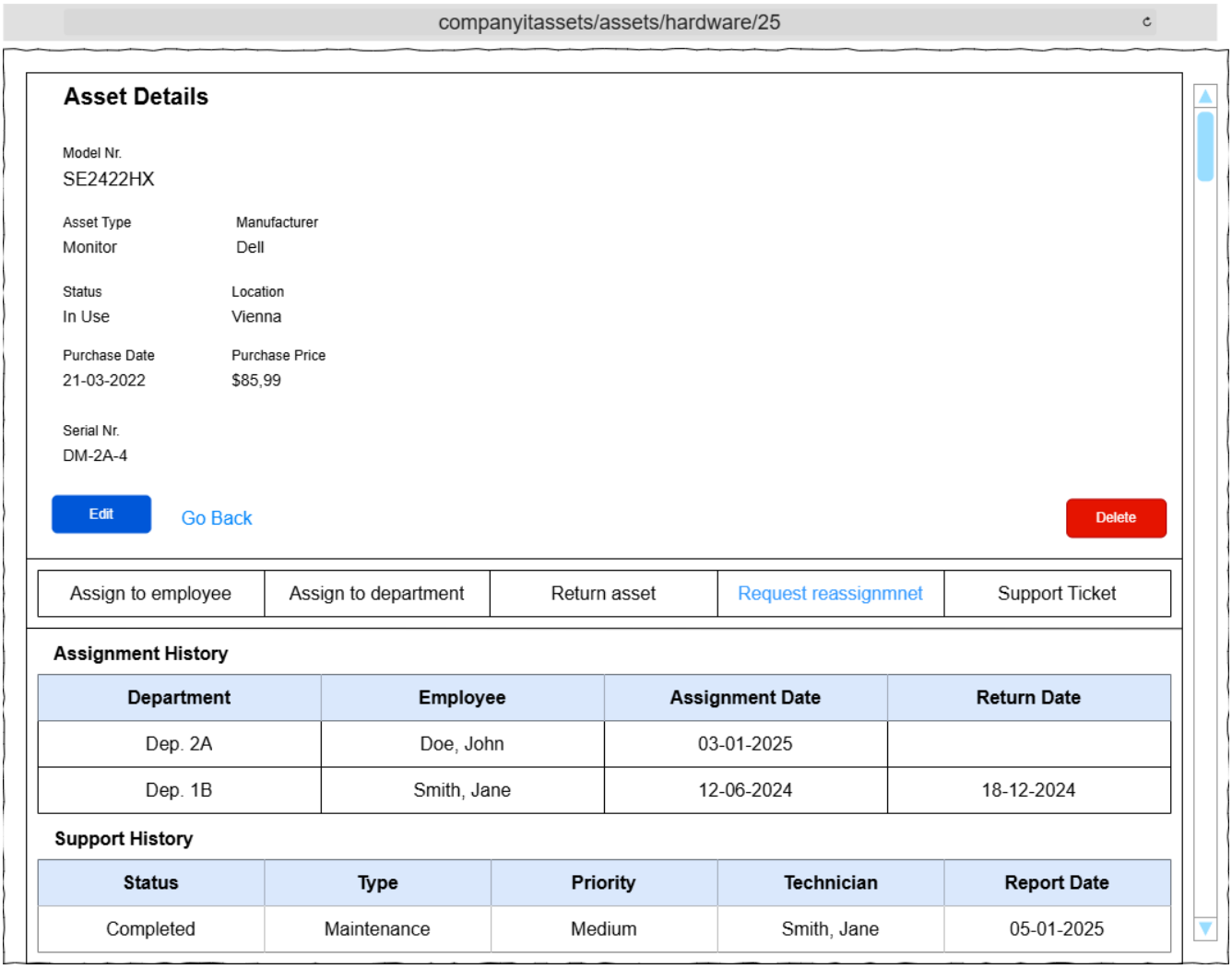
The ‘All Aseets’ link redirects to a Filter Page first, then to the Assets Index Page which is accessible to Asset Managers and Admins. Here all company assets are shown.





Clicking the ‘Add’ button will first redirect to a list of manufacturers to choose from and then to the ‘Add Asset’ form.





The Asset Details page shows various information of the selected asset including an assignment and support ticket history.

This page allows for various actions depending on the user's role

Employees and Help Desk Technicians: Can create a support ticket

Asset Manager and Admin: Access to all options displayed in the Wireframe

Software License Manager: Access to all options displayed in the Wireframe (for software assets)

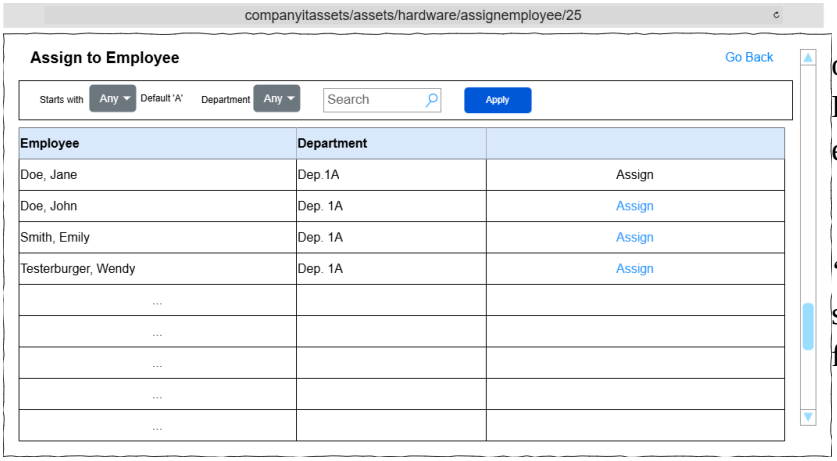
Department Manager: Can request an asset reassignment

The 'Edit' button leads to the Edit form which is similar to the 'Add Asset' form.

The 'Delete' button will redirect to a confirmation page before deleting the asset.

Assigning assets to employees or department can be done in two ways.

1. From the Asset Details Page:

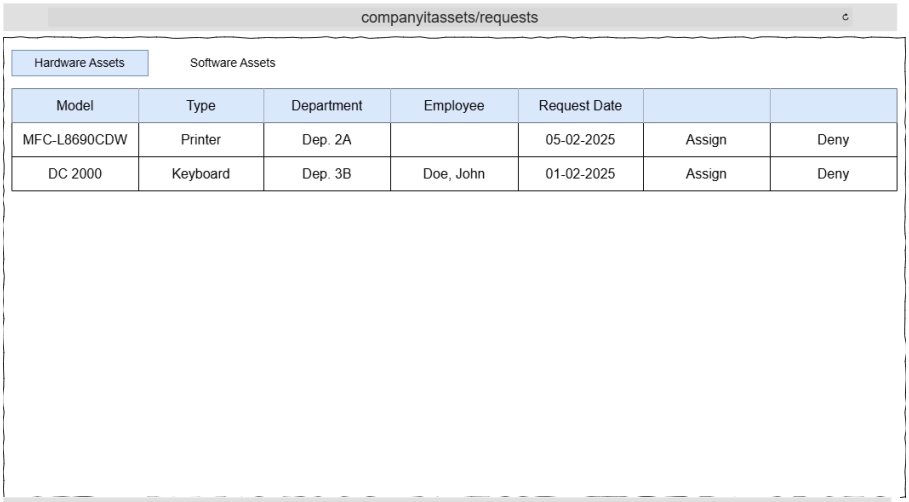


'Assign to Employee' will show a list of employees to choose from. The DepartmentID will be set to the chosen employees assigned department.

'Assign to Department' will instead show a list of Departments to choose from.

2. From an Asset Request

Asset Managers can access the header's *Requests* link.



If the request is instead denied the confirmation page will be almost identical with red coloration instead.

From the Asset Details page you can also create a **new support ticket**

companyitassets/tickets/add

New Support Ticket

Asset
KB-1A-03

Reported By
Doe, John

Issue Description

Type
- SELECT -

Priority
Medium

Save Go Back

Ticket Status and Resolution default to "Open".
Technical Employee is being assigned in the "Ticket Details" page.
Reporting Employee defaults to the logged in user

After hitting the save button, the user is being redirected to the details page of the new ticket, where reporting and assigned employee can be changed with proper authorization.

companyitassets/tickets/123

Ticket Details

Edit Go Back

Status	Resolution	Type	Priority	Report Date	Closed Date
Closed	Comleted	Issue	Medium	25-01-2025	03-02-2025

Asset Information

Asset Type	Model Nr.
Monitor	SE2422HX
Serial Nr.	**Location**
DM-3B-01	Vienna

Reported By
Doe, John Change

Assigned Technician
Jameson, James Re/Assign

Issue Description:

Monitor doesn't turn on anymore after Employee violently punched it with their fist because "I deleted a comment and suddenly the code wouldn't compile anymore".

Notes

Add Note

Note	Employee	Date	
Replacement parts arrived. Starting repair tomorrow.	Jameson, James	29-01-2025	Edit | Delete
Replacement parts ordered	Jameson, James	26-01-2025	Edit | Delete

The ‘Tickets’ link in the header redirects to the tickets index page.

companyitassets/tickets/

Order
Report Date Desc.
Status
Any
Type
Any
Priority
Any
Only Unassigned
Search
Apply

Tickets Overview

Add

Asset	Status	Type	Priority	Report D.	
DM-3B-01	Open	Issue	High	03-02-2025	Details
DM-3B-02	Open	Issue	High	03-02-2025	Details
MS-3B-02	In Progress	Maintenance	Medium	03-02-2025	Details
KB-3B-01	Open	Maintenance	Critical	02-02-2025	Details
KB-3B-01	In Progress	Request	Low	02-02-2025	Details
DM-4A-01	In Progress	Maintenance	Medium	01-02-2025	Details
DM-4A-02	Closed	Issue	Critical	01-02-2025	Details
KB-4A-01	Closed	Other	Low	01-02-2025	Details
KB-IT-08	Closed	Issue	High	31-01-2025	Details
KB-4A-01	Closed	Maintenance	Medium	31-01-2025	Details
MS-3B-01	Closed	Maintenance	Low	31-01-2025	Details

Tickets that are not assigned to a technician are displayed with a light red background.
Here clicking the “Add” button will simply redirect to the Assets Index Page where an asset can be searched for and a new ticket can be created via the asset’s details page.

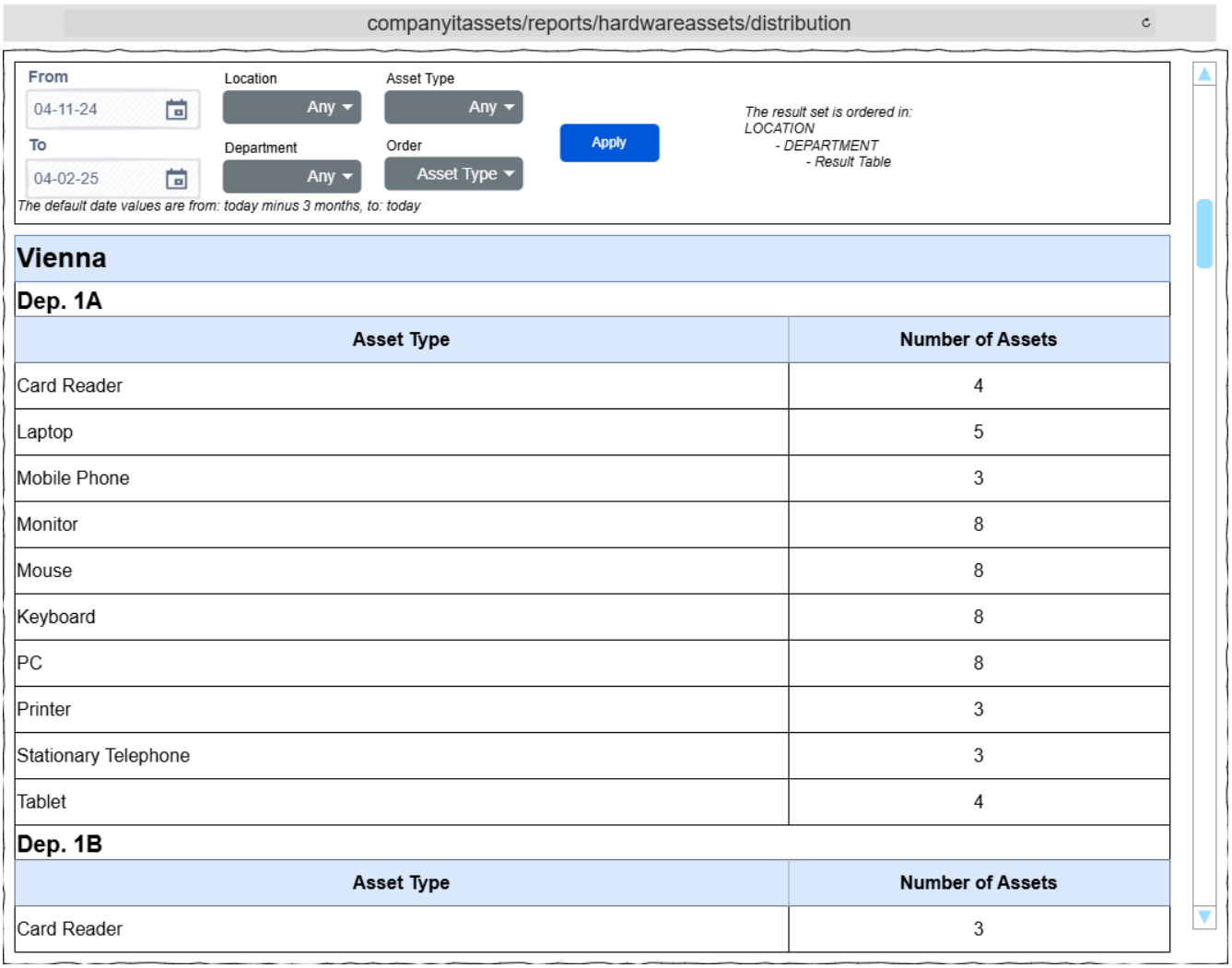
Changing the assigned technician or reporting employee redirects to employee select page which is similar to the employee assign page shown in the asset assignment section above.

Users with the *Department Manager* Role also have read-only access to the Tickets page. However only tickets for assets assigned to their department will be shown.

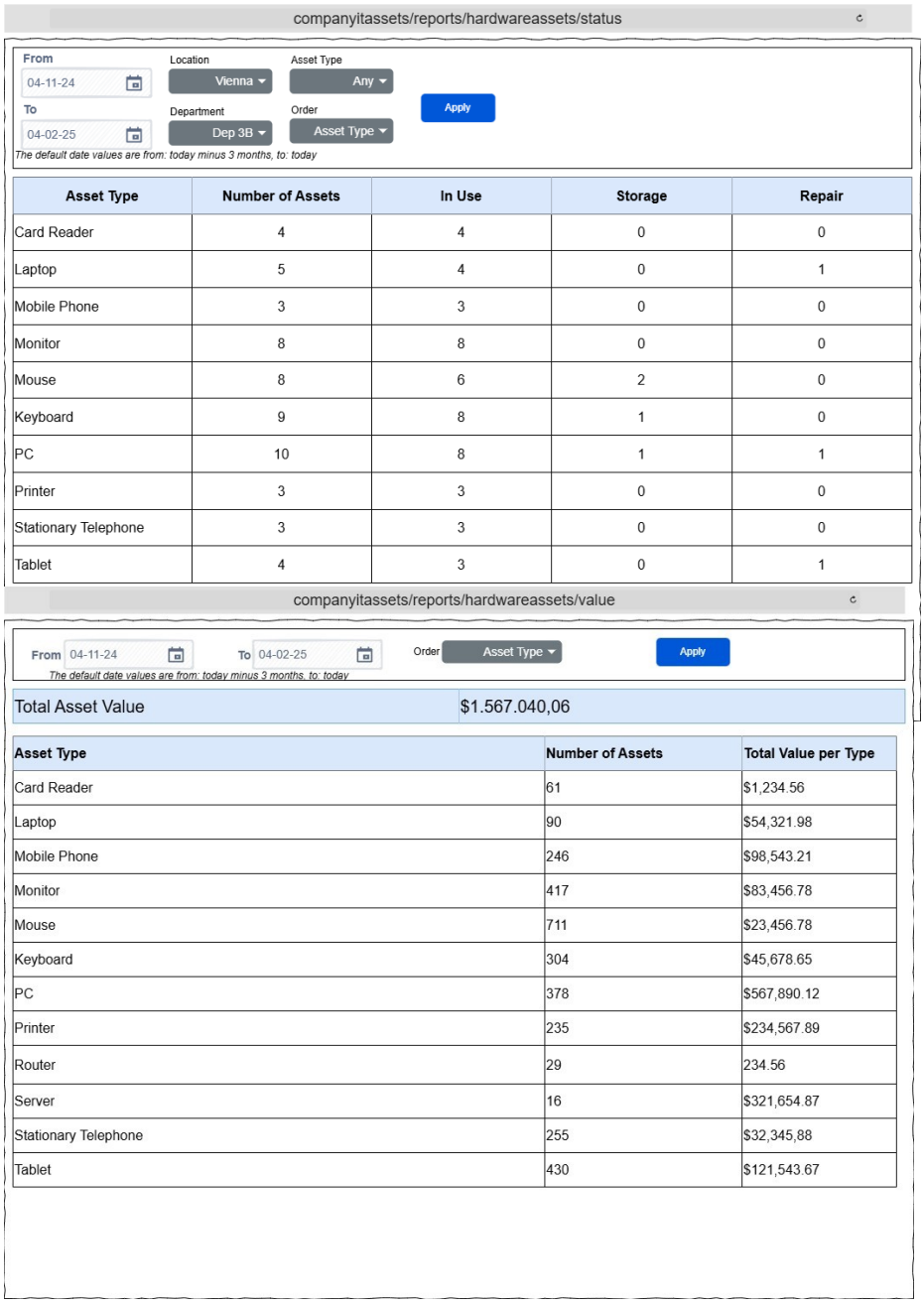
Lastly and Auditor can create various **reports** from ‘Reports’ link in the header.
Asset Managers can create Hardware Asset Reports and Software License Managers can create Software Compliance reports

The reports index page will feature a simple interface where the user has multiple options to choose from:

- Hardware Assets Reports
  - Distribution
  - Status
  - Value
- Software Assets Reports
  - Distribution
  - Status
- Tickets

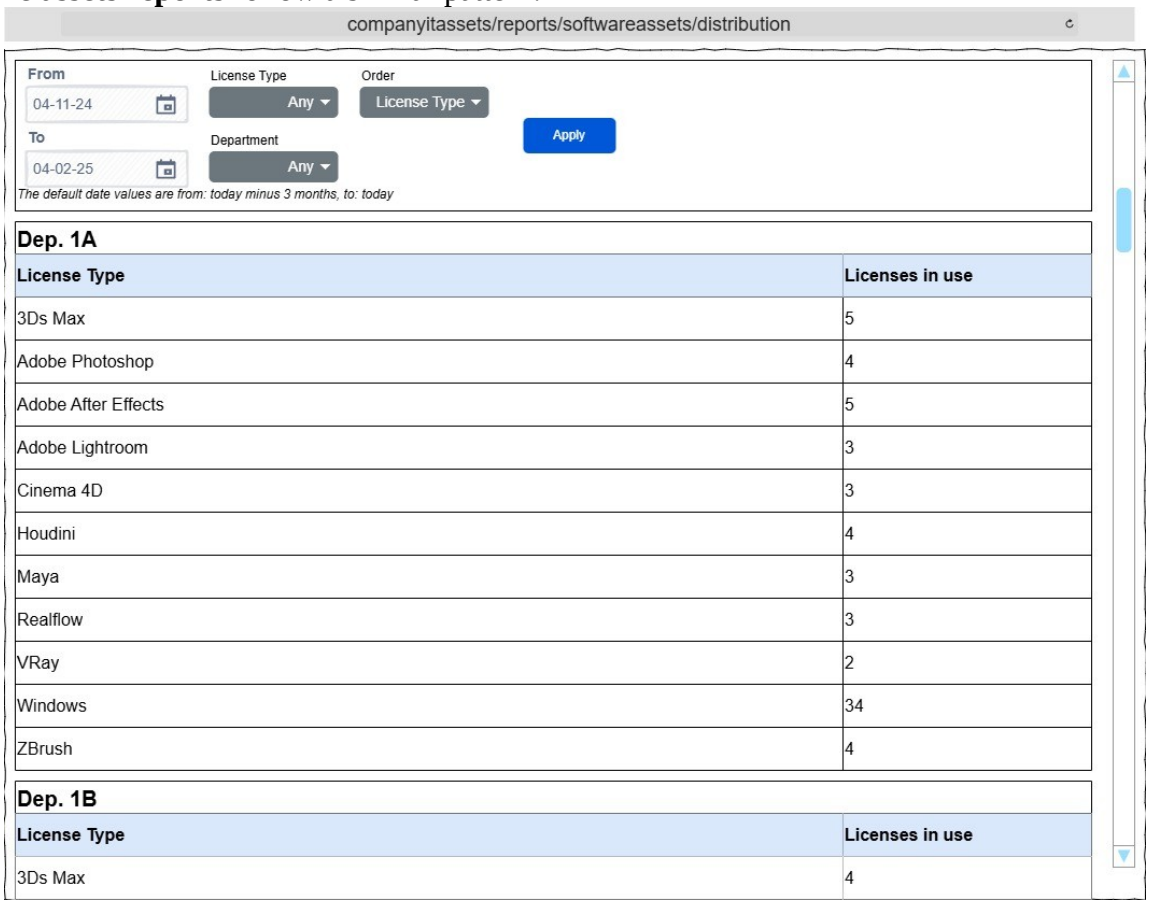


Example Hardware Distribution Reports



The Date Filter for all reports defaults to: From: Today – 3 Months To: Today

Software assets reports follow a similar pattern:



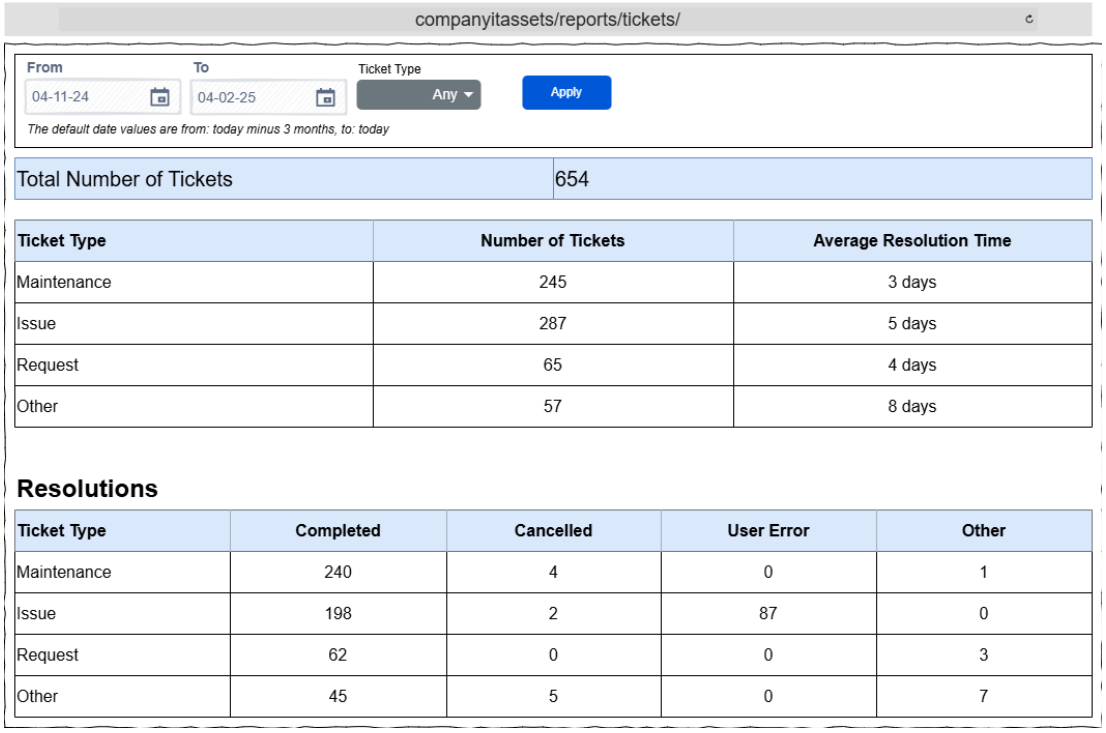
Dep. 1A

License Type	Licenses in use
3Ds Max	5
Adobe Photoshop	4
Adobe After Effects	5
Adobe Lightroom	3
Cinema 4D	3
Houdini	4
Maya	3
Realflow	3
VRay	2
Windows	34
ZBrush	4

Dep. 1B

License Type	Licenses in use
3Ds Max	4

For **Ticket reports** all information is in one page:



Total Number of Tickets	654

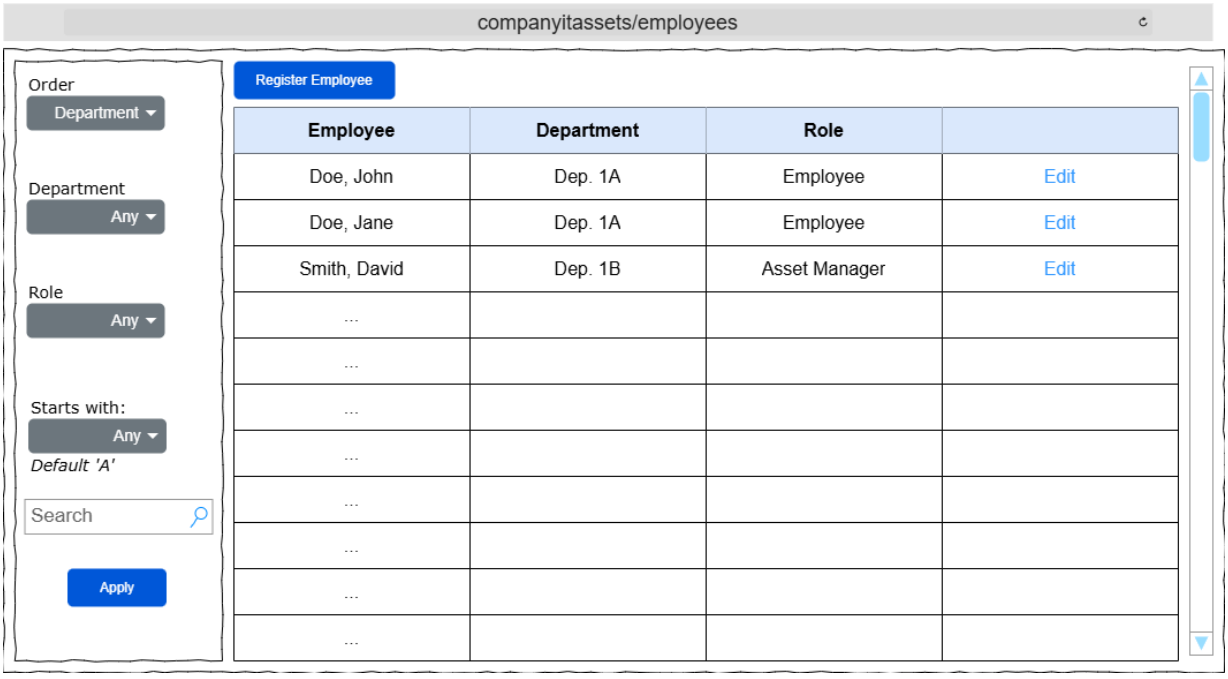
Ticket Type	Number of Tickets	Average Resolution Time
Maintenance	245	3 days
Issue	287	5 days
Request	65	4 days
Other	57	8 days

Resolutions

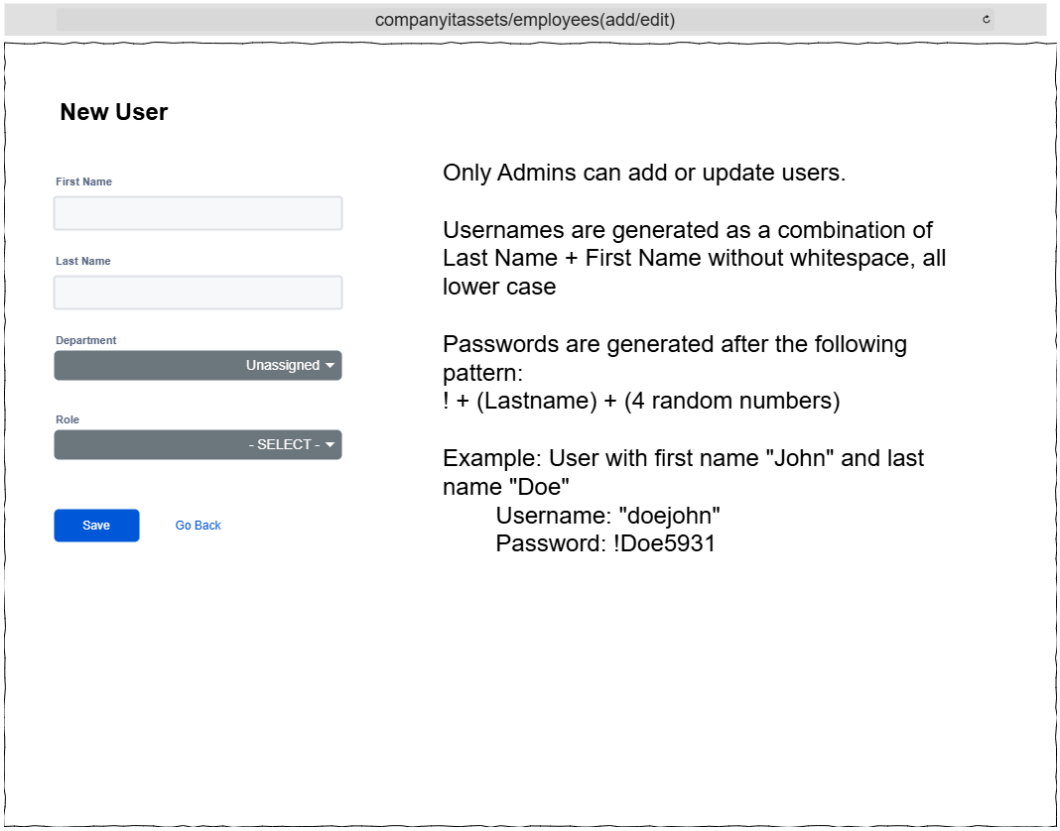
Ticket Type	Completed	Cancelled	User Error	Other
Maintenance	240	4	0	1
Issue	198	2	87	0
Request	62	0	0	3
Other	45	5	0	7

Admins can add or edit employee information and roles

The Employees Link in the header lead to the employees index page:



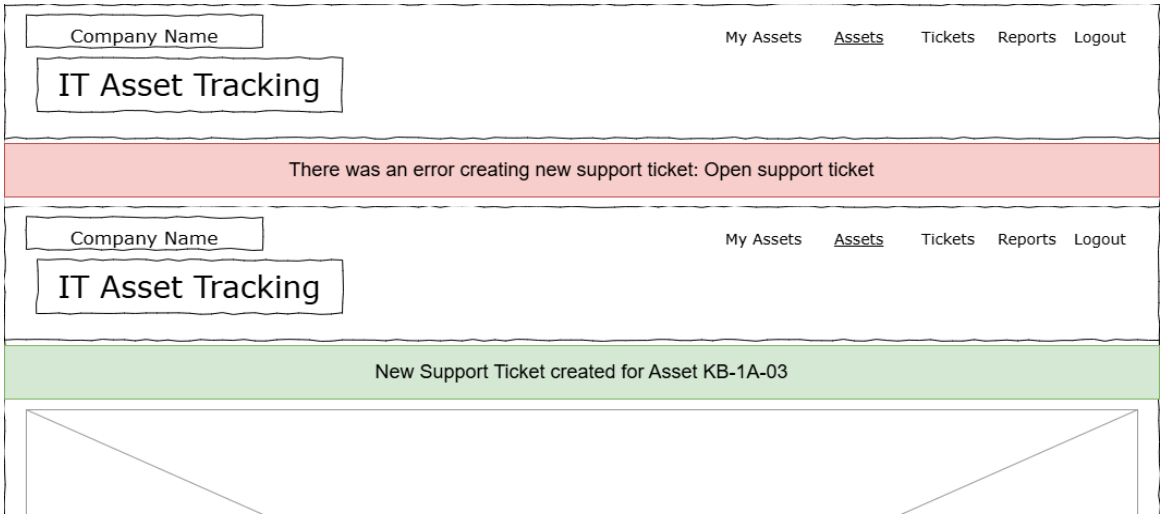
From here new Employees can be registered or existing ones edited



App Feedback and Error handling:

Error messages, as well as success or informational messages are displayed below the header.

Error messages are displayed with red background and white text while success or informational messages are displayed with green background and white text.



Error messages can be displayed for various reasons, like database connection errors, form invalidity, and database creating/updating/deleting errors.

Every success, information and error message is also being logged to the console.

Upon error the user is redirected to either the previous page, home page or form page where an error message, as described above will be displayed.

Rest API

CORS policies

If the environment is set to development the **AllowAllOrigins** policy is used. Which allows all origins as the name suggests.

In production, the app uses the **AllowedOrigins** policy which reads the allowed origins from the appsettings.json file.

Error Handling

Errors are being logged and controllers send an appropriate response of either 401 Unauthorized, 404 Not Found, 500 Internal Server Error etc.

The following pages contain a list of **endpoints**

API Endpoints

Assets Endpoints					
Endpoint	**Route**	**Method**	**Roles**	**Response**	**Request Body**
Get Assets	/assets	GET	• Admin • Asset Manager • Auditor	200(JSON), 401	
Get Department Assets	/assets/department/{departmentID}	GET	• Admin • Asset Manager • Auditor • Department Manager	200(JSON), 401	
Get Employee Assets	/assets/employee/{employeeID}	GET	• Admin • Asset Manager • Auditor • All other roles**	200(JSON), 401	
Get Single Asset	/assets/{assetID}	GET	• Admin • Asset Manager • Auditor • Department Manager* • All other roles**	200(JSON), 404, 401	
Get Available Assets	/assets/available	GET	• Admin • Asset Manager • Auditor • Department Manager	200(JSON), 401	

Add Asset	/assets	POST	• Admin
			• Asset Manager
				201, 400, 401, 409	{
					AssetTypeID: int
					ManufacturerID: int?***
					ModelID: int?
					LocationID : int
					SerialNumber: string (maxL: 50)
					PirchaseDate: date
					PurchasePrice: decimal(10,2)
					}

Edit Asset	/assets/{assetID}	PUT	• Admin
			• Asset Manager
				204, 400, 404, 401	{
					AssetTypeID: int
					ManufacturerID: int?
					ModelID: int?
					AssetStatusID: int
					LocationID : int
					SerialNumber: string (maxL: 50)
					PirchaseDate: date
					PurchasePrice: decimal(10,2)
					}

Delete Asset	/assets/{assetID}	DELETE	• Admin • Asset Manager	204, 404, 401	

* Only if authorized Department Manager is assigned the same DepartmentID as the requested Asset, ** Only if Asset is Assigned to EmployeeID of the requesting Employee, *** ? = optional

Get Licenses	/licenses	GET	• Admin • Software License Manager • Auditor	200(JSON), 401	
Get Employee Licenses	/licenses/employee/{employeeID}	GET	• Admin • Software License Manager • Auditor • All other roles**	200(JSON), 401, 404	
Get Single License	/licenses/{softwareAssetID}	GET	• Admin • Software License Manager • Auditor • Department Manager* • All other roles**	200(JSON), 404, 401	

Add License	/licenses	POST	• Admin
			• Software License Manager
				201, 400, 401	{
					ManufacturerID: int
					LicenseTypeID: int
					LicenseKey: string (maxL: 50)
					NumberOfLicenses: int
					Version: string (maxL: 10)
					ExpirationDate: date
					}

Edit License	/licenses/{softwareAssetID}	PUT	• Admin
			• Software License Manager
				204, 400, 404, 401	{
					ManufacturerID: int
					LicenseTypeID: int
					AssetStatusID: int
					LicenseKey: string (maxL: 50)
					NumberOfLicenses: int
					Version: string (maxL: 10)
					ExpirationDate: date
					}

Delete License	/licenses/{softwareAssetID}	DELETE	• Admin • Software License Manager	204, 404, 401	

* Only if authorized Department Manager is assigned the same DepartmentID as the Employee the requested Asset is assigned to,
** Only if Asset is Assigned to EmployeeID of the requesting Employee

Asset Assignment Endpoints					
Endpoint	**Route**	**Method**	**Roles**	**Response**	**Request Body**
Get assignemt history for one asset	/assets/{assetID}/assignments	GET	• Admin • Asset Manager • Auditor	200(JSON), 404, 401	

Assign Asset	/assets/assignments/	POST	• Admin
			• Asset Manager
				201, 400, 401, 409	{
					AssetID: int
					DepartmentID: int
					EmployeeID: int?
					}

Reassign Asset / Return Asset	/assets/assignments/ {assetAssignmentID}	PUT	• Admin
			• Asset Manager
				204, 400, 401	{
					AssetID: int
					DepartmentID: int
					EmployeeID: int?
					ReturnDate: date?
					}

Get assignemt history for one license	/licenses/{softwareAssetID}/ assignments	GET	• Admin • Software License Manager • Auditor	200(JSON), 404, 401	

Assign License	/licenses/assignments/	POST	• Admin
			• Software License Manager
				201, 400, 401, 409	{
					SoftwareAssetID
					AssetID: bigint?
					EmployeeID: int?
					}

Reassign License / Return License	/licenses/assignments/ {assetAssignmentID}	PUT	• Admin
			• Software License Manager
				204, 400, 401	{
					SoftwareAssetID: int
					AssetID: bigint?
					EmployeeID: int?
					ReturnDate: date?
					}

Asset Request Endpoints					
Get asset request history	/requests	GET	• Admin • Asset Manager • Auditor	200(JSON), 401	
Get open asset requests	/requests/open	GET	• Admin • Asset Manager • Auditor	200(JSON), 401	

Request asset assignment	/requests/assets	POST	• Admin
			• Department Manager
				201, 400, 401, 409	{
					AssetID: int
					EmployeeID: int?
					DepartmentID: int
					}

Request license assignment	/requests/licenses	POST	• Admin
			• Department Manager
				201, 400, 401, 409	{
					SoftwareAssetID: int
					EmployeeID: int?
					AssetID: int
					}

Update / Resolve asset request	/requests/assets/{AssetRequestID}	PUT	• Admin
			• Asset Manager
				204, 400, 401, 409	{
					AssetID: int
					EmployeeID: int?
					DepartmentID: int
					RequestResultID: int?
					RequestNote: string?
					}

Update / Resolve license request	/requests/licenses/ {SoftwareLicenseRequestID}	PUT	• Admin
			• Software License Manager
				204, 400, 401, 409	{
					SoftwareAssetID: int
					EmployeeID: int?
					AssetID: int
					RequestResultID: int?
					RequestNote: string?
					}

Support Tickets Endpoints					
Endpoint	**Route**	**Method**	**Roles**	**Response**	**Request Body**
Get Ticket History	/tickets(?page=1)	GET	• Admin • Help Desk Technician • Auditor	200 (JSON), 401	
Get unresolved Tickets	/tickets/open	GET	• Admin • Help Desk Technician • Auditor	200 (JSON), 401	
Get unassigned Tickets	/tickets/unassigned	GET	• Admin • Help Desk Technician • Auditor	200 (JSON), 401	
Get Ticket Details	/tickets/{TickedID}	GET	• Admin • Help Desk Technician • Auditor	200 (JSON), 401, 404	

Create Ticket	/tickets	POST	• Admin
			• Help Desk Technician
			• Auditor
			• Department Manager
			• Employee
				201, 400, 401, 409	{
					TicketTypeID: int
					TicketPriorityID: int
					AssetID: int
					ReportedByEmployeeID: int
					IssueDescription: string (maxL: 1000)
					}

Update Ticket	/tickets/{TicketID}	PUT	• Admin
			• Help Desk Technician
				204, 400, 401, 409	{
					TicketStatusID: int
					TicketTypeID: int
					TicketPriorityID: int
					TicketResolutionID: int?
					AssignedToEmployeeID: int
					IssueDescription: string (maxL: 1000)
					}

Delete Ticket	/tickets/{TicketID}	DELETE	• Admin	204, 401, 404	
Get Ticket Notes	/tickets/{TickedID}/notes	GET	• Admin • Help Desk Technician • Auditor	200 (JSON), 401, 404	

Add Ticket Note	/tickets/{TickedID}/notes	POST	• Admin
			• Help Desk Technician
				201, 400, 401, 404	{
					EmployeeID: int
					Note: string
					}

Edit Ticket Note	/tickets/{TickedID}/notes/{TicketNoteID}	PUT	• Admin
			• Help Desk Technician
				204, 400, 401, 404	{
					TicketID: int
					EmployeeID: int
					Note: string
					}

Delete Ticket Note	/tickets/{TickedID}/notes/{TicketNoteID}	DELETE	• Admin • Help Desk Technician	204, 401, 404	

Reports Endpoints					
Endpoint	**Route**	**Method**	**Roles**	**Response**	**Request Body**

Generate Asset Reports	/reports/assets	GET	• Admin
			• Auditor
			• Asset Manager
				200 (JSON), 400, 404, 401	{
					From: date?
					To: date?
					LocationID: int?
					AssetTypeID: int?
					DepartmentID: int?
					}
					as query params

Generate Software Reports	/reports/licenses	GET	• Admin
			• Auditor
			• Software License Manager
				200 (JSON), 400, 404, 401	{
					From: date?
					To: date?
					LicenseTypeID: int?
					DepartmentID: int?
					}
					as query params

Generate Ticket Reports	/reports/tickets	GET	• Admin
			• Auditor
				200 (JSON), 400, 404, 401	{
					From: date?
					To: date?
					TicketTypeID: int?
					}
					as query params

User Management Endpoints					
Get all Employees	/employees	GET	• Admin	200 (JSON), 401	
Get Employee	/employees/{EmployeeID}	GET	• Admin	200 (JSON), 401	
Search Employees	/employees?search="string"	GET	• Admin	200 (JSON), 401	

Create Employee	/employees	POST	• Admin	201, 400, 401	{
					FirstName: string
					LastName: string
					DepartmentID: int?
					Role: RoleID
					}

Update Employee	/employees/{EmployeeID}	PUT	• Admin	204, 400, 404, 401	{
					FirstName: string
					LastName: string
					DepartmentID: int?
					Role: RoleID
					}

Authentication Endpoint					

Login	/login	POST		200 (bearer token), 401	{
					Username: string
					Password: string
					}

Other Endpoint					
Get Locations	/locations	GET	• Admin • Department Manager • Asset Manager • Software License Manager • Auditor • Help Desk Technician	200 (JSON), 401	
Get Departments	/departments	GET	• Admin • Department Manager • Asset Manager • Software License Manager • Auditor • Help Desk Technician	200 (JSON), 401	

Security

The MVC and API will leverage **ASP Identity Entity Framework** to manage Accounts and Roles.

The Password requirements are:

- 8 characters minimal length

  - Passwords are generated => ! + Lastname + 4RandomNumbers (!Doe5278)

  - if the user's last name is less than three characters long the generated password will substitute with additional random numbers at the end.

- at least one upper case character
- at least one lower case character
- at least one digit or other symbol

As described in the MVC users are assigned one of the following roles

- **Admin:** Access to all sites content and functionalities, can add or update users
- **Asset Manager:** Can add, edit and delete assets, assign assets, generate asset reports.
- **Software License Manager:** Manages software assets and their assignment, generate software compliance reports
- **Help Desk Technician:** Manage support tickets. Creating, updating, assigning and adding notes to tickets.
- **Department Manager:** View assets assigned to their department, request new assets or reassignments, view ticket status for their department's assets
- **Employee:** View and create support tickets for their assigned assets.
- **Auditor:** Read-Only access to all asset and ticket information, generate and view any reports.

Only User with the Admin Role may assign roles to other users.

See MVC Layer and API Endpoints section for role restricted access.

Logging

The MVC and API, by default, log information level events and above to the console.

I use **Serilog** for logging with Console and MSSqlServer Sink

- Information Logs on Database changes like creating, updating or deleting data.
- Warning Logs for invalid or unauthorized http requests.
- Error Logs for caught exceptions.

The applications write logs to the console and optionally to the database.

Database Logging is configurable in the appsettings.json of each program within the “DbLogging” section:

If “Enabled” is set to “true” the application writes events of the configured log level and above to the database.

The “LogLevel” can be set to:

- “Information”
- “Debug”
- “Warning”
- “Error”

The MVC and API write Logs to the “LogEvents” Table.