TagItSmart
Smart tags driven service platform for enabling ecosystems of connected objects

Grant agreement 688061

DuraTag
D2 Development and integration report

Deliverable ID:	**D2**
Deliverable Title:	**Development and integration report**
Project acronym	**TagItSmart!**
Project title:	**Smart Tags driven service platform for enabling ecosystems of connected objects**
Extension acronym:	**DuraTag**
Extension title:	**Durability Tag Pilot for Work Conditions Assessment of Heavy Duty Mechanical Parts in Machinery Industry**
Contractual date of delivery:	**01.01.2019**

Contents

Introduction ***3***

Section 1 - Architecture ***4***

1.1 FC-Scanner v2 ..... **4**

1.2 TagItSmart SmartTags **4**

1.3 Web Service **5**

1.4 Web Application for product passport management **5**

1.5 Web Application for marketplace **5**

Section 2 - DuraTag platform ***6***

2.1 API **6**

Authentication 6

API controllers 6

Api Data Layer 8

2.2 Dashboard **9**

Stakeholders 9

Companies 14

Products 19

QR codes 22

Logs 23

2.3 Mobile Application **24**

Login Page 25

Unscanned Products 25

Scanning Page 26

Section 3 - Conclusions ***28***

Introduction

In this document, the developed platform is presented. Developed components are supporting product passport by provisioning FC-scanner module (logs the temperature and the usage time), mobile application as an enabler for scanning the smart tags and web application with different user roles for stakeholders, companies and management of products' and users' data. The goal was to facilitate reusing machine parts in waste processing industry over the platform for data exchange (i.e. Electric motors, Geared motors, Bearings) by providing information on composition materials thus making a contribution to the Circular Economy.

From the existing TagItSmart architecture SmartTags and FC-scanner SDK are being used while the rest of functional components, Encoding and Printing module, were developed from scratch. Product passport is developed based on specified user stories and its features for managing machine data (add, modify, delete) and data sharing. D1 document provides these user stories and was driving the development activities, based on interviews done with stakeholder (BankPro) to extract data that are required for each machine part.

Section 1 - Architecture

The developed platform consist of native TagItSmart components adapted for the DuraTag needs and additional components developed as modular functional blocks with APIs. The DuraTag is implemented as a multiple module application:

1. FC-scanner (adapter TagItSmart component)
2. SmartTags (native TagItSmart component)
3. Web Service with API controllers
4. Web Application for product passport management
5. Web Application for marketplace

Developed and adapted TagItSmart technology including FC-scanner camera module, encoding and printing SDK, and specific product passport modules.

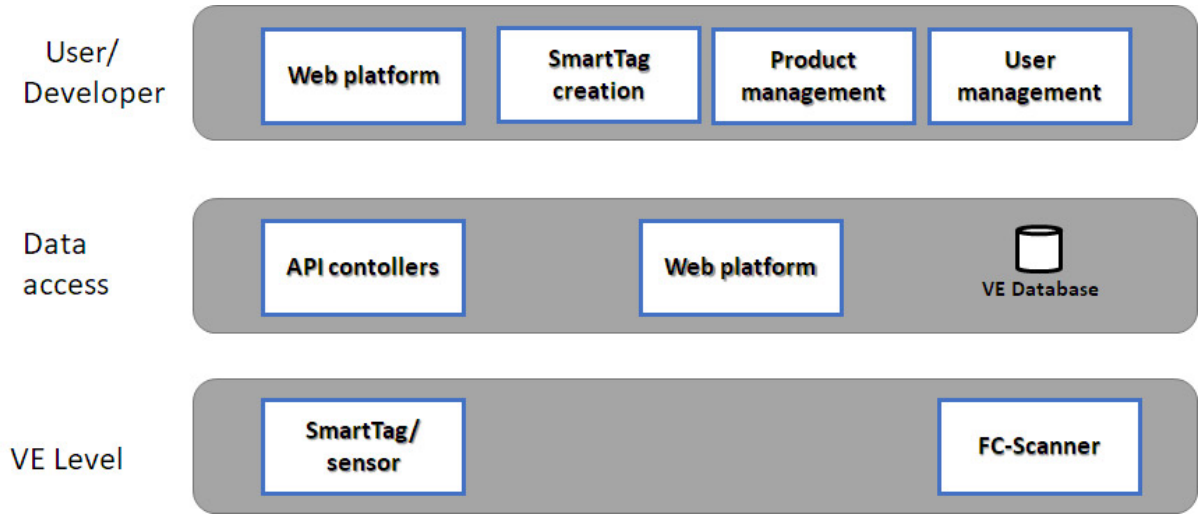


Figure 1 DuraTag extension of the TagItSmart architecture

1.1 FC-Scanner v2

Modified FC-scanner module was developed to support the scanning of the SmartTags. The usability was strictly dependent on the size of the tags, as the size of the machine components was limiting the label surface and accordingly the distance of the camera and was increasing the number of failed scannes. For this reason we did not used camera placed in front of each machine part as installation was not user friendly. Mobile application was developed using Xamarin – Microsoft C# based platform for developing cross-platform mobile applications. Application is communicating with Web Service trough RESTfull API, and also with TagItSmart API trough SDK.

1.2 TagItSmart SmartTags

Before taking into account requirements for the SmartTags, we have considered the reversible ink availability for high temperatures. The highest temperature for reversible inks is 65°C as reversible pigments do not survive high temperatures. However, irreversible inks are available for higher temperatures. The design of the tag is presented in figure below.

Figure 2 DuraTag SmartTag design

1.3 Web Service

Web Service is based on .NET Core 2.0 and is hosted on Azure in Web App resource. .NET Core 2.0 is powerful web service and web application framework created by Microsoft. Web Service uses MSSQL relational database, which is hosted on Azure in SQL Database resource. Web Service expose RESTfull API for both mobile applications and web applications usage. Web Service communicates with TagItSmart trough exposed API. The REST design follows to enable **TagItSmart Service Manager, TagItSmart repository** and **TagItSmart Encoding and Printing** modules integrated into the service.

1.4 Web Application for product passport management

Web Application will provide the user (producer) with ability to add and administrate data about machines and all of their components to the platform. Web Application will be developed as single page web application using AngularJS, a JavaScript-based open-source frontend web application framework used for developing single-page applications. Web Application will communicate with Web Service via RESTfull API. Producer can add machines and components using this application.

1.5 Web Application for marketplace

This component enables registration of external 3rd parties over the website, specific queries (over the APIs as well as UI) based on the developed ontology to provide different information exchange such as: Which companies produce a certain product composed of certain recycled material; which companies generate outputs of certain material that can be used as input by company for repairing of certain product; which products are in good state (condition) and are available for reuse (post-use activity) from a specific stakeholder; etc.

Section 2 - DuraTag platform

In the first 5 months of the project implementation, the DuraTag platform (http://duratag-dashboard.azurewebsites.net) was developed. The development included DuraTag Dashboard following user stories specified in D1 deliverable, containing management enablers for Stakeholders, Companies, Products, QR Codes as well as Product Passport Logs. In this section platform is depicted in form of API description with actual platform screenviews.

2.1 API

API is built on latest .NET core framework, using WebApi technology, as well as MSSQL database with Entity Framework core as ORM support library.

Authentication

Authentication is token based authentication, where API, as trusted source, is issuing JWT token based on security claims and generating it into string. Later on, every authorized API call needs to have Authorization header with value Bearer + token.
Authentication is also role based, meaning that JWT token that was issued by API has, besides other security claims, role claim of logged in user.

API controllers

Namespace: namespace DuraTag.Api.Controllers
There are 5 main WebApi controllers that serve as entry points to API:

1. Auth Controller
2. Companies Controller
3. Stakeholders Controller
4. Products Controller
5. QRCodes Controller

Auth Controller

AuthController is used for authenticating users with issuing JWT tokens.
Url: api/auth
Methods:

- POST login – authenticating user with username/password provided and issuing JWT token with claims of that authenticated user.

Companies Controller

CompaniesController is used for managing (CRUD) companies entities and user connected to companies (operators and administrators). This controller requires authentication.
Url: api/companies
Methods:

- POST companies – adding new company to database
- GET companies – getting all companies from the database
- GET companies/:id – getting specific company from the database
- PUT companies/:id – editing specific company in database
- DELETE companies/:id – deleting specific company from database
- POST companies/admins – adding company admin user to database
- GET companies/admins – getting all company admin users from database

- PUT companies/admin/:id – editing specific company admin from database
- DELETE companies/admin/:id – deleting specific company admin from database
- POST companies/operators – adding company operator user to database
- GET companies/operators – getting all company operator users from database
- PUT companies/operator/:id – editing specific company oprerator from database
- DELETE companies/operator/:id – deleting specific company admin from database

Stakeholders Controller

StakeholdersController is used for managing (CRUD) stakeholders entities and user connected to stakeholders (operators and administrators). This controller requires authentication.

Url: api/stakeholders

Methods:

- POST stakeholders – adding new stakeholder to database
- GET stakeholders – getting all stakeholders from the database
- GET stakeholders/:id – getting specific stakeholder from the database
- PUT stakeholders/:id – editing specific stakeholder in database
- DELETE stakeholders/:id – deleting specific stakeholder from database
- POST stakeholders/admins – adding stakeholder admin user to database
- GET stakeholders/admins – getting all stakeholder admin users from database
- PUT stakeholders/admin/:id – editing specific stakeholder admin from database
- DELETE stakeholders/admin/:id – deleting specific stakeholder admin from database
- POST stakeholders/operators – adding stakeholder operator user to database
- GET stakeholders/operators – getting all stakeholder operator users from database
- PUT stakeholders/operator/:id – editing specific stakeholder operator from database
- DELETE stakeholders/operator/:id – deleting specific stakeholder operator from database

Products Controller

ProductsController is used for managing (CRUD) product passports as well as recording logs related to product passports and registering (connecting) new products with QR codes. This controller requires authentication.

Url: api/products

Methods:

- GET products – getting all products from database
- GET products/bearings – getting all products with type bearing from database
- GET products/cylinders – getting all products with type cylinder from database
- GET products/logs – getting all product logs
- GET products/:id/logs – getting product logs that are related to product provided
- GET products/logs/:id – getting specific log
- GET products/unscanned – getting products that are added to database but are still not connected with QR code
- POST products/bearings – adding product with bearing type to the database
- POST products/cylinders – adding product with cylinder type to the database
- POST products/:id/unlink – unlinking the product from current linked qr code
- POST products/image – uploading image for selected product
- PUT products/bearings/:id– updates selected bearing
- DELETE products/bearing/:id– deletes selected bearing

- PUT products/cylinders/:id– updates selected motor
- DELETE products/cylinders/:id– deletes selected motor

QRCodes Controller

QRCodesController is used for generating batch of QRCodes. This controller requires authentication.
Url: api/qrcodes
Methods:

- POST qrcodes – used for generating batch of QRcodes. After generation is done, zipped batch of qrcodes are downloaded to user's system.

Api Data Layer

Api data layer is composed of several parts. Entity Framework Core is ORM used for persisting entities to database.

Models

Namespace: DuraTag.Api.DataLayer.Models
In this namespace, all the system entity models are described, as well as Application context (used by Entity Framework to map the models to DB entities)
Expanding Products table for more detailed info about the products.

Repository pattern

Repository pattern is used in order to separate concern for all queries that are executed to DB with Entity Framework with well known unit of work implementation.
It consists of 2 parts:

1. Repositories
2. Data managers

Repositories

Namespace: DuraTag.Api.DataLayer.Repositories
This namespace holds all repository interfaces with all regular CRUD methods and additional methods described. Also, this namespace holds IUnitOfWork (interface for unit of work implementation).

Data Managers

Namespace: DuraTag.Api.DataLayer.DataManager
This namespace holds all data managers which are implementations for appropriate repository interfaces. Also, this namespace holds UnitOfWork that is implementation of IUnitOfWork interface, which unifies all data managers.

2.2 Dashboard

Dashboard is developed with Angular 6+ Framework, coded in TypeScript.

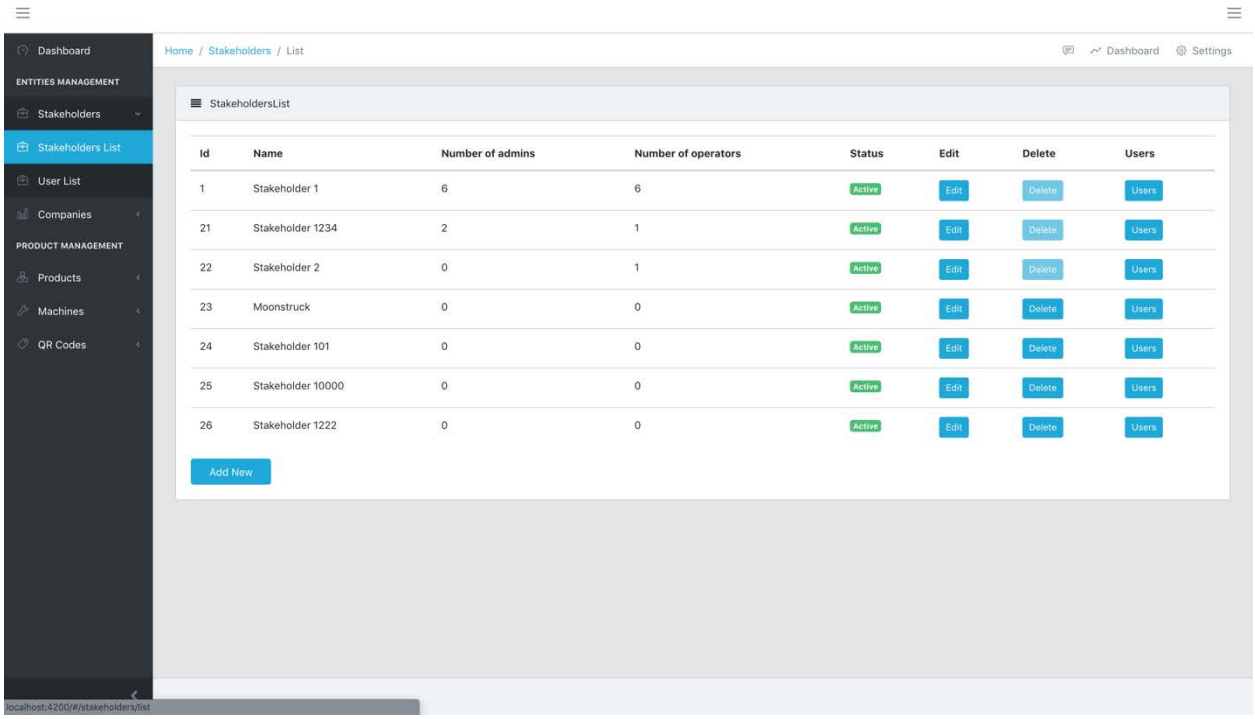
Stakeholders

Stakeholders modules is composed of 2 parts:

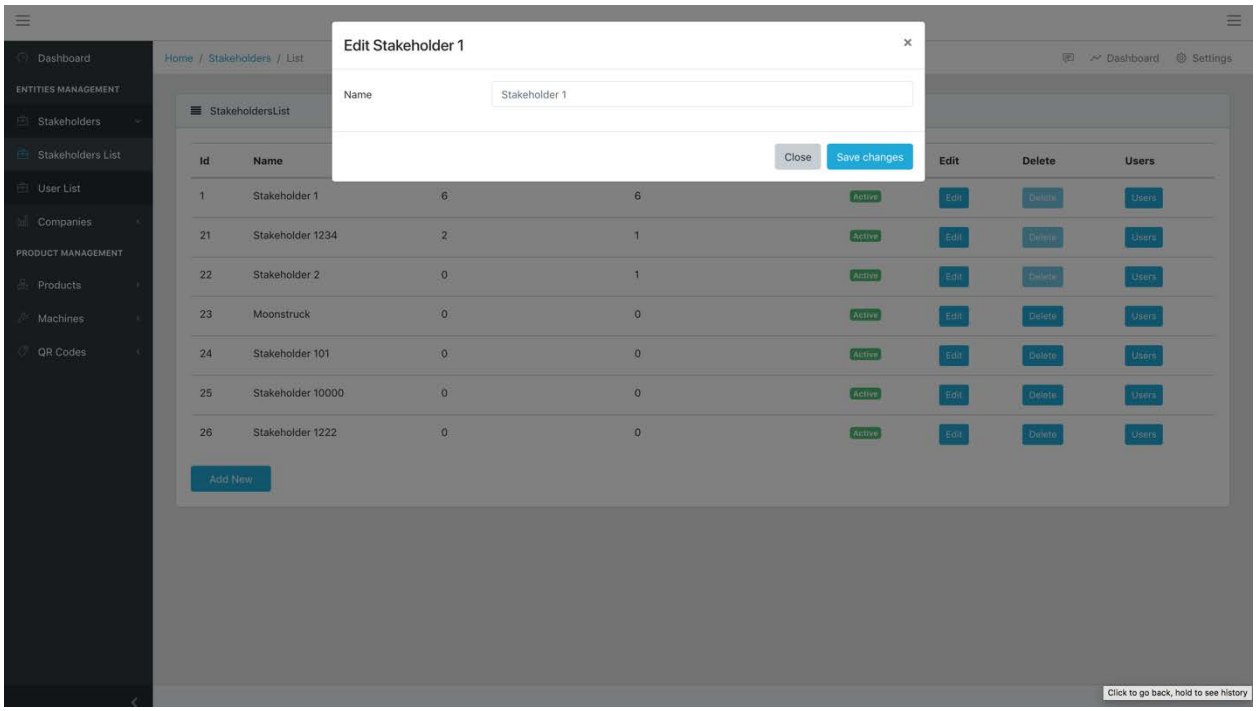
1. Stakeholders List
2. Users List

Stakeholders List

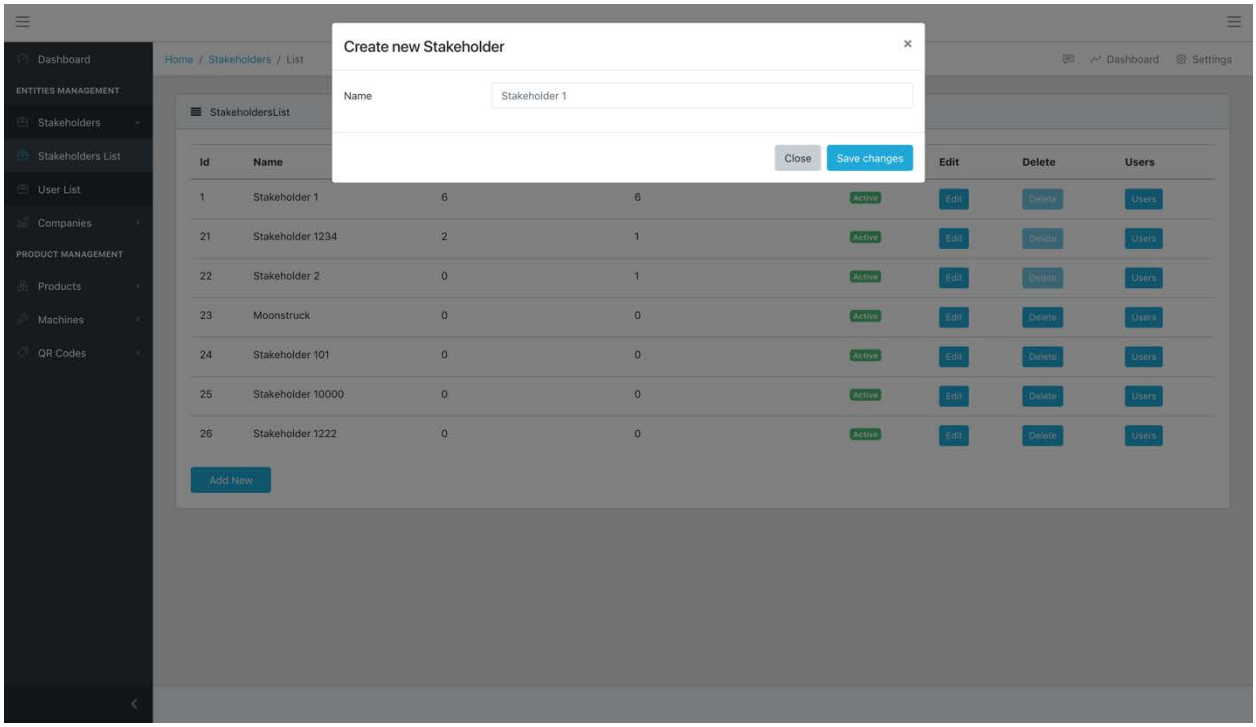
Stakeholders List module is used for displaying and managing all stakeholders in system.



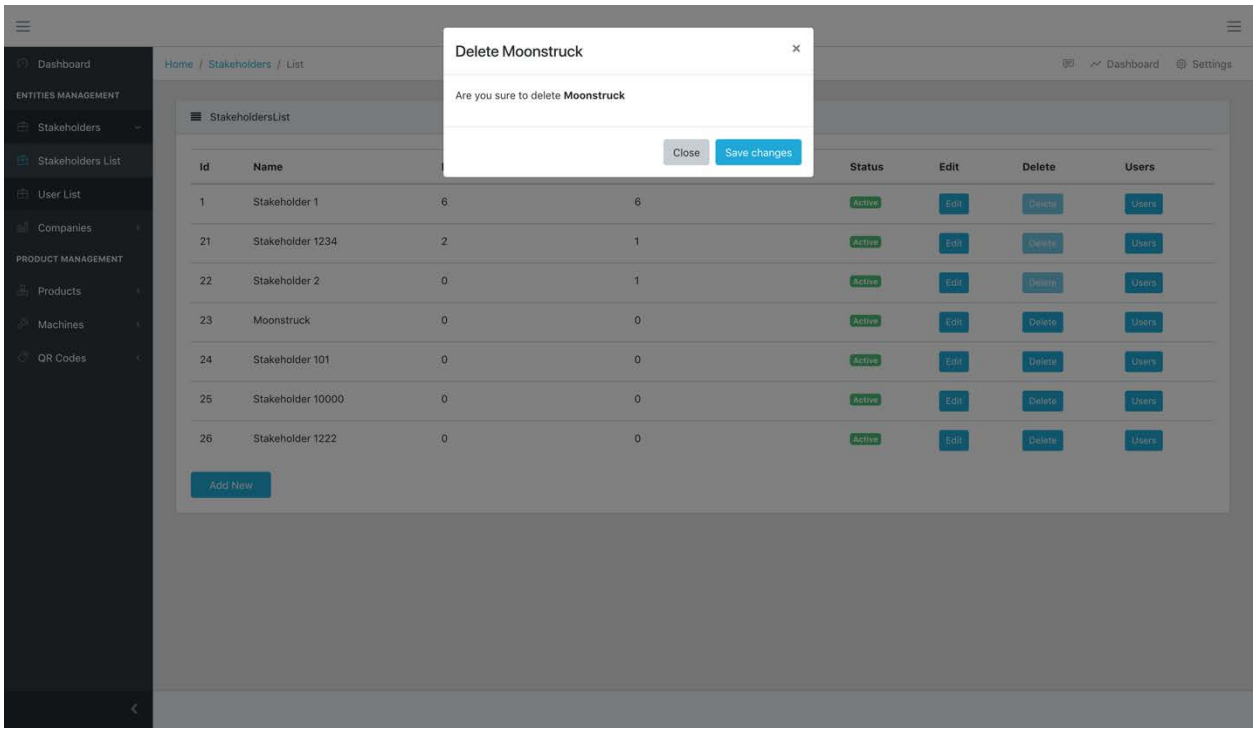
Edit


Edit Stakeholder 1
Name
Stakeholder 1
Close
Save changes
Dashboard
ENTITIES MANAGEMENT
Stakeholders
Stakeholders List
User List
Companies
PRODUCT MANAGEMENT
Products
Machines
QR Codes
Home / Stakeholders / List
StakeholdersList
Id
Name
Edit
Delete
Users
1 Stakeholder 1
21 Stakeholder 1234
22 Stakeholder 2
23 Moonstruck
24 Stakeholder 101
25 Stakeholder 10000
26 Stakeholder 1222
Add New

Add new


Create new Stakeholder
Name
Stakeholder 1
Close
Save changes
Dashboard
ENTITIES MANAGEMENT
Stakeholders
Stakeholders List
User List
Companies
PRODUCT MANAGEMENT
Products
Machines
QR Codes
Home / Stakeholders / List
StakeholdersList
Id
Name
Edit
Delete
Users
1 Stakeholder 1
21 Stakeholder 1234
22 Stakeholder 2
23 Moonstruck
24 Stakeholder 101
25 Stakeholder 10000
26 Stakeholder 1222
Add New

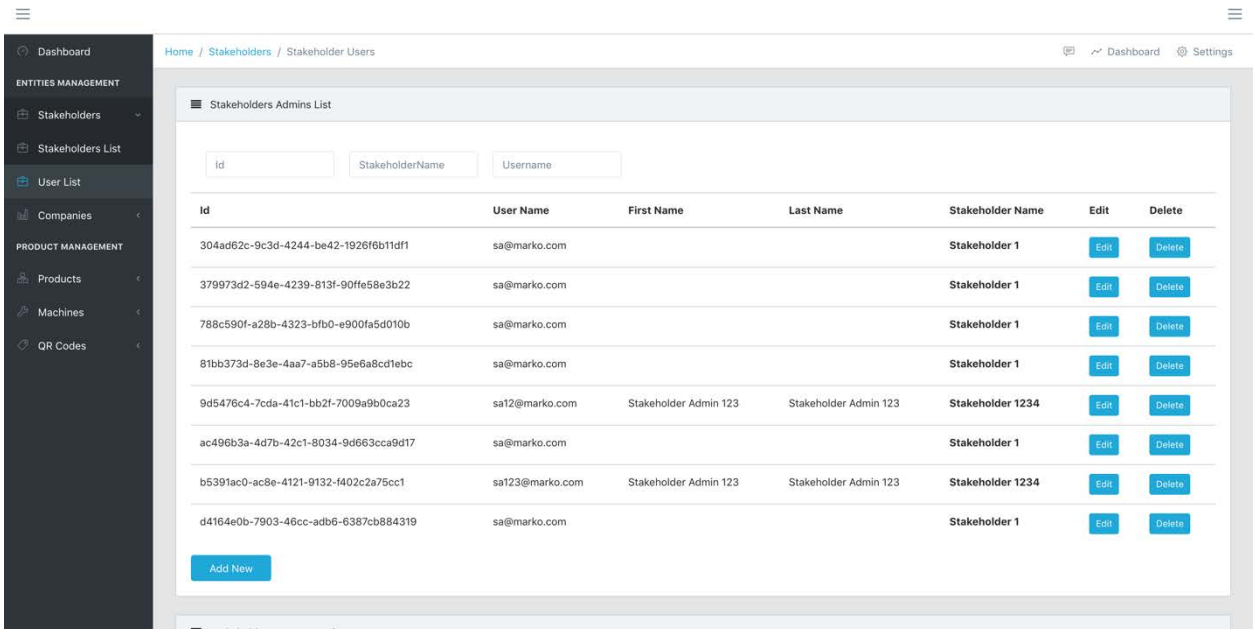
Delete



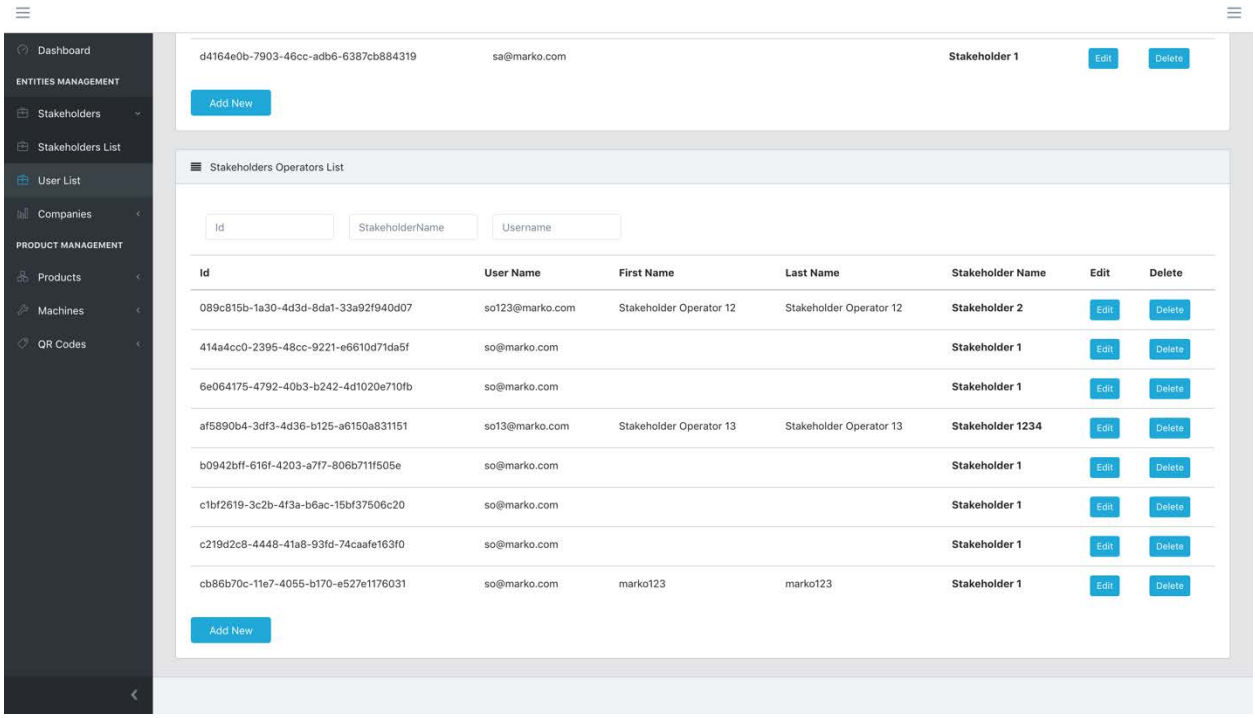
Stakeholders Users List

Users List module is used for displaying and managing all stakeholder users in system.

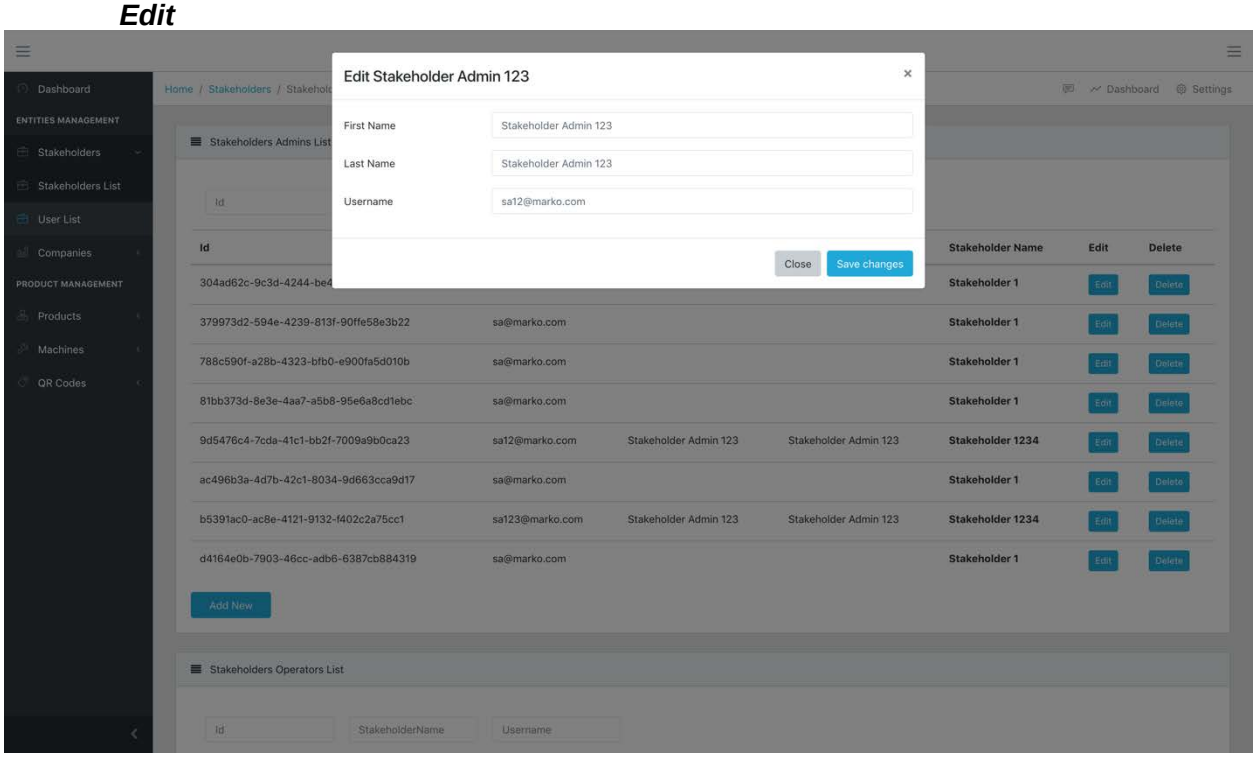
List Stakeholder Administrators



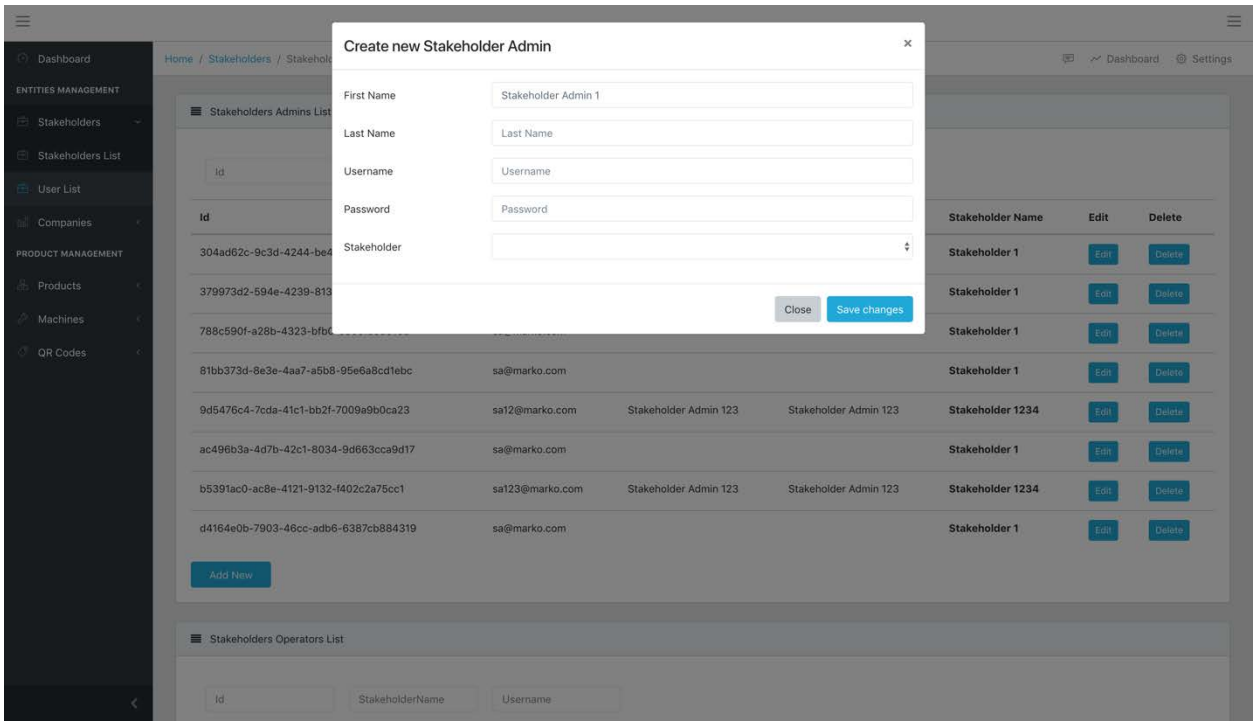
Stakeholder Operators



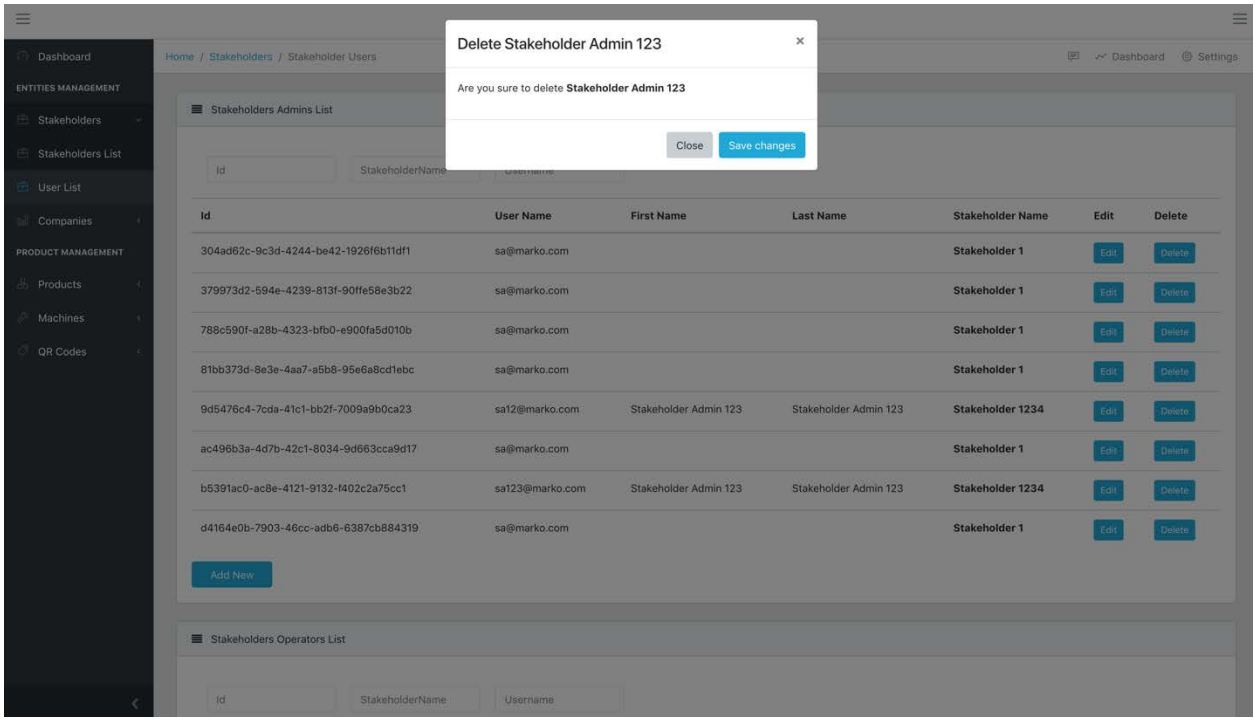
Edit



Add new



Delete



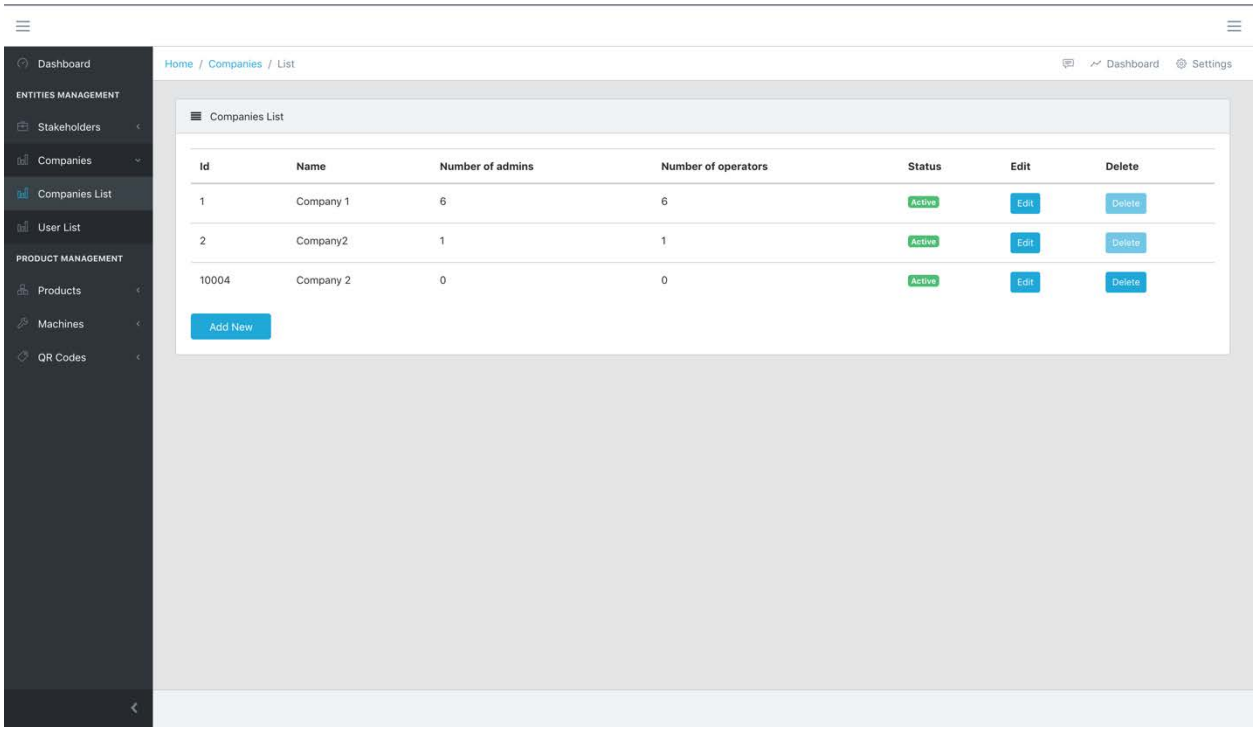
Companies

Companies module is composed of 2 parts:

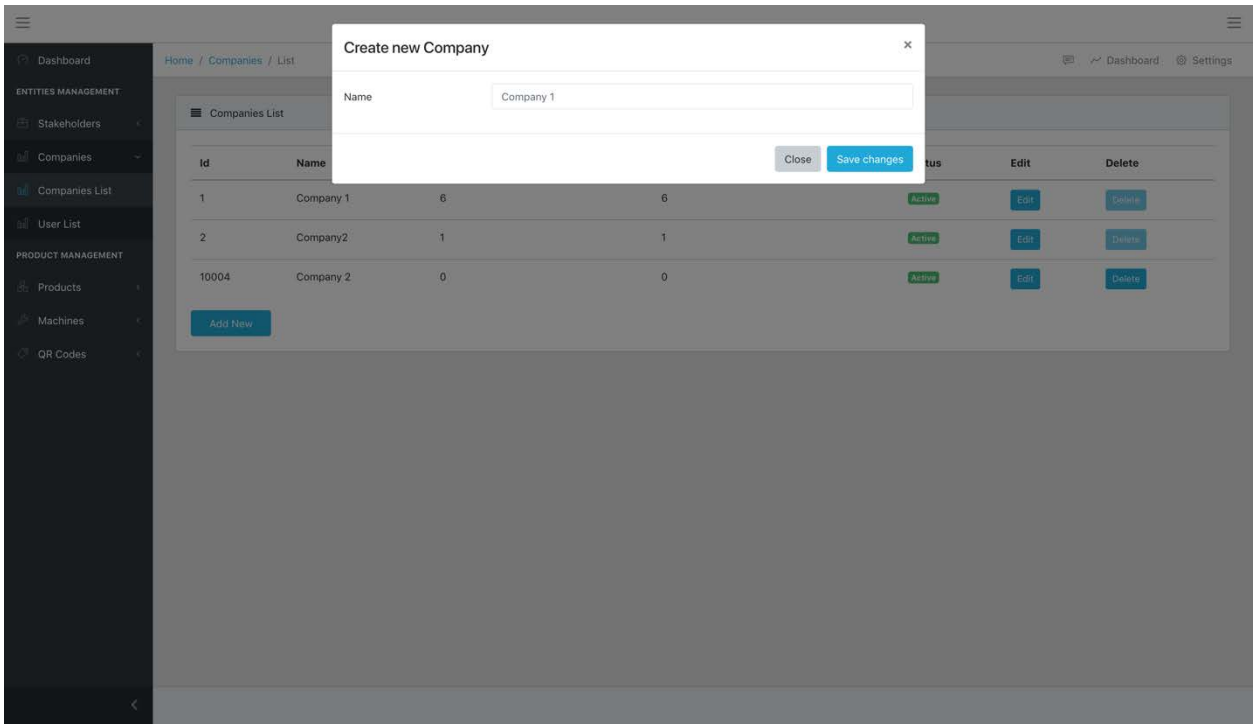
1. Companies List
2. Users List

Companies List

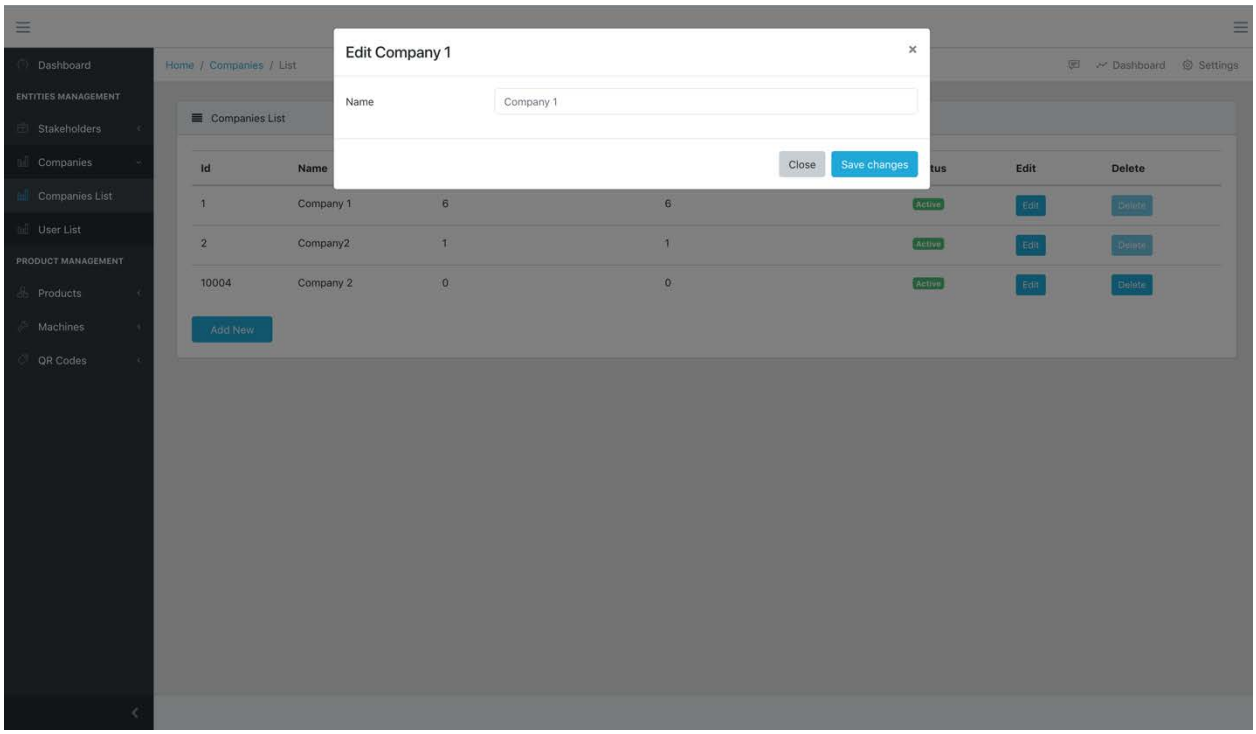
Companies List module is used for displaying and managing all companies in system.



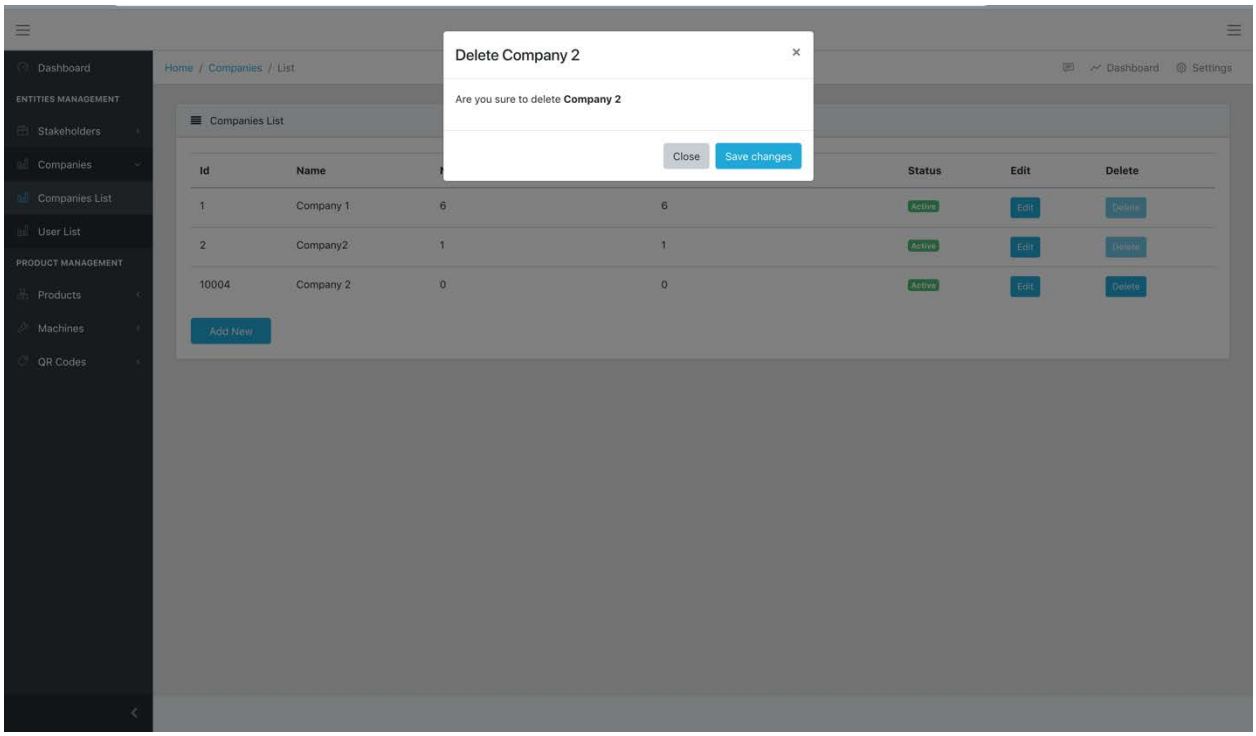
Add New


Create new Company
Name
Company 1
Close
Save changes
Dashboard
ENTITIES MANAGEMENT
Stakeholders
Companies
Companies List
User List
PRODUCT MANAGEMENT
Products
Machines
QR Codes
Home / Companies / List
Companies List
Id
Name
Edit
Delete
1
Company 1
2
Company2
10004
Company 2
Add New

Edit


Edit Company 1
Name
Company 1
Close
Save changes
Dashboard
ENTITIES MANAGEMENT
Stakeholders
Companies
Companies List
User List
PRODUCT MANAGEMENT
Products
Machines
QR Codes
Home / Companies / List
Companies List
Id
Name
Edit
Delete
1
Company 1
2
Company2
10004
Company 2
Add New

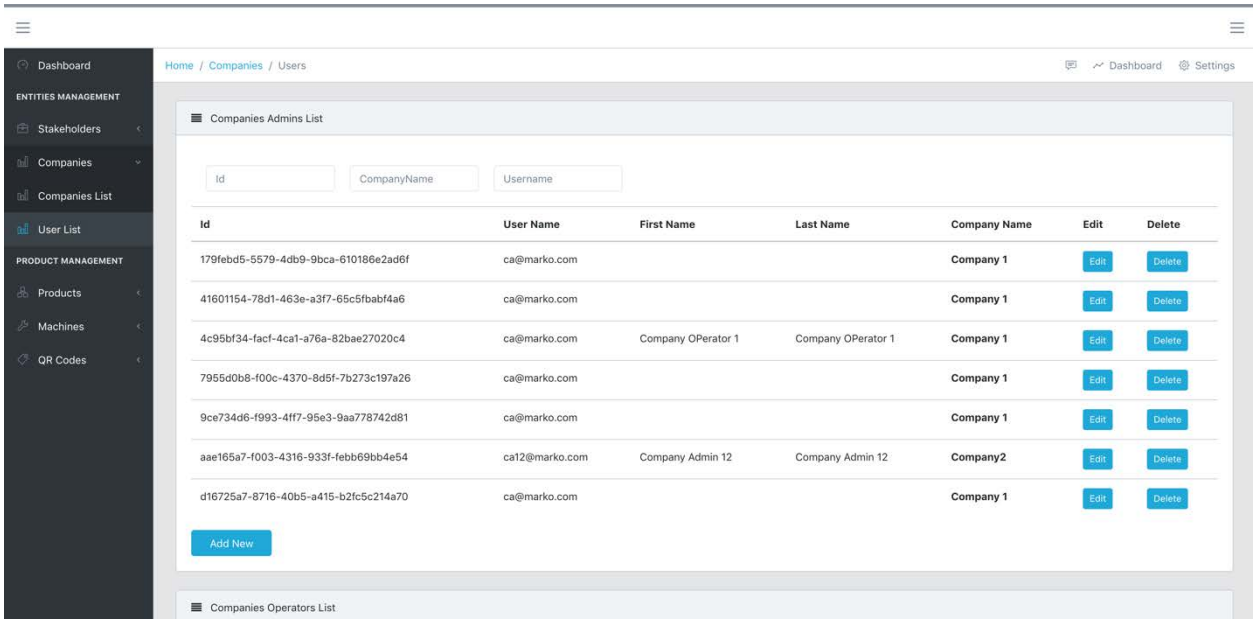
Delete



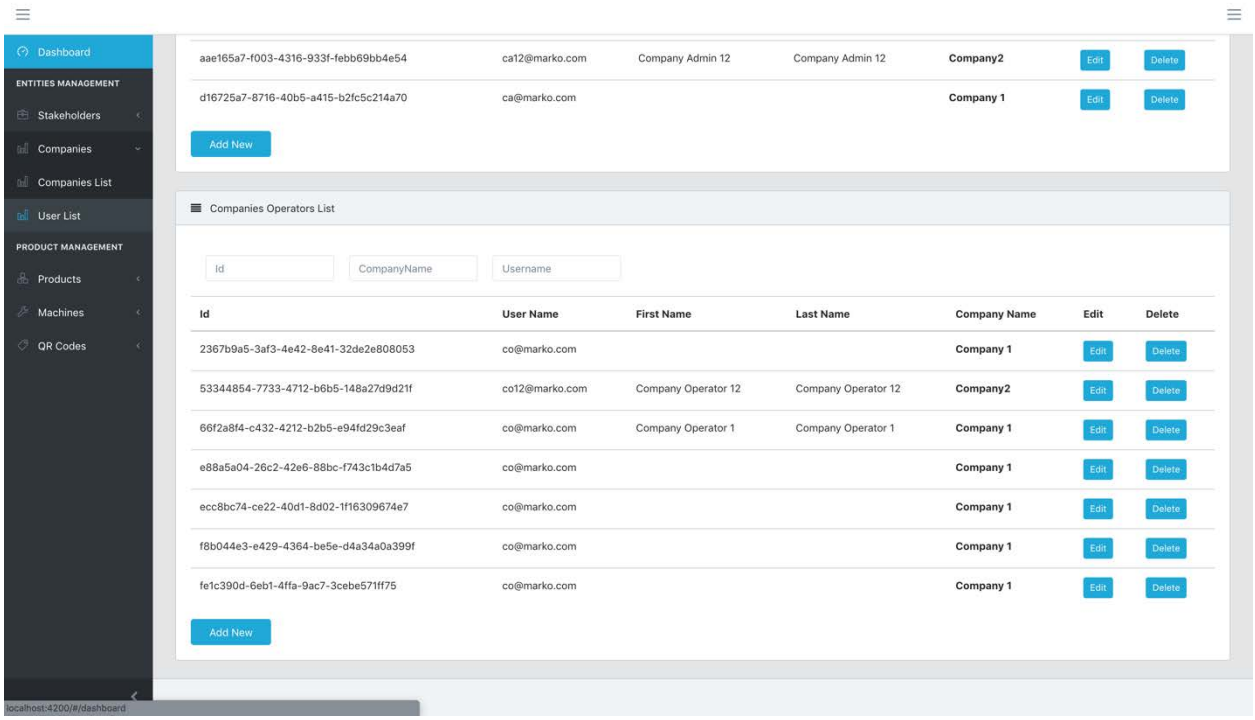
Companies Users List

Users List module is used for displaying and managing all companies users in system.

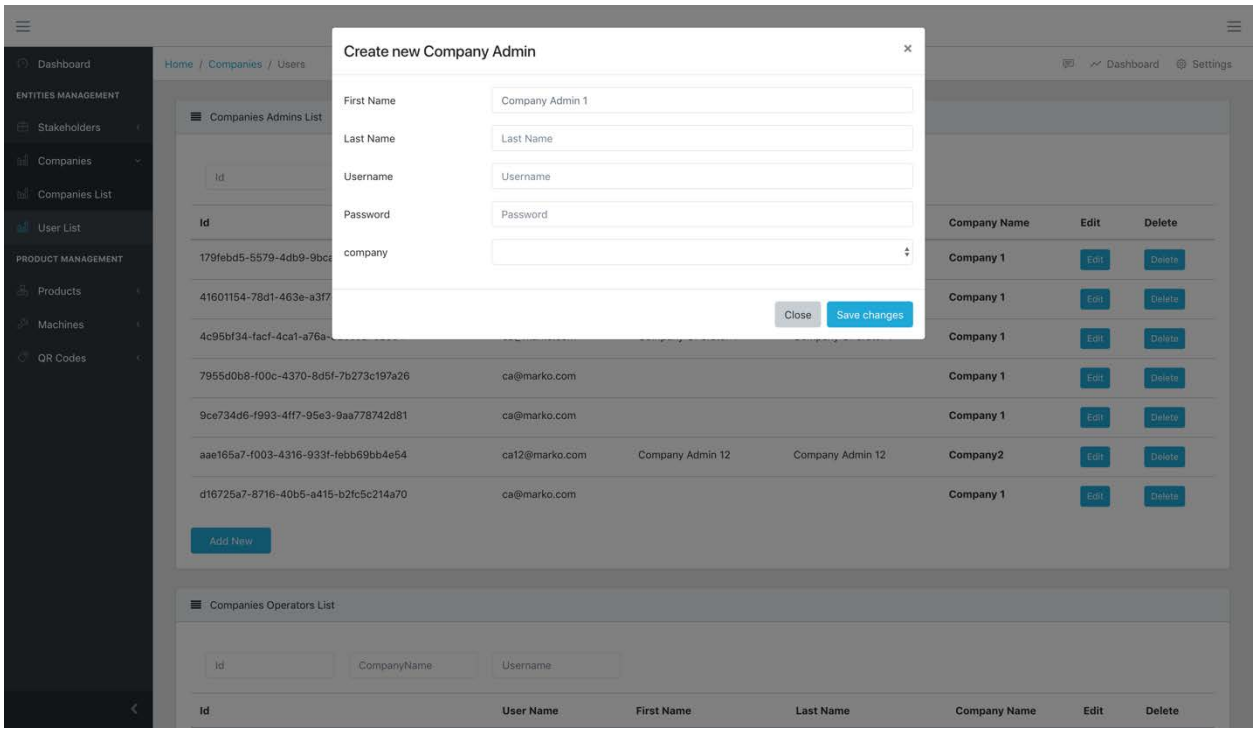
Administrator List



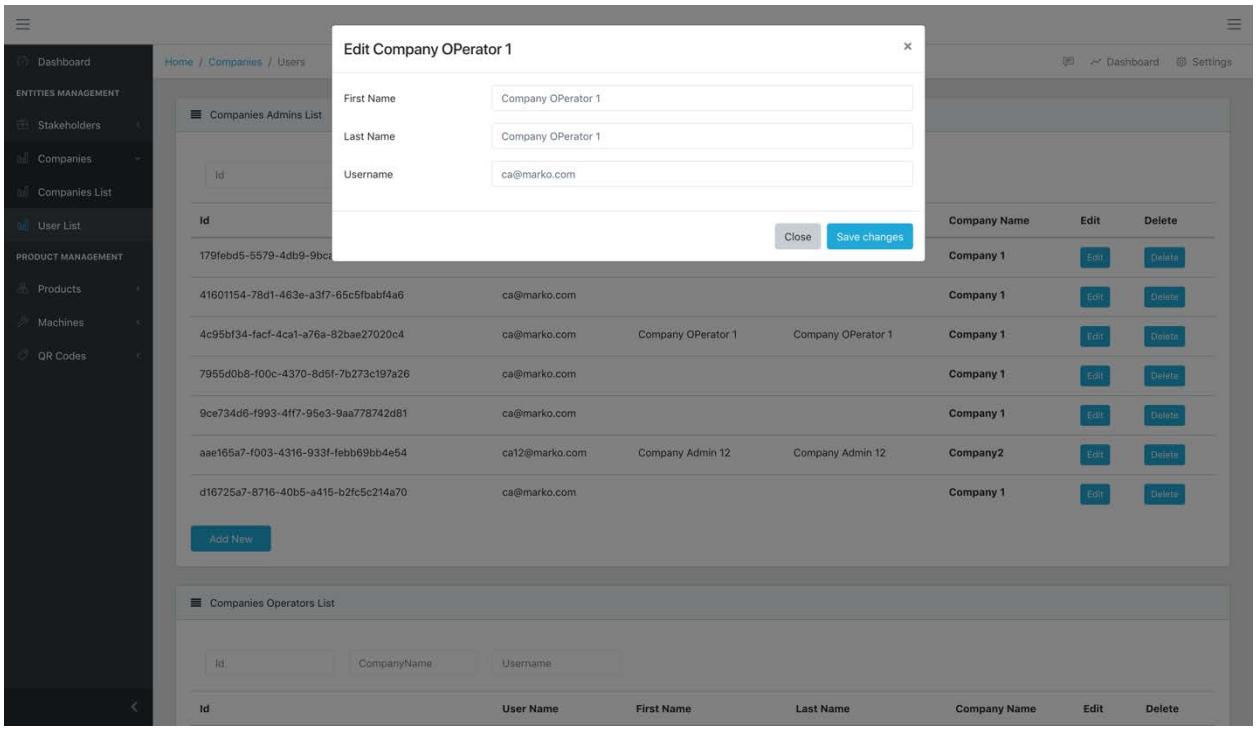
Operators List



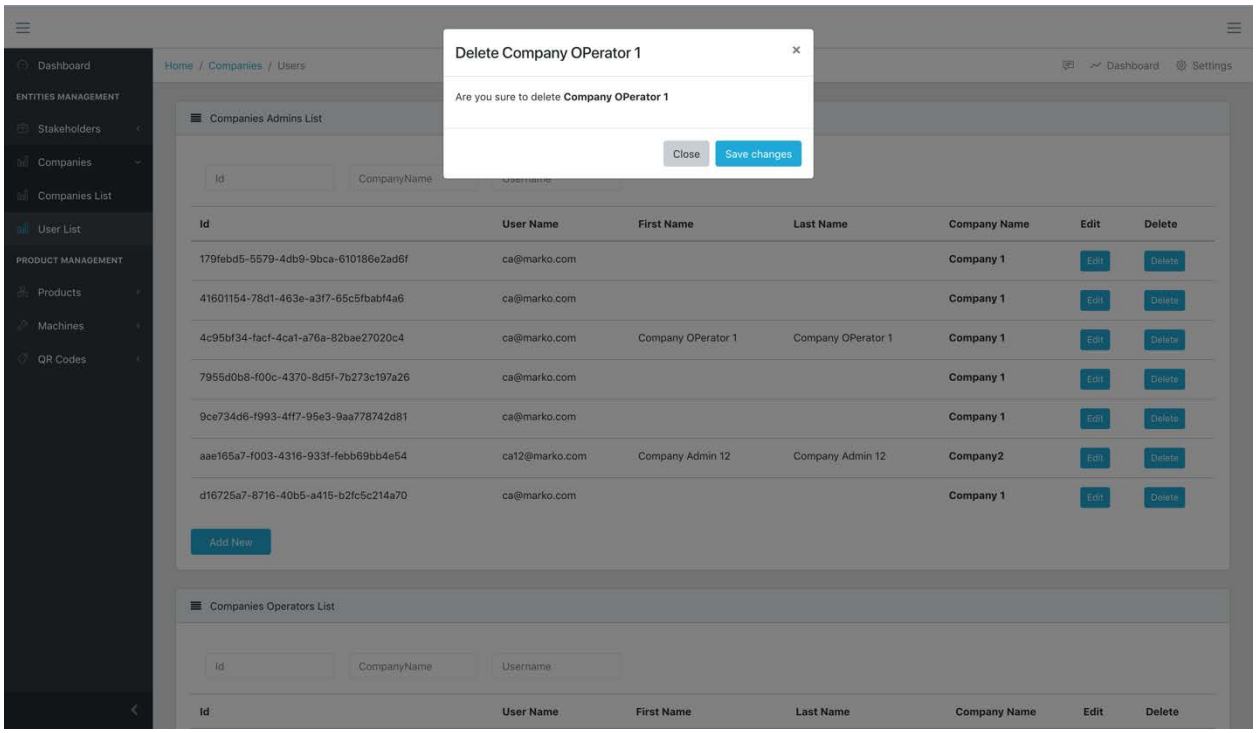
Add New



Edit


Edit Company OPerator 1
First Name
Company OPerator 1
Last Name
Company OPerator 1
Username
ca@marko.com
Close
Save changes
Companies Admins List
Companies Operators List
Add New

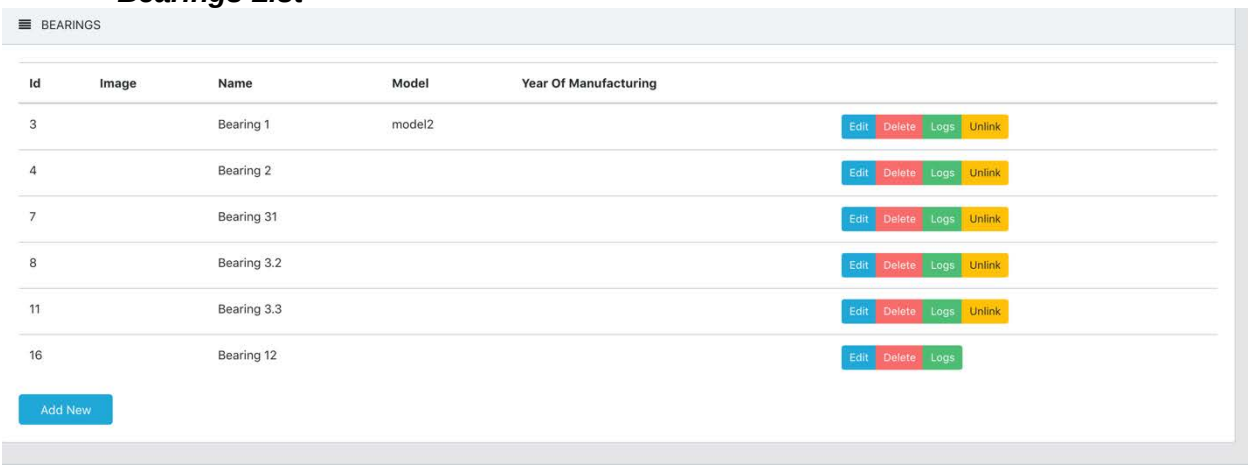
Delete


Delete Company OPerator 1
Are you sure to delete Company OPerator 1
Close
Save changes
Companies Admins List
Companies Operators List
Add New

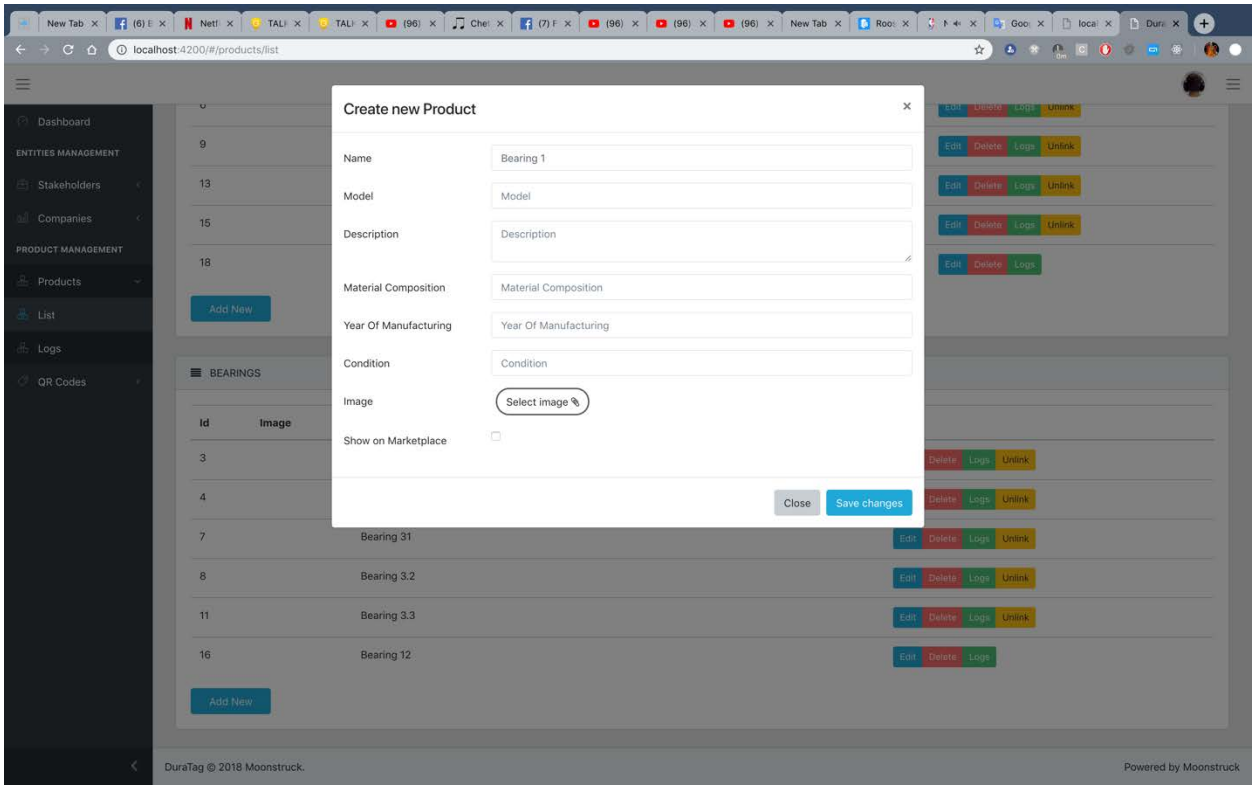
Products

Products module is used for displaying and managing all products in system.

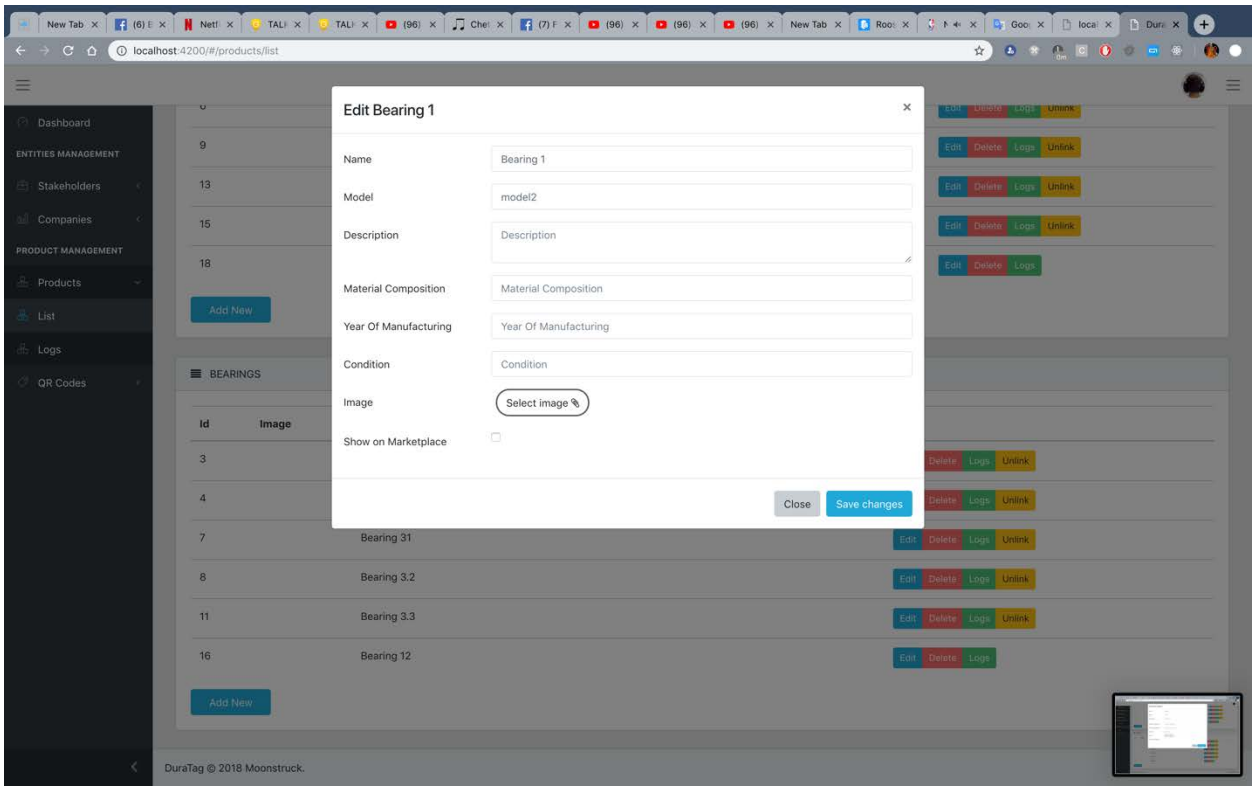
Bearings List



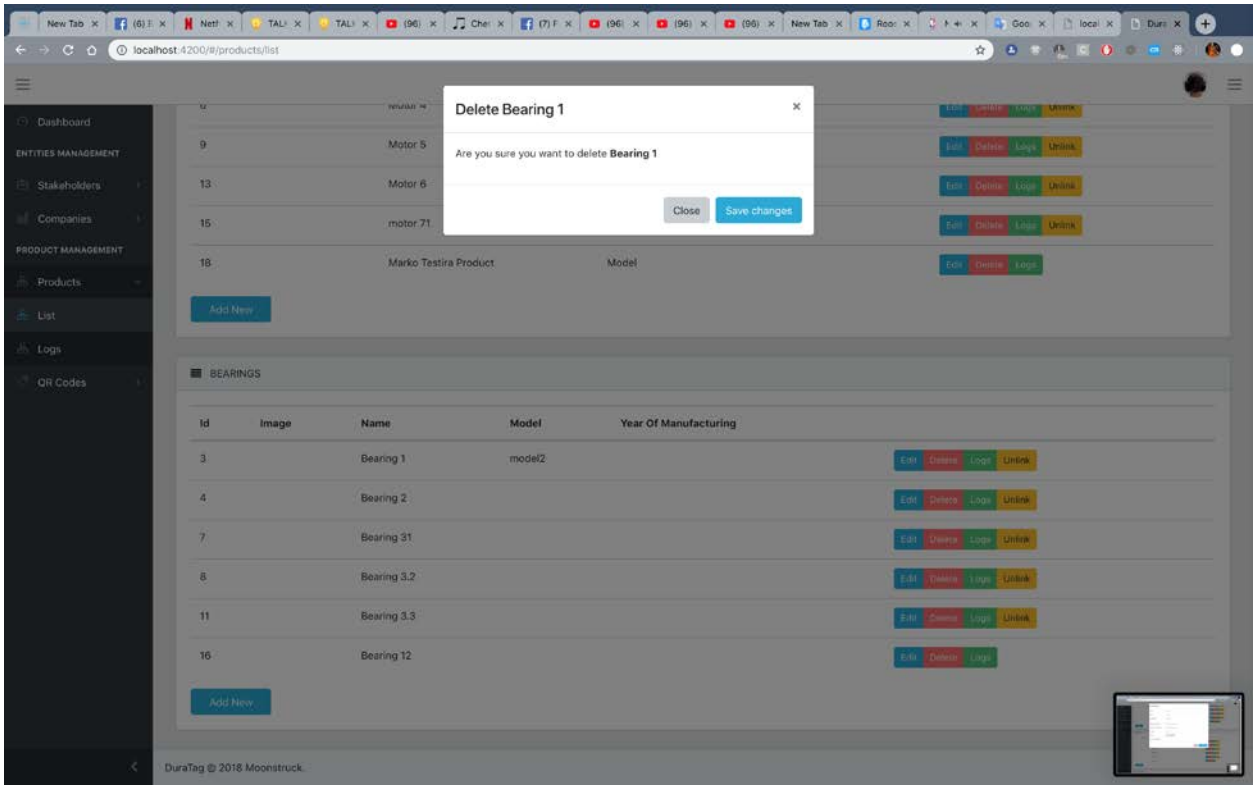
Add New



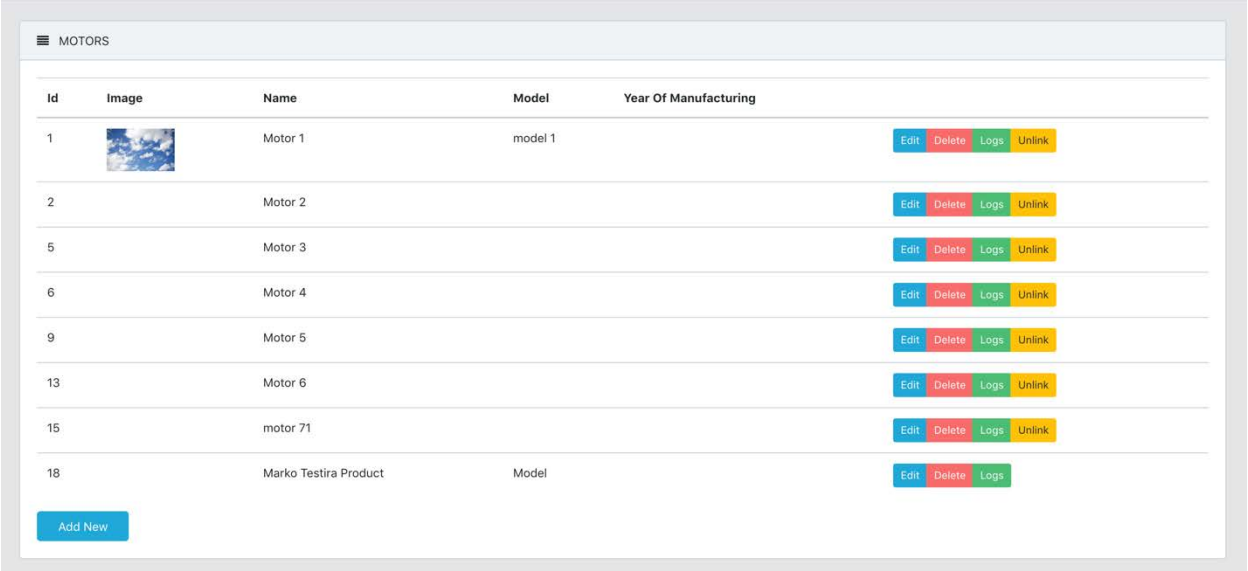
Edit


Edit Bearing 1
Name
Bearing 1
Model
model2
Description
Description
Material Composition
Material Composition
Year Of Manufacturing
Year Of Manufacturing
Condition
Condition
Image
Select image
Show on Marketplace
Close
Save changes
BEARINGS
Id
Image
Add New
DuraTag © 2018 Moonstruck.

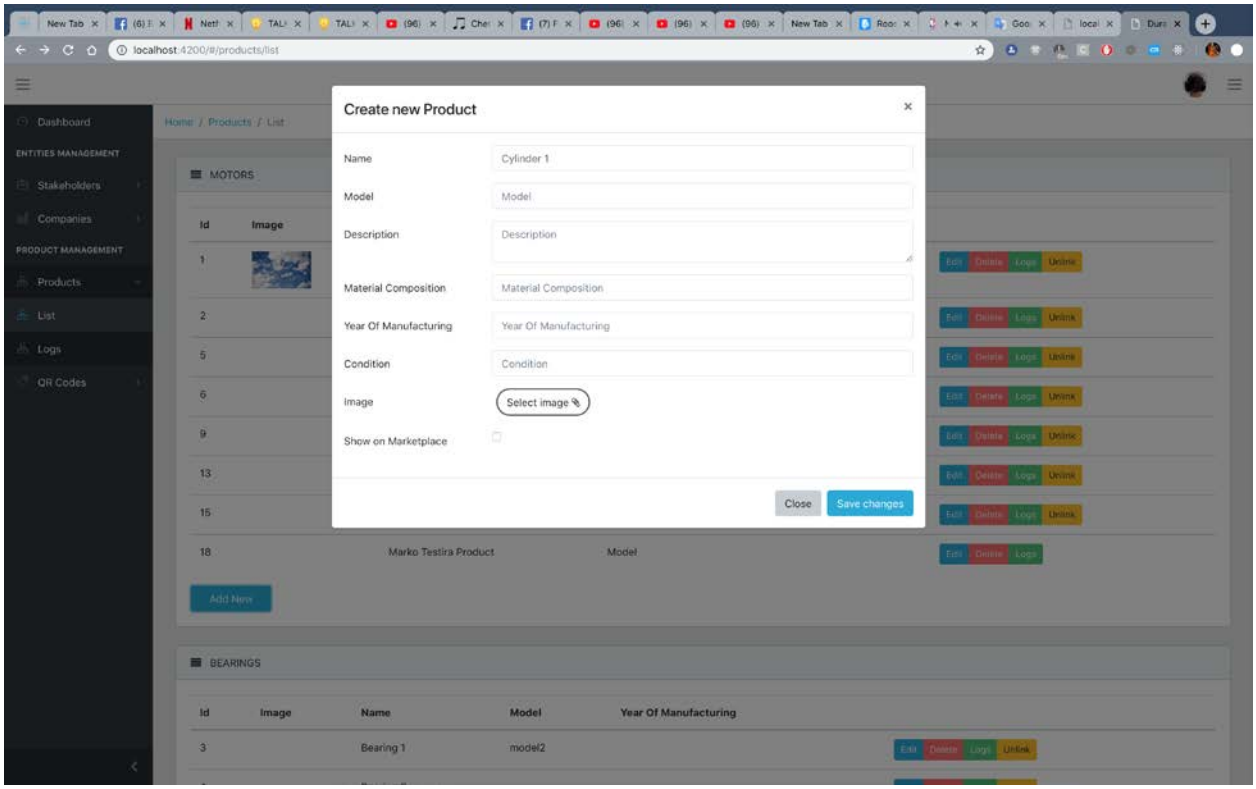
Delete


Delete Bearing 1
Are you sure you want to delete Bearing 1
Close
Save changes
BEARINGS
Id
Image
Name
Model
Year Of Manufacturing
Add New
DuraTag © 2018 Moonstruck.

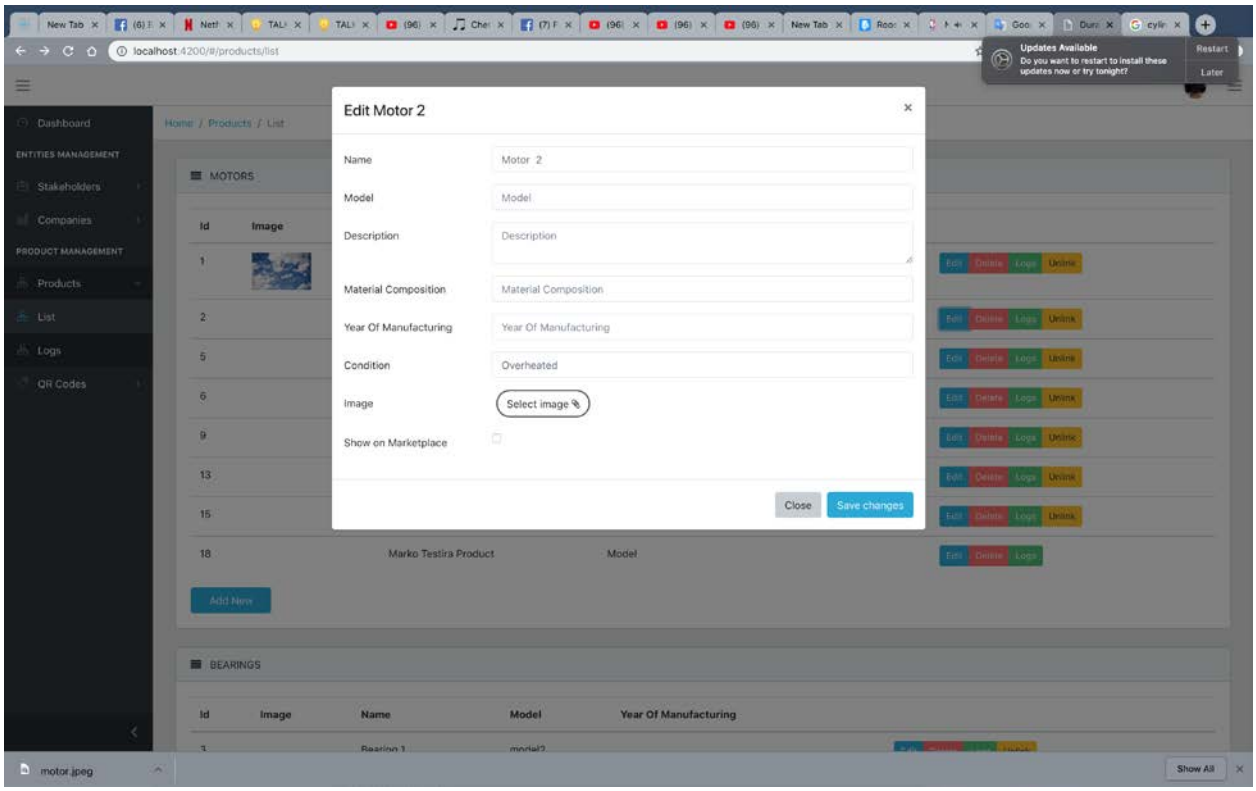
Motors List



Add New

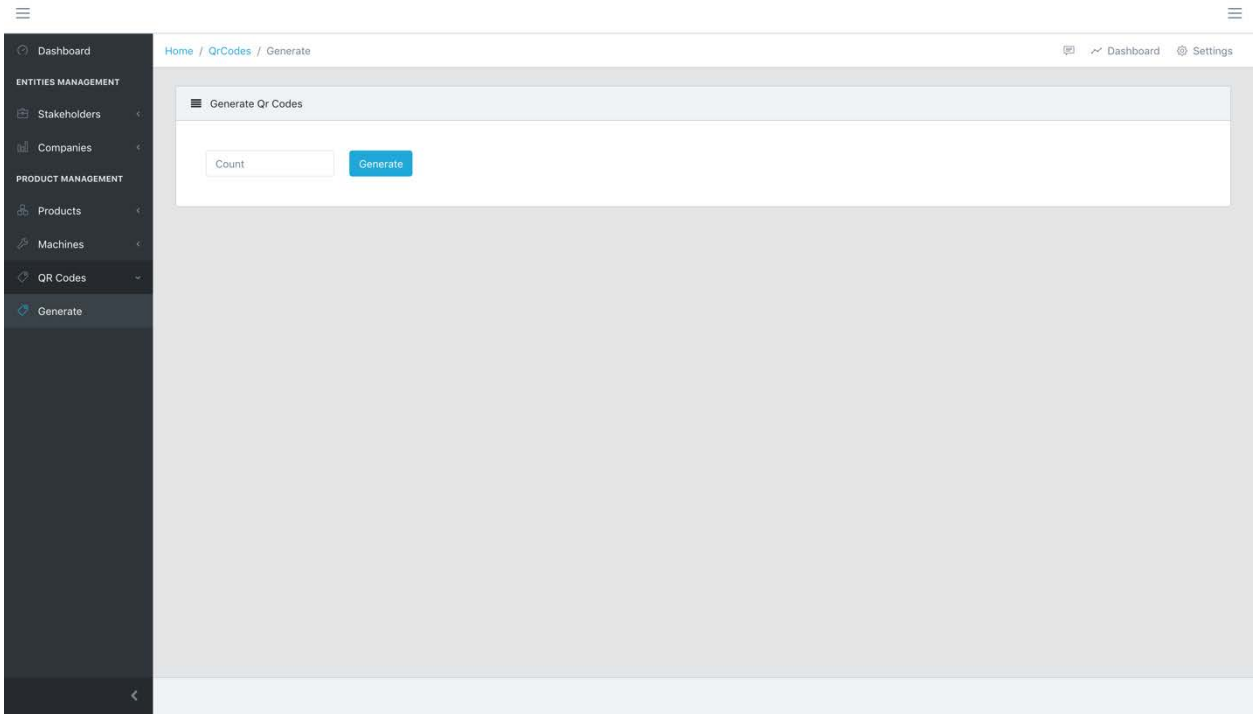


Edit



QR codes

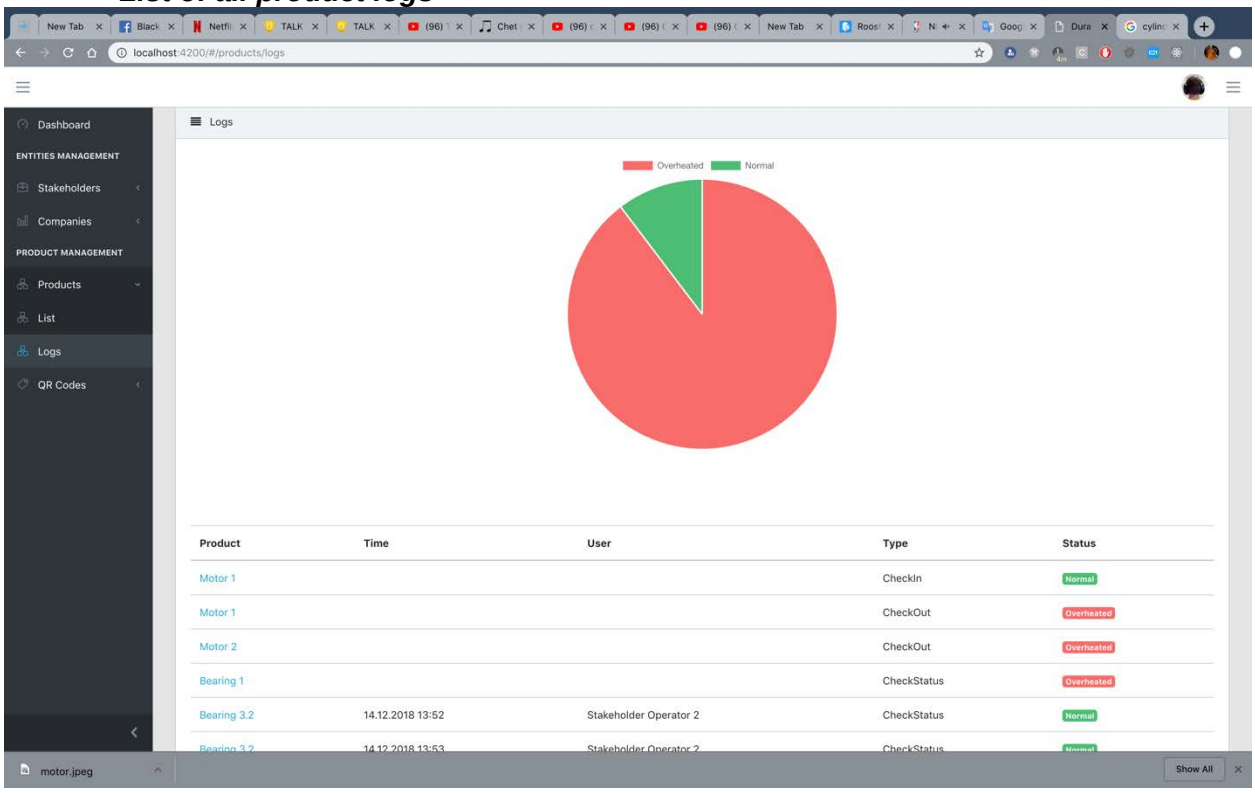
QR codes module is used for generating desired number of QR Codes batch.



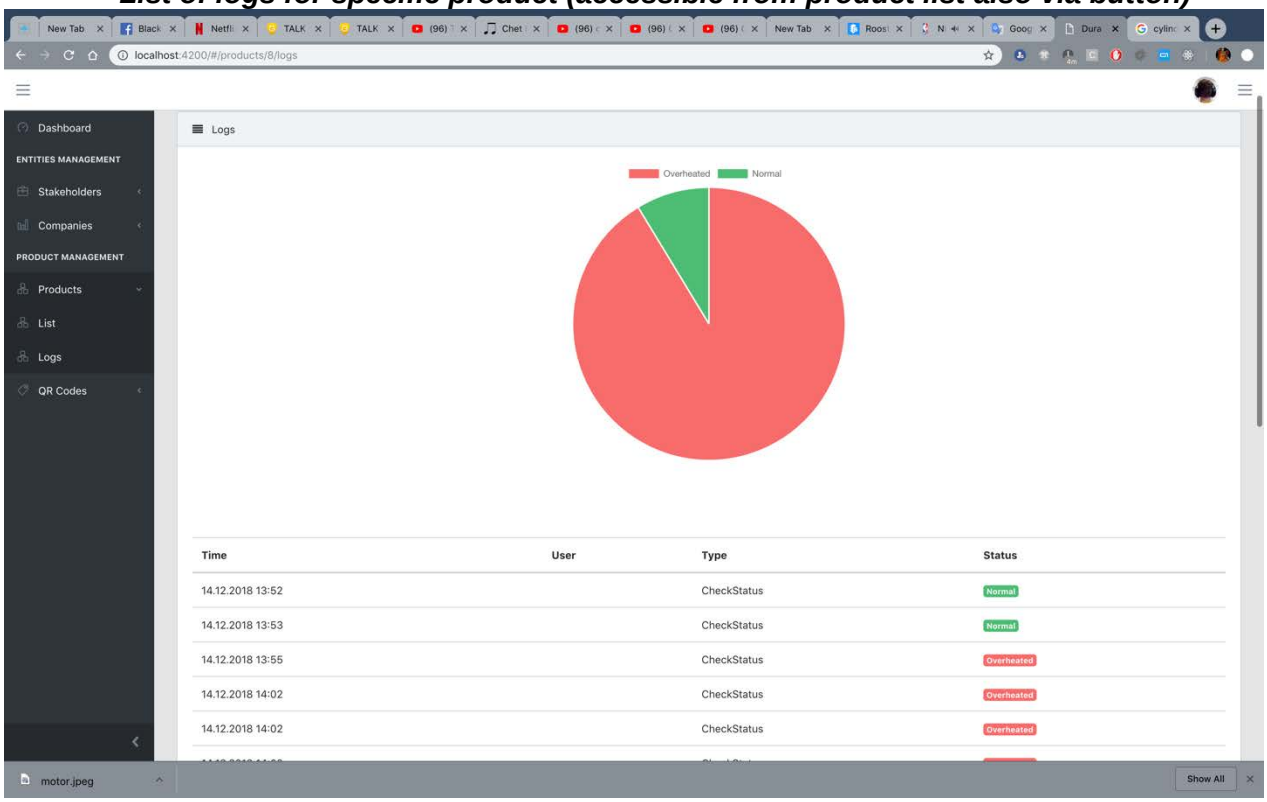
Logs

Logs module is used for viewing data, logged from scanning the product tags registered in the system.

List of all product logs



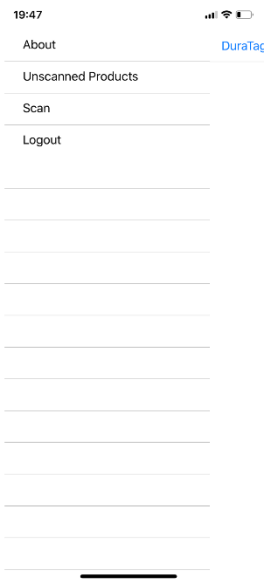
List of logs for specific product (accessible from product list also via button)



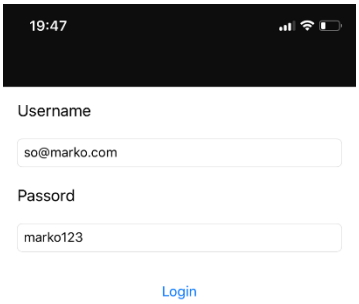
2.3 Mobile Application

Mobile Application is built using Xamarin Forms and open source ZXing library for scanning QR Codes. Application is communicating with API described in previous sections.

Application has login page and is composed of several parts:



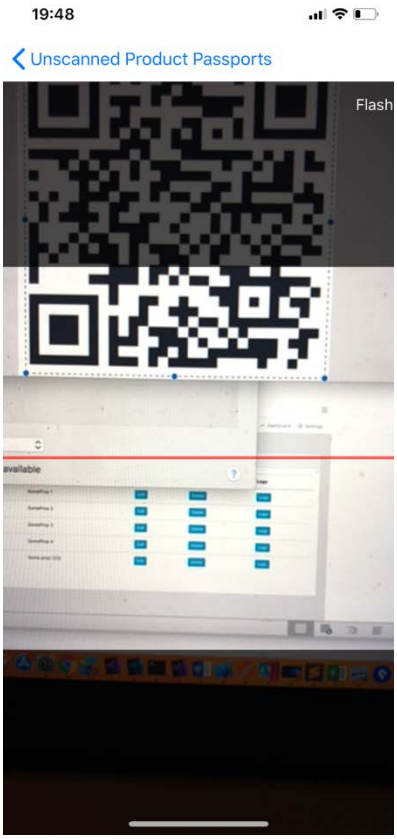
Login Page



Unscanned Products

This page is used for registering newly added products with associated QR codes.

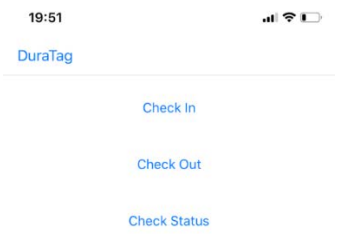
When user selects each unscanned product, scanning view screen is automatically presented, for scanning the QR code on the SmartTag (QR code is not depicted here in full view, because it is impossible to capture screenshot when scanning QR code is that fast).



Scanning Page

On scanning screen, there are 3 functional buttons:

- **CheckIn** – this button is used for user to check in at start of working day for desired product by scanning its QR code. It also checks if there is temperature indicator present on QR code, and it sends corresponding data to API.
- **CheckOut** – this button is used for user to check out at the end of working day for desired product by scanning its QR code. It also checks if there is temperature indicator present on QR code, and it sends corresponding data to API.
- **CheckStatus** – this button is used for user to check status of desired product by scanning its QR code.


19:51
DuraTag
Check In
Check Out
Check Status

Section 3 - Conclusions

In this document the DuraTag technical platform details are specified for the developed software components enabling reuse, refurbish, and repair of mechanical parts and materials in machinery industry, specifically waste processing machinery. The FC-scanner component is modified, SmartTags are encoded using developed encoding modul and printed by TagItSmart consortium. Web Service is developed to enable communication with the platform not only for the users and FC-scanner, but the APIs can also be used by third party to exploit extracted data and information about composition materials, mechanical parts and its current condition via Product Passport.