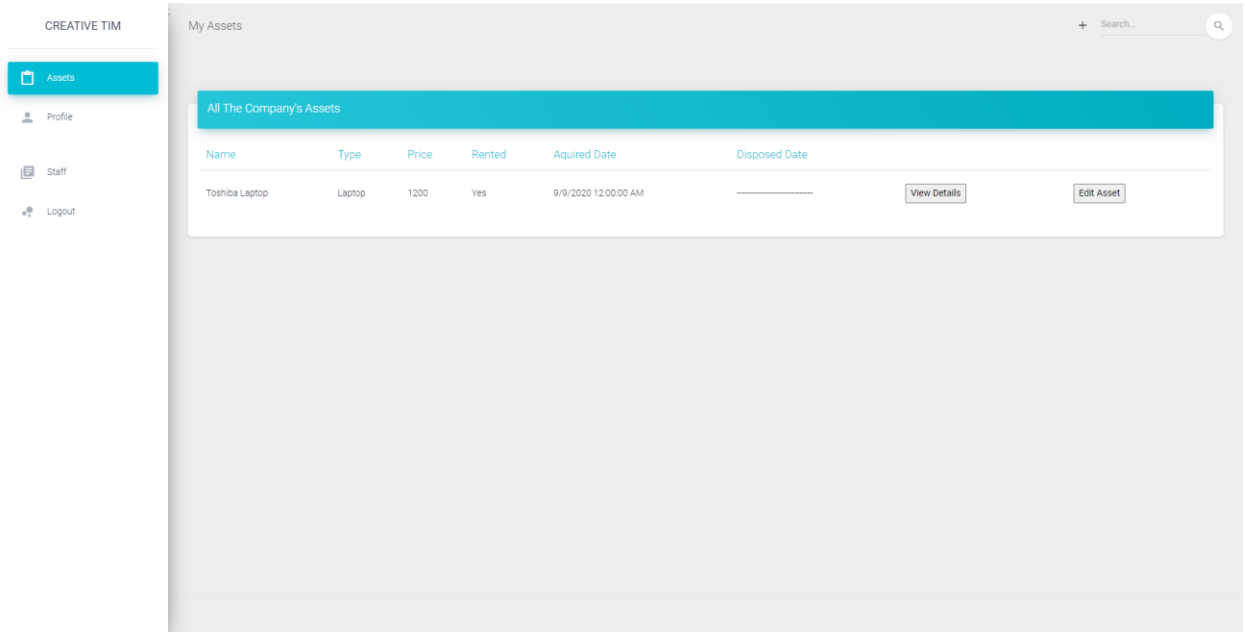
Asset Management

1- Project Conception and Features

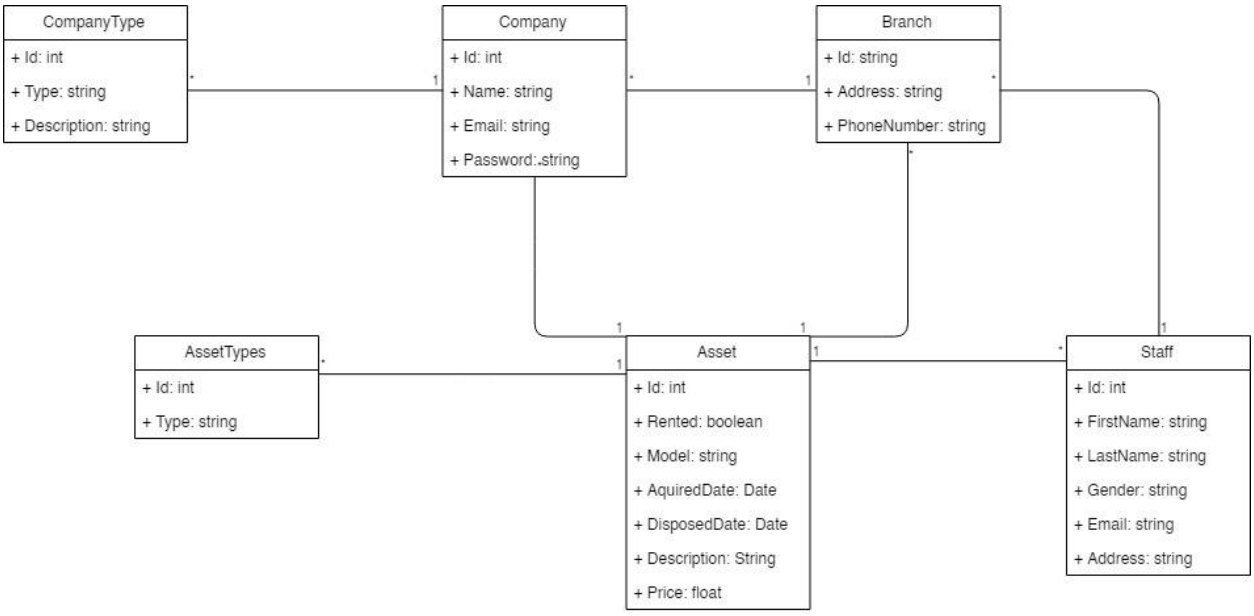
This project consists of a web application for managing a company's assets. The user (company) may create new assets, edit them (if any of its info changes), and assign an employee (or staff) to be responsible for that asset (like the case of personal computers).
Furthermore, it is possible to dispose of a given asset. A company can hire new staff, edit their info (in case of any change), and assign them to a specified branch of the company.



Contents

2- Database Conception:

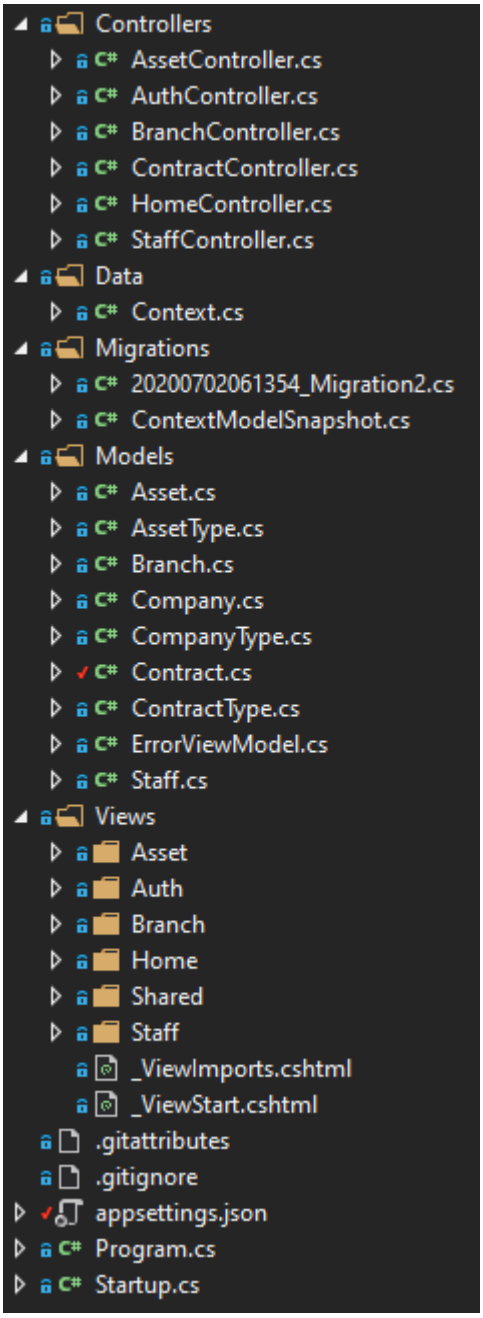
The UML Diagram (representing also the database) is as follows:



- A company has zero (if online only) or many branches, a single type and many assets.
- A branch has many staff, and belongs to only one company.
- There can be many companies of the same type.
- For the assets, each asset belongs to one company only, may or may not belong to a branch (like Stocks the company bought), has only one responsible (staff), and of course has only one type.
- There can be many assets of the same type.

3- Program Files:

The program files are organized as follows:



3.1- Controllers:

- AuthController: responsible for registering, logging in, logging out, and showing the company profile, all using Microsoft's Identity.
- AssetController: Contains CRUD requests handlers for everything related to assets.
- Staff/Branch controller: responsible for adding and editing staff/branches.

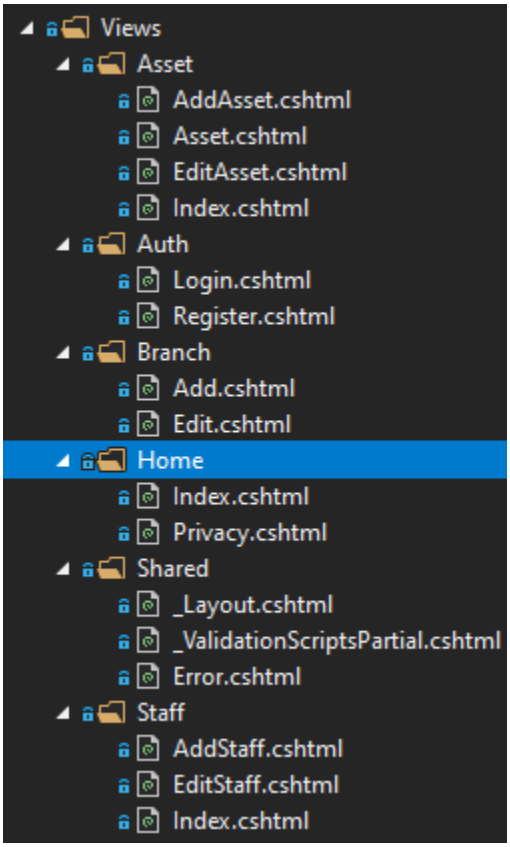
- ContractController: will be used for the final version of the project.

3.2- Models/Context:

- These models are conforming to the UML diagram above, and named in a way that makes them explain themselves. They are used with entity framework integration (Microsoft's ORM), to communicate in an efficient way (asynchronous) with the database.

3.3-Views:

The views are organized as follows



- Our views are organized in folders relatively to the controller that calls them ("ControllerName/ActionMethon") to make easier redirections and returning views without more code.
- Plus, the name of each view is self-explanatory.

4- Note:

More features will be added for the final version of the project (with files transfer like images, a better code architecture with custom services and dependency injection).

Carlita Khawand (55028)
Yorgo Wakim (54689)