https://www.halvorsen.blog

Tool Inventory System

Hans-Petter Halvorsen

Tool Inventory System - Background

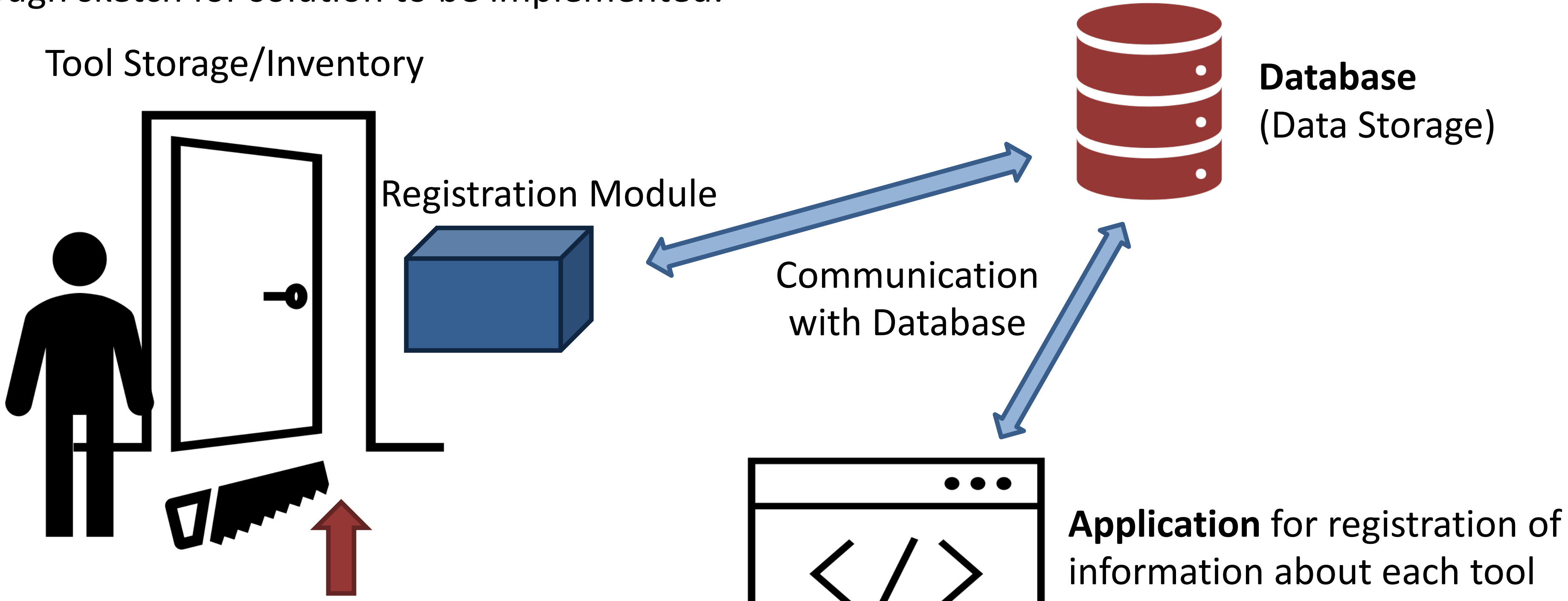
- A company has a room (inventory room) full of expensive tools
- The tools are used by the employees in their daily work
- The company wants to keep track of the tools in this inventory room
- The door into the room has access control (ID card is needed to open the door) but there is no tracking of the tools

Tool Inventory System - Goals

- The system should track tools in the inventory, i.e., track when someone borrow tools and when someone return tools
- The system should have a card reader at the door to the tool storage/inventory, which in addition to unlocking the door means that the tool that is taken out is registered on this person.

Tool Inventory System – Suggested Solution

Rough sketch for solution to be implemented:


Tool Storage/Inventory
Registration Module
Communication with Database
Database
(Data Storage)
Application for registration of information about each tool

Functional Requirements

- The system should track tools in the inventory, i.e., track when someone borrow tools and when someone return tools
- The system should have a card reader at the door to the tool storage/inventory, which in addition to unlocking the door means that the tool that is taken out is registered on this person.
- The system should automatically register the time and person in the database when tools are taken out (/ in) of the inventory.
- The system should automatically register when a tool leaves the inventory
- The inventory should have access control
- The following information should be stored for each tool: Tool Number, Vendor, Model, Purchase Date, Calibration Date, Price, Owner, Categories and Subcategories

Functional Requirements

- A program to easily add new tools or modify existing ones in the database, preferably with a web interface for easier access
- Create a graphical display application on a screen:
  - If the tool is outside / inside
  - When it was taken out
  - Person who has taken the tool
  - Possible picture of tools
  - Here it is possibly desirable with a touch screen to be able to browse the list, perhaps with a small search function or some filtering options, but without access to make changes.

Functional Requirements

- It should be possible to get lists of available tools, possibly who borrowed it and when, etc.
- It should be possible to send reminders to get tools back, etc.
- It should be possible to add users, edit user information and delete old non existing users
- The user needs to login to the web system with a Username and a Password
- Possible to reserve tools that you need

Non-Functional Requirements

- The system shall use RFID tags for access control
- The system should be wireless, preferably using RFID tags for tracking tools in the inventory
- The range for the RFID reader should be 2m
- The web application should be created using non-licensed based software
- The Web Application should preferable be hosted on Microsoft Azure because the customer already uses this service
- The system and programs should be designed in such a way that they are as general and configurable as possible.

Non-Functional Requirements

- The system should satisfy the GDPR and satisfy basic security requirements, login, etc.
- The web application(s) should work with both Microsoft Edge and Google Chrome
- The customer mainly wants a total solution that is as license free as possible. Assessments must be made of development tools, programming languages, frameworks, etc. so that this is taken care of in the best possible way
- The User Passwords should have necessary security such as “Hashing”
- The system should be modularized for easy maintenance and further development by the customer
- The code should be well structured and well documented to make it possible for the customer to maintain and to further develop the system

RFID

- Radio-frequency identification (RFID)
- RFID is the method of uniquely identifying items using radio waves
- An RFID system comprises a tag, a reader, and an antenna
- Unlike a barcode, the tag does not need to be within the line of sight of the reader

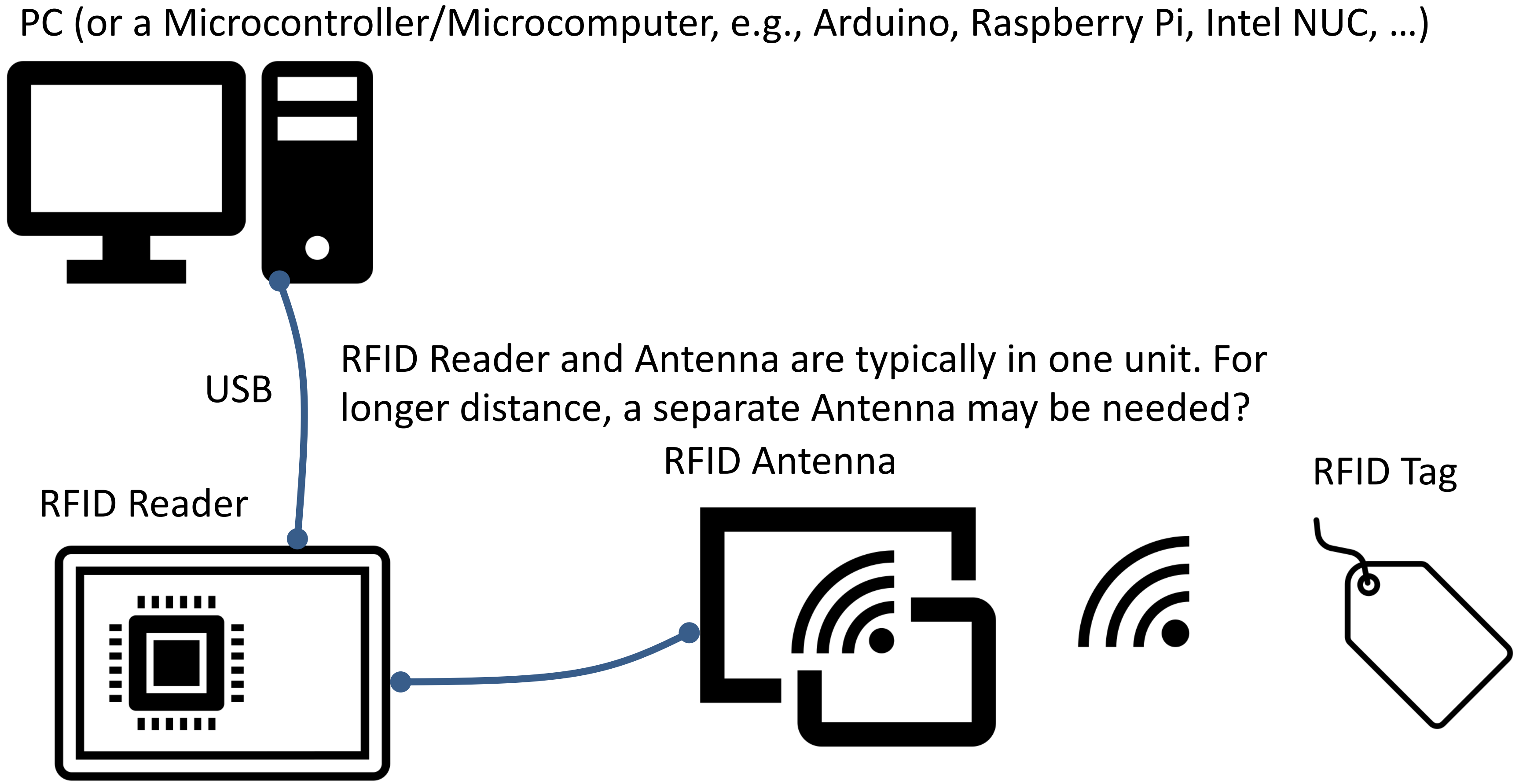
https://en.wikipedia.org/wiki/Radio-frequency_identification

RFID Parts

- An **RFID tag** in its most simplistic form, is comprised of two parts – an antenna for transmitting and receiving signals, and an RFID chip (or integrated circuit, IC) which stores the tag's ID and other information. RFID tags are affixed to items in order to track them using an RFID reader and antenna.
- An **RFID reader** is the brain of the RFID system and is necessary for any system to function
- **RFID Antennas** are necessary elements in an RFID system because they convert the RFID reader's signal into RF waves that can be picked up by RFID tags
- Many RFID readers has an integrated antenna

https://www.atlasrfidstore.com/rfid-beginners-guide/

RFID System Overview


PC (or a Microcontroller/Microcomputer, e.g., Arduino, Raspberry Pi, Intel NUC, …)
USB
RFID Reader
RFID Antenna
RFID Tag

RFID Reader and Antenna are typically in one unit. For longer distance, a separate Antenna may be needed?

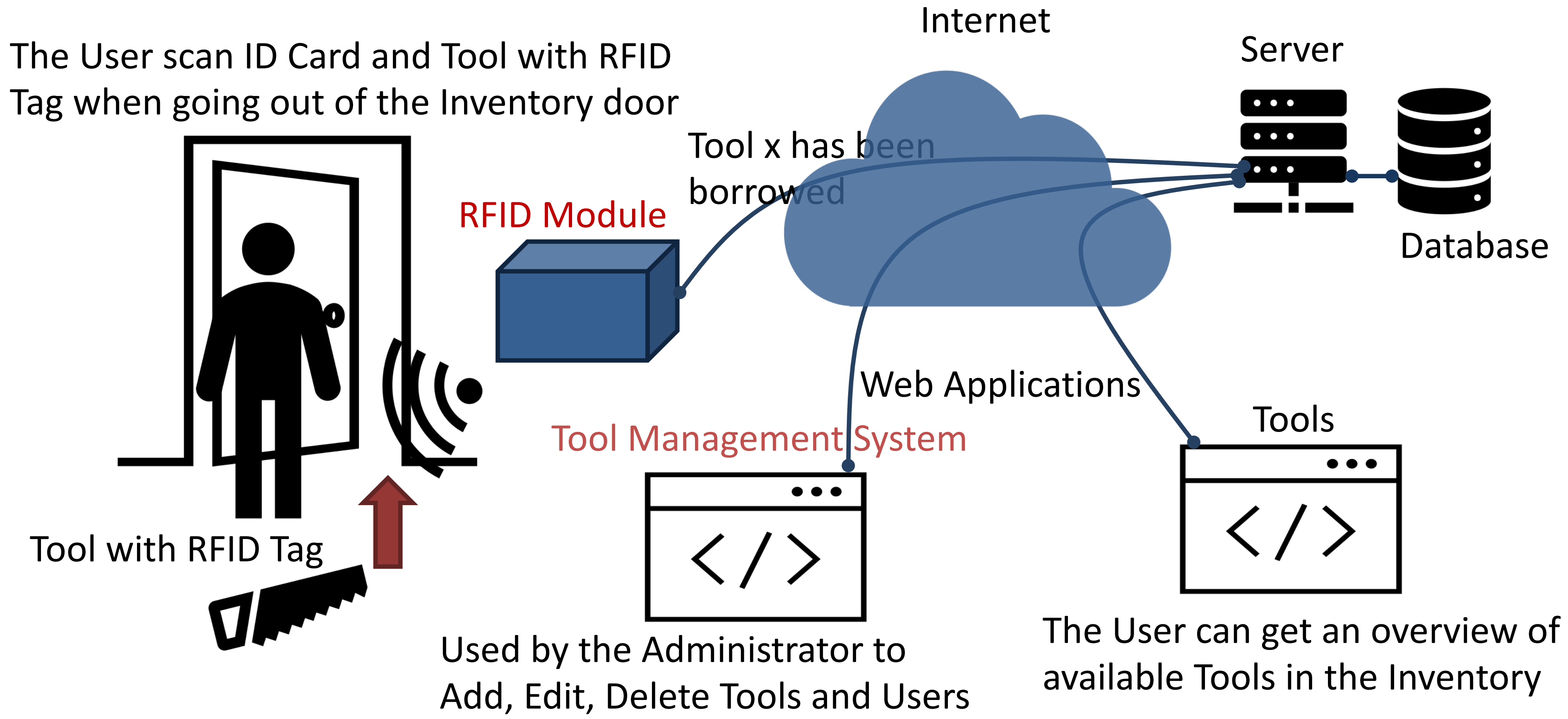
RFID Tags

There are two types of RFID tags:

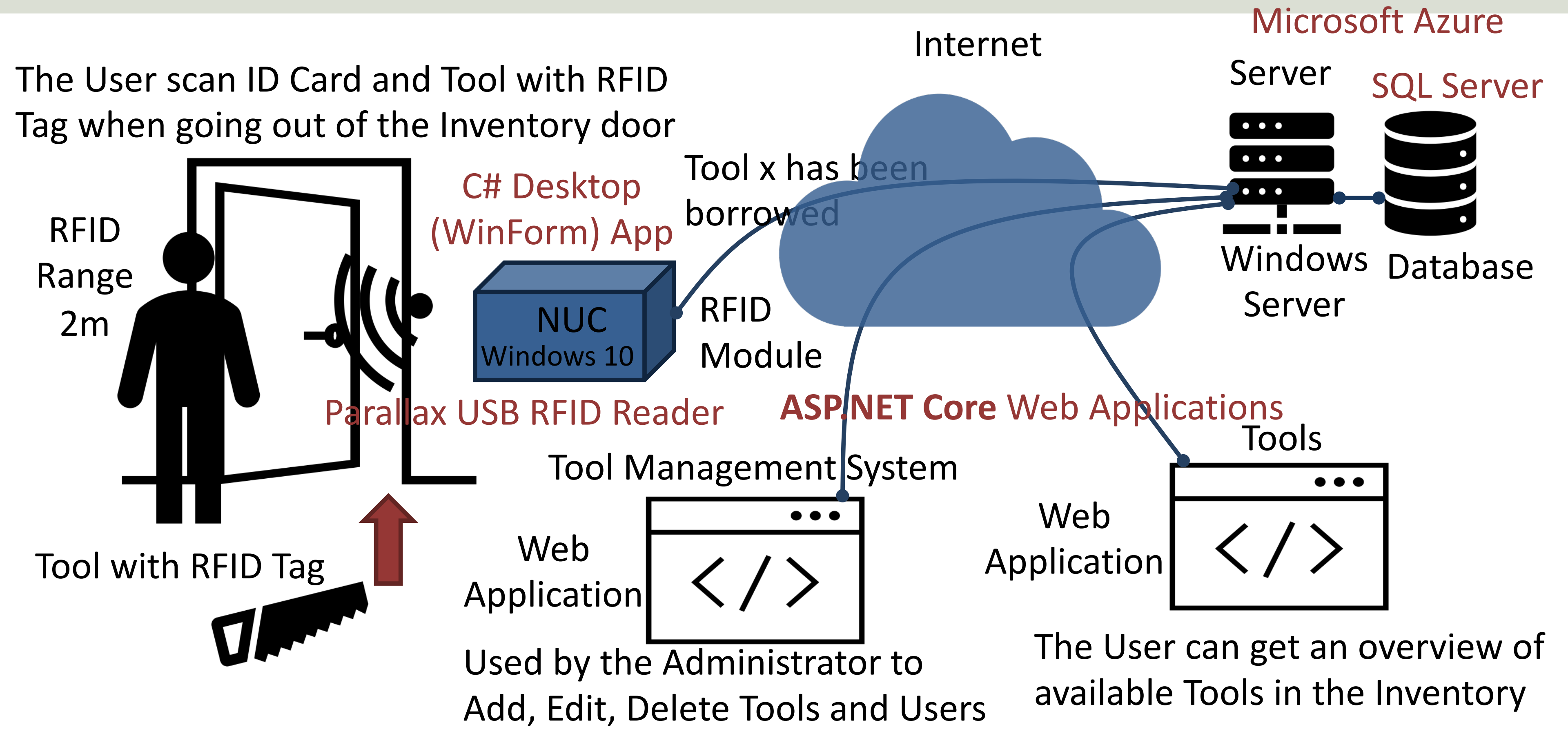
- **Passive** tags are powered by energy from the RFID reader's interrogating radio waves.
- **Active** tags are powered by a battery and thus can be read at a greater range from the RFID reader, up to hundreds of meters.

RFID tags can be attached to physical objects, clothing, and possessions, or implanted in animals and people

Inventory System - System Overview


Internet
Server
Database
The User scan ID Card and Tool with RFID Tag when going out of the Inventory door
Tool x has been borrowed
RFID Module
Web Applications
Tools
Tool Management System
Tool with RFID Tag
Used by the Administrator to Add, Edit, Delete Tools and Users
The User can get an overview of available Tools in the Inventory

Inventory System - Architecture



Risk Analysis

- The RFID reader has not the necessary range
- Tool registered on wrong person if 2 persons are in the inventory and/o leave the room at the same time
- Existing Access Card cannot be used because they don’t have RFID or different standard is used
- Risk of GDPR violation
- Hacker attacks
- Data stored in a cloud service out of control of the company
- User data/information stored in 2 different systems (this system and existing access control system)
- What if an ID Card has been stolen?
- A person uses an ID card that is not his (borrowed from another)
- The RFID Tag on the tool has fallen of
- The tools are used in a hazardous environment which can case that the RFID falls off or is destroyed in some way

Prototype

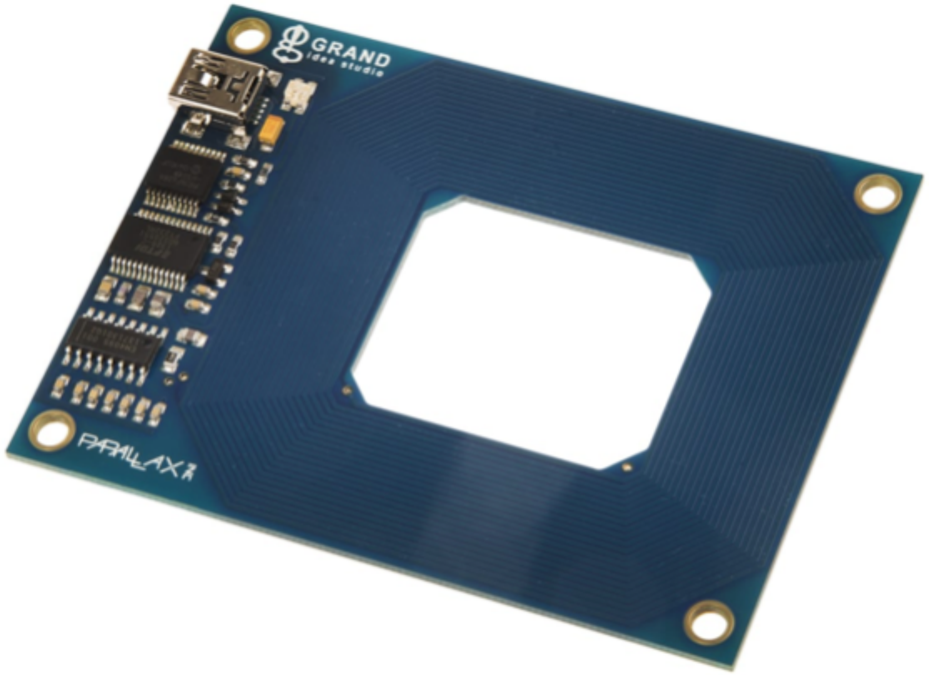
Prototyping Development Time

20-80 Rule:

- It takes 20% of the total development time to make the system 80% finished (The main functionality but it lacks robustness, systematic testing, etc.)
- It takes 80% of the time to finish the remaining 20% of the system (Robustness, bug fixes, fin-tuning, change in requirements and customer wants some changes or new functionality, etc.)
- 80% of the users are only using 20% of the features in an application

RFID Reader

Parallax USB RFID Reader



User ID Cards with RFID for Access Control

RFID Tags for mounting on each tool

https://www.parallax.com/product/rfid-card-reader-usb/

https://no.rs-online.com/web/p/identification-sensors/8430793

Parallax USB RFID Reader

From Parallax USB RFID Reader Documentation

- It reads passive **125 kHz** RFID transponder tags
- The Parallax RFID Card Reader USB version can be connected directly to any PC, Macintosh, or Linux machine that has a USB port and the appropriate drivers installed. The module is powered from the host computer's USB port and uses an industry-standard **FTDI FT232R** device to provide the USB connectivity
- A visual indication of the state of the RFID Card Reader is given with the on-board LED. When the module is successfully powered-up and is in an idle state, the LED will be **GREEN**. When the module is in an active state searching for or communicating with a valid tag, the LED will be **RED**.
- The RFID Card Reader USB version is activated via the **DTR** line of the USB Virtual COM port. When the DTR line is set HIGH, the module will enter the active state. When the DTR line is set LOW, the module will enter the idle state.
- RFID Tag read distance of approximately 4 inches (**10cm**).

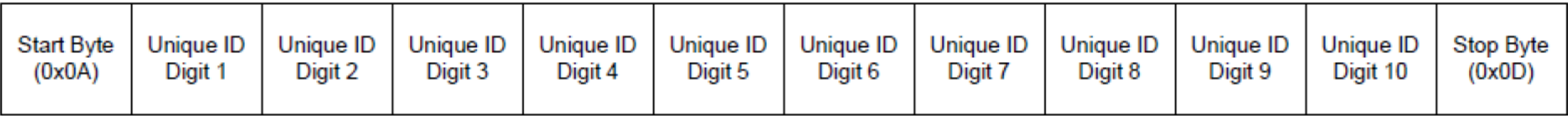
Parallax USB RFID Reader

Communication Protocol:

- The RFID Card Reader USB version transmits the data through the USB Virtual COM Port driver
- All communication is 8 data bits, no parity, 1 stop bit, and least significant bit first (8N1) at 2400 bps.
- When the RFID Card Reader is active and a valid RFID transponder tag is placed within range of the activated reader, the tag's unique ID will be transmitted as a 12-byte printable ASCII string serially to the host in the following format:

Parallax USB RFID Reader

Communication Protocol:



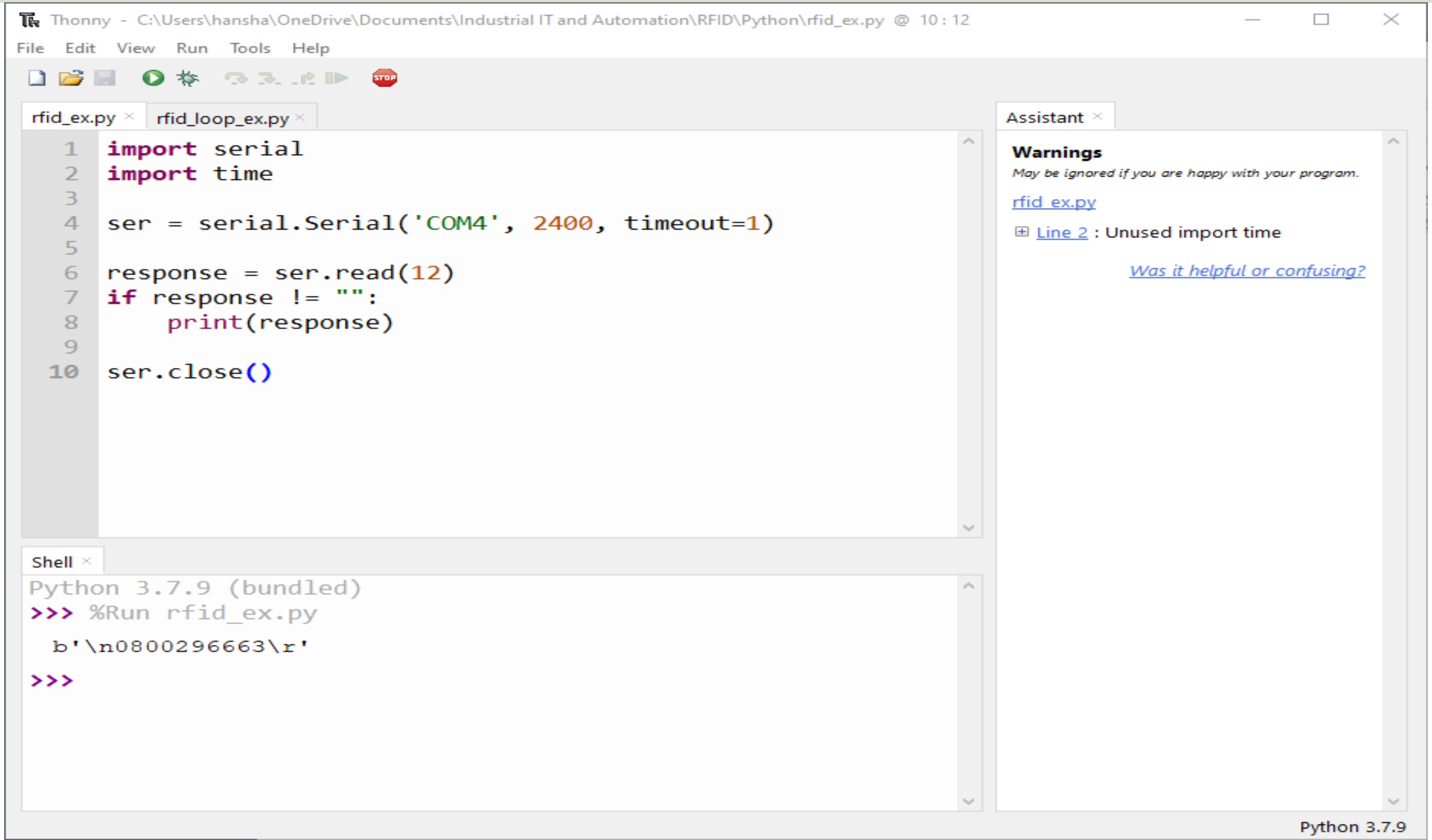
Start Byte (0x0A)	Unique ID Digit 1	Unique ID Digit 2	Unique ID Digit 3	Unique ID Digit 4	Unique ID Digit 5	Unique ID Digit 6	Unique ID Digit 7	Unique ID Digit 8	Unique ID Digit 9	Unique ID Digit 10	Stop Byte (0x0D)

The start byte and stop byte are used to easily identify that a correct string has been received from the reader (they correspond to line feed (\n)and carriage return (\r) characters, respectively).

The middle ten bytes are the actual tag's unique ID.

For example, for a tag with a valid ID of 0F0184F07A, the following bytes would be sent: 0x0A, 0x30, 0x46, 0x30, 0x31, 0x38, 0x34, 0x46, 0x30, 0x37, 0x41, 0x0D.

RFID Python Prototype


Thonny - C:\Users\hansha\OneDrive\Documents\Industrial IT and Automation\RFID\Python\rfid_ex.py @ 10 : 12
File Edit View Run Tools Help
rfid_ex.py
rfid_loop_ex.py
Assistant
Shell
Python 3.7.9

```python
import serial
import time

ser = serial.Serial('COM4', 2400, timeout=1)

response = ser.read(12)
if response != "":
    print(response)

ser.close()
```

```
Python 3.7.9 (bundled)
>>> %Run rfid_ex.py
  b'\n0800296663\r'
>>>
```

Warnings

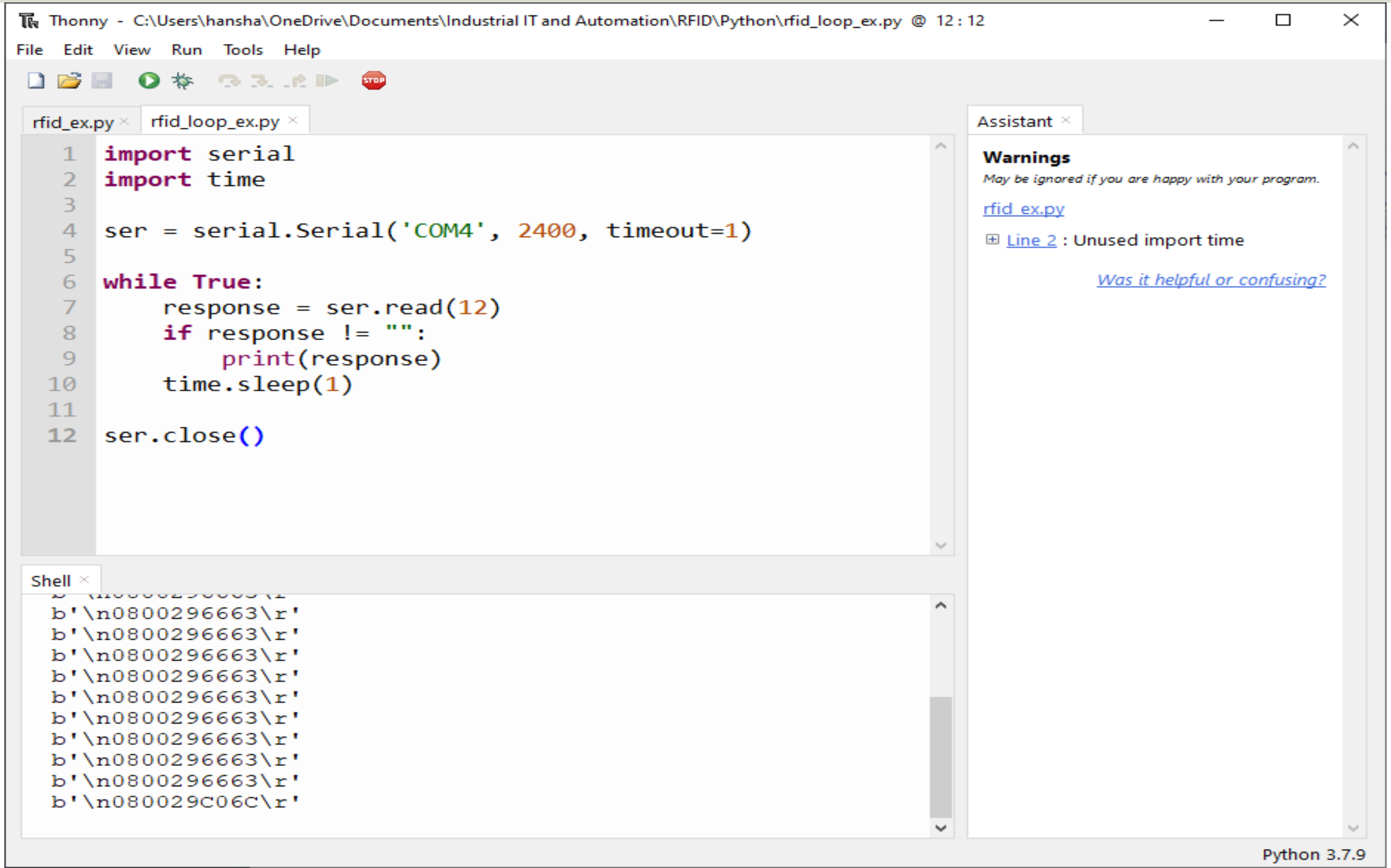
May be ignored if you are happy with your program.

rfid_ex.py

Line 2 : Unused import time

Was it helpful or confusing?

RFID Python Prototype


Thonny - C:\Users\hansha\OneDrive\Documents\Industrial IT and Automation\RFID\Python\rfid_loop_ex.py @ 12 : 12
File Edit View Run Tools Help
rfid_ex.py
rfid_loop_ex.py
Assistant
Shell
Python 3.7.9

```python
import serial
import time

ser = serial.Serial('COM4', 2400, timeout=1)

while True:
    response = ser.read(12)
    if response != "":
        print(response)
    time.sleep(1)

ser.close()
```

Warnings

May be ignored if you are happy with your program.

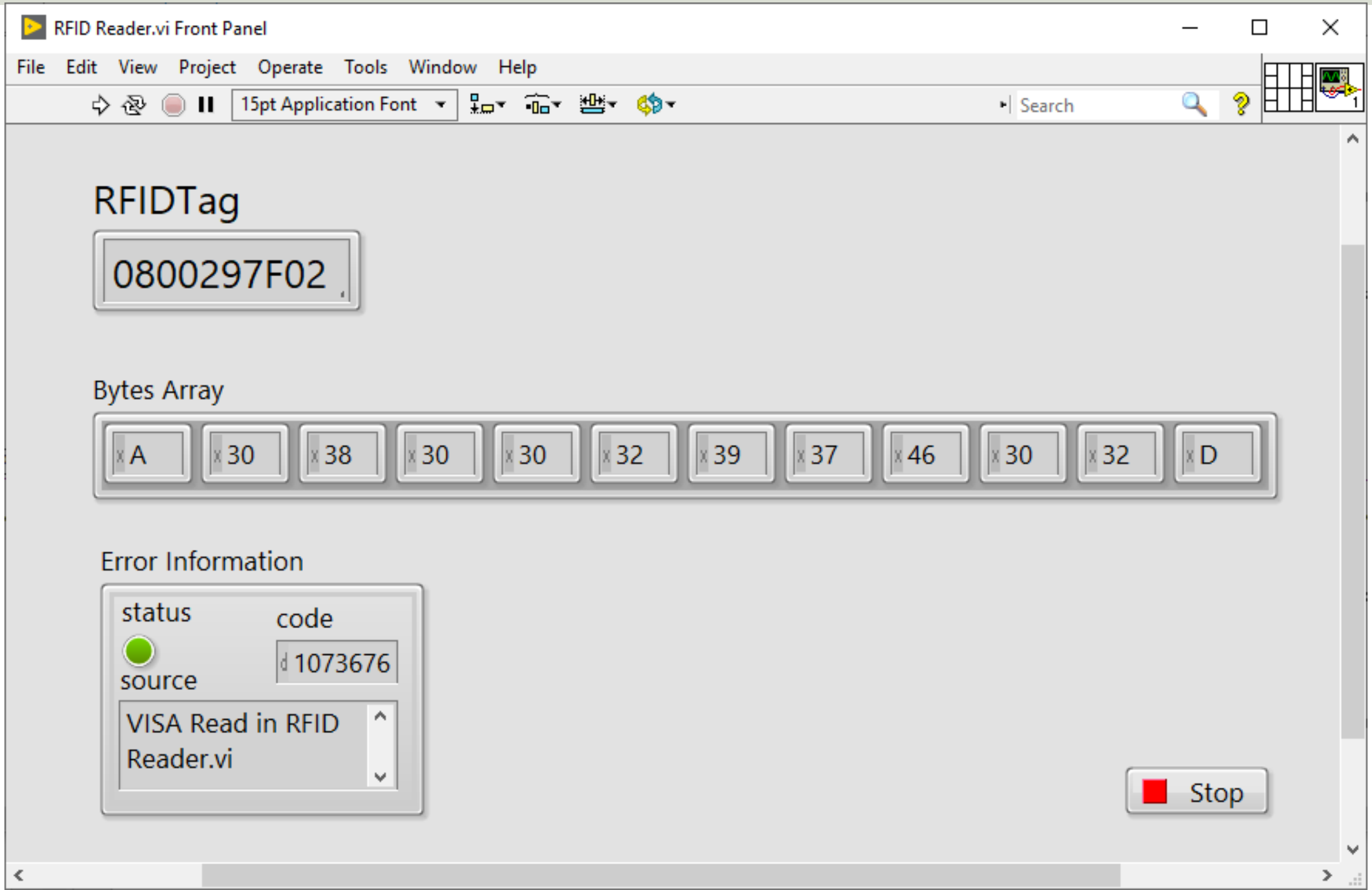
rfid_ex.py

⊞ Line 2 : Unused import time

Was it helpful or confusing?

```
b'\n0800296663\r'
b'\n0800296663\r'
b'\n0800296663\r'
b'\n0800296663\r'
b'\n0800296663\r'
b'\n0800296663\r'
b'\n0800296663\r'
b'\n0800296663\r'
b'\n0800296663\r'
b'\n080029C06C\r'
```

RFID LabVIEW Prototype


RFID Reader.vi Front Panel

RFIDTag

0800297F02

Bytes Array

A 30 38 30 30 32 39 37 46 30 32 D

Error Information

status

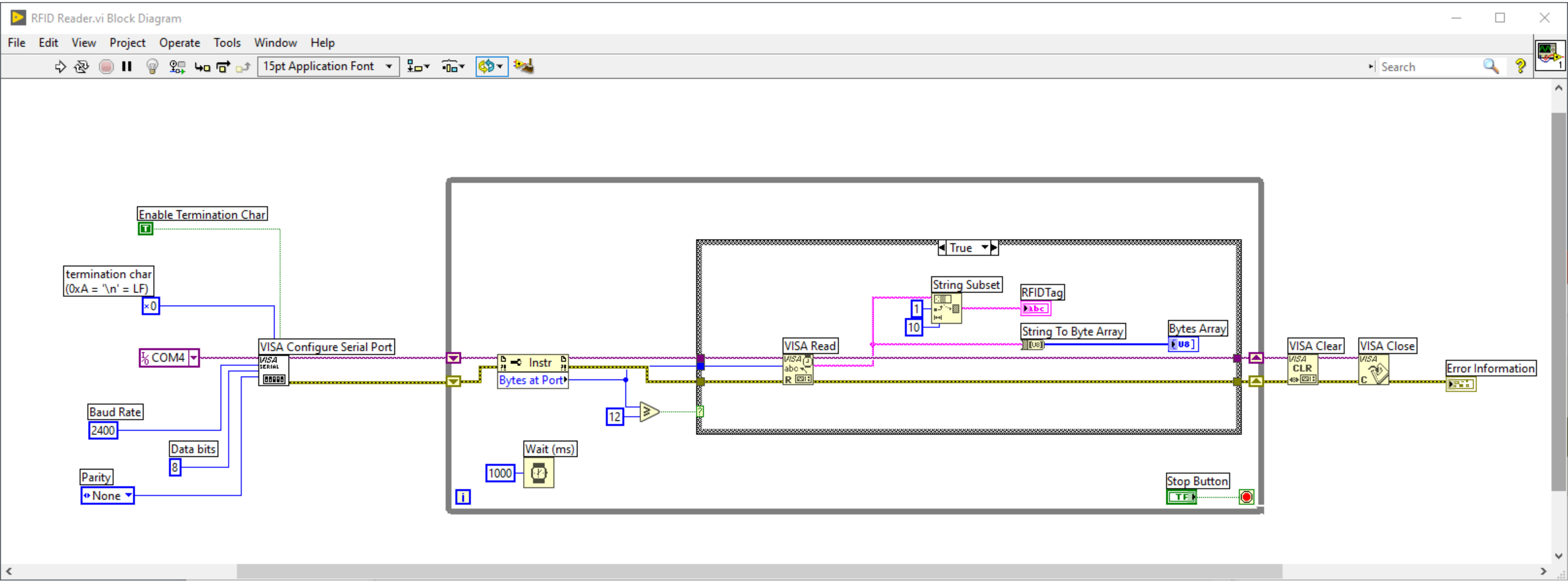
code

1073676

source

VISA Read in RFID Reader.vi

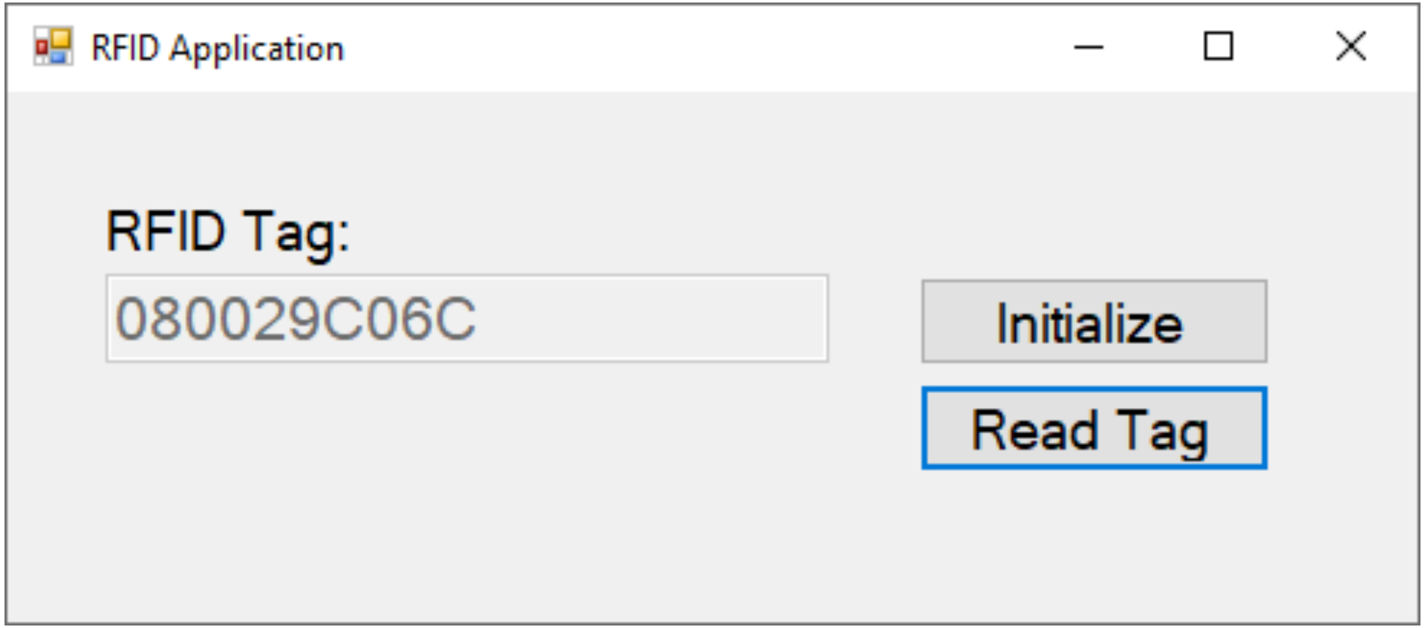
RFID LabVIEW Prototype



Read RFID Tag with C#

```
using System.IO.Ports;

SerialPort port = new System.IO.Ports.SerialPort("COM4", 2400, System.IO.Ports.Parity.None,
                                                             8, System.IO.Ports.StopBits.One);

port.Open();
port.DtrEnable = true;

int numberBytesToRead = 12;
byte[] data = new byte[numberBytesToRead];
port.ReadTimeout = 1000;
port.Read(data, 0, numberBytesToRead);

string rfidTag;
rfidTag = System.Text.Encoding.UTF8.GetString(data, 0, data.Length);

rfidTag = rfidTag.Replace("\n", "");
rfidTag = rfidTag.Replace("\r", "");

port.Close();
```

RFID C# Prototype


RFID Application

RFID Tag:

080029C06C

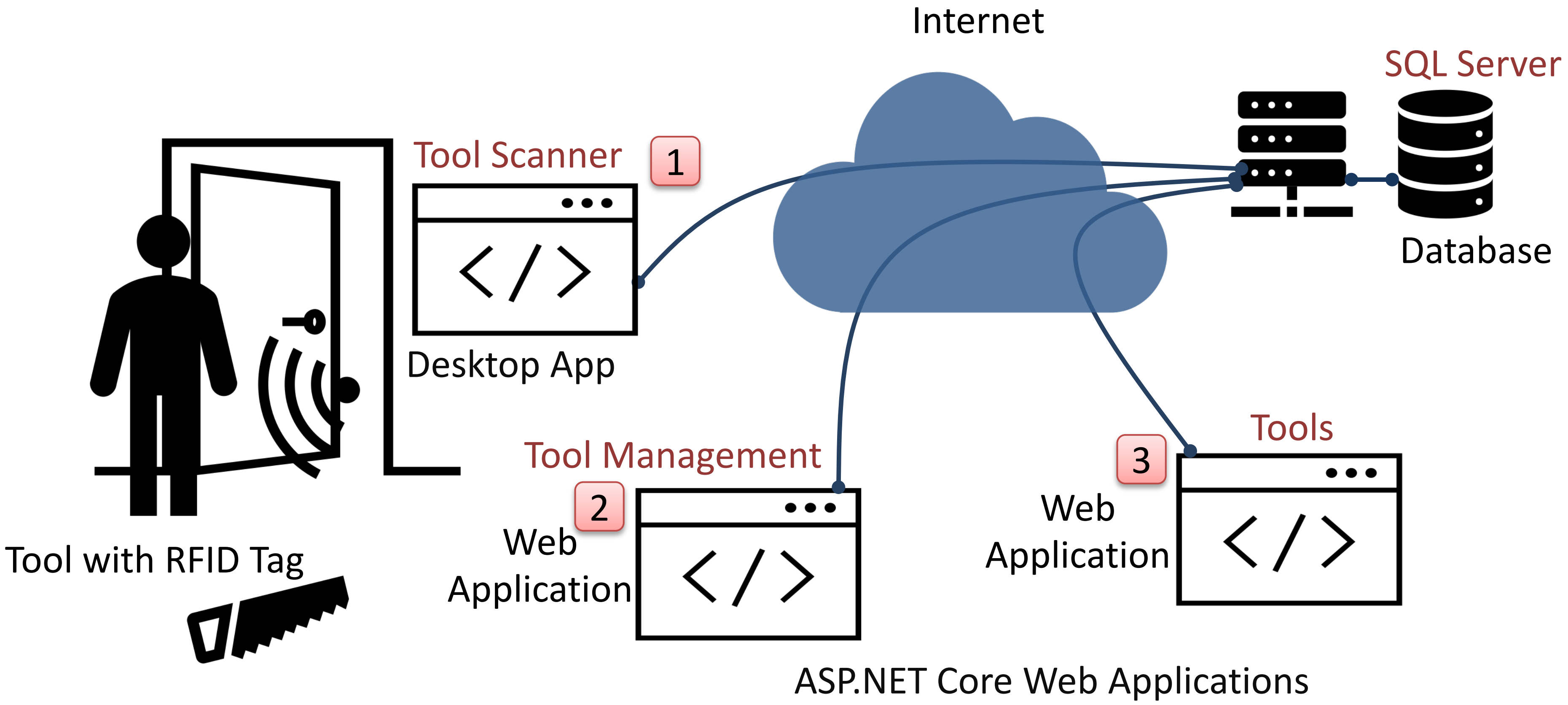
RFID C# Prototype

```
using System;
using System.IO.Ports;
using System.Windows.Forms;


namespace ReadRfidApp
{
    public partial class Form1 : Form
    {
        string rfidTag;
        SerialPort port = new System.IO.Ports.SerialPort("COM4", 2400, System.IO.Ports.Parity.None, 8, System.IO.Ports.StopBits.One);

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        { }

        private void btnInitialize_Click(object sender, EventArgs e)
        {
            port.Open();
            port.DtrEnable = true;

            txtTagData.Text = "";
        }

        private void btnReadTag_Click(object sender, EventArgs e)
        {
            int numberBytesToRead = 12;
            byte[] data = new byte[numberBytesToRead];
            port.ReadTimeout = 1000;
            port.Read(data, 0, numberBytesToRead);

            rfidTag = System.Text.Encoding.UTF8.GetString(data, 0, data.Length);

            rfidTag = rfidTag.Replace("\n", "");
            rfidTag = rfidTag.Replace("\r", "");

            txtTagData.Text = rfidTag;

            port.Close();
        }
    }
}
```

Applications

The applications are basic CRUD applications implemented in C# (WinForm and ASP.NET Core)

Inventory System - Applications

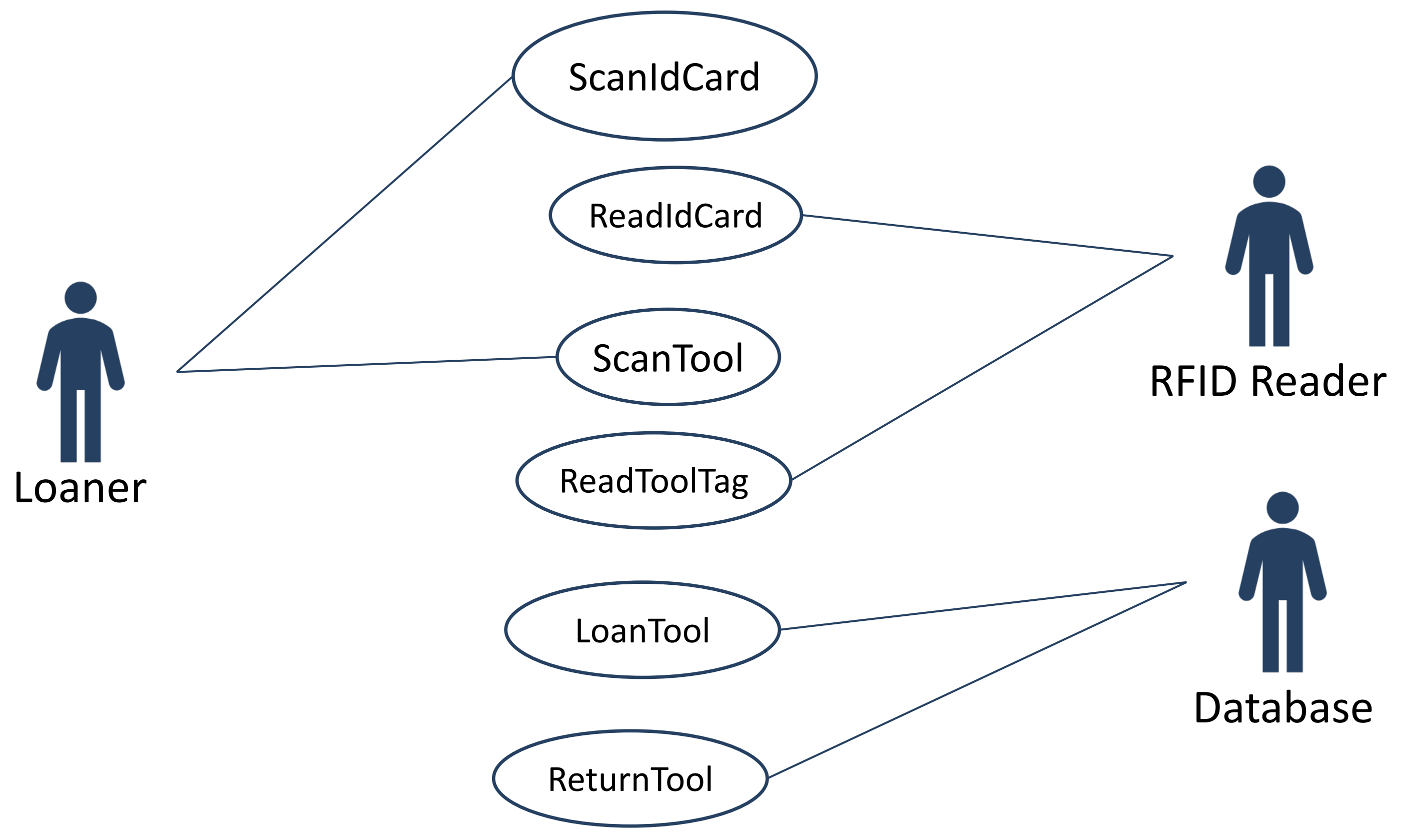


https://www.halvorsen.blog

Tool Scanner (App #1)

Hans-Petter Halvorsen

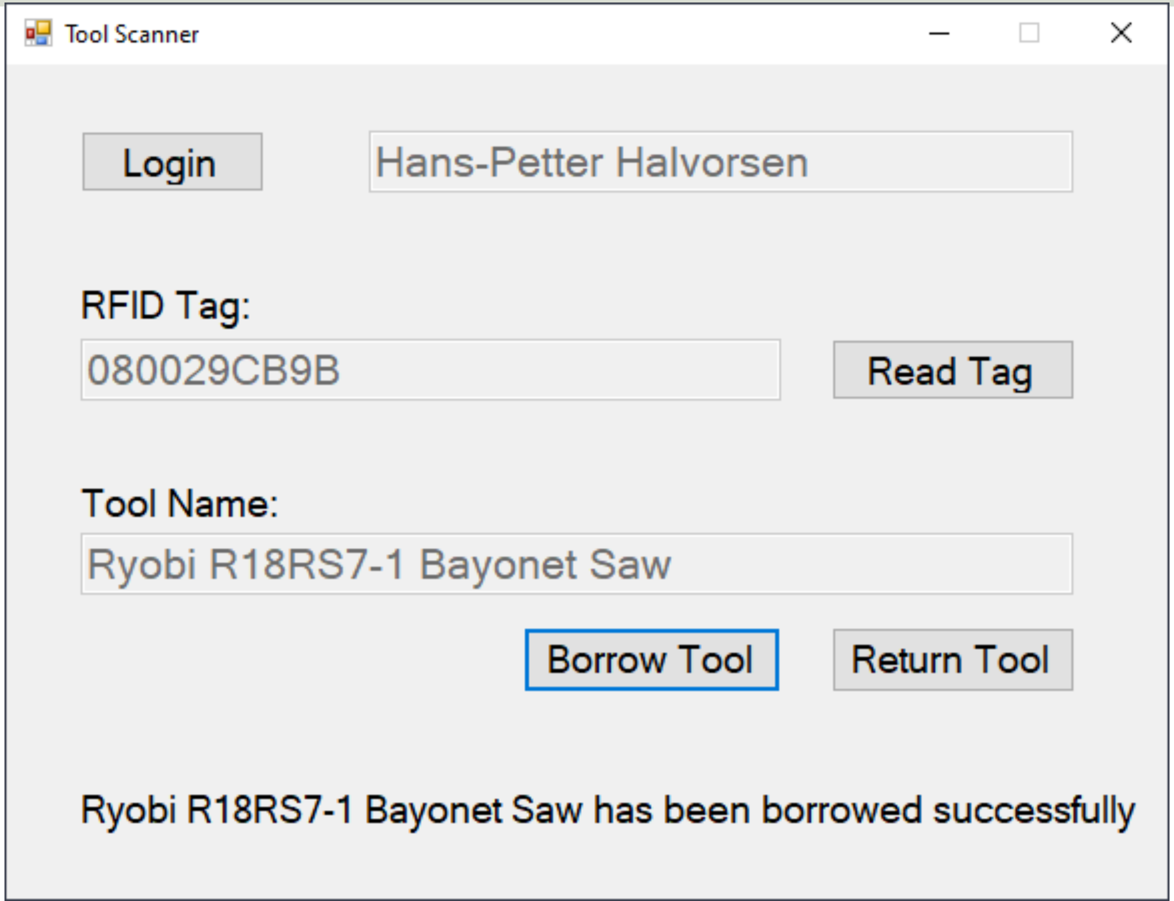
Use Case Diagram



Tool Scanner (App #1)

C# WinForm Desktop Application

The User scan ID Card and Tool with RFID Tag when going out of the Inventory door



Hans-Petter Halvorsen

RFID Tag: 080029CB9B

Tool Name: Ryobi R18RS7-1 Bayonet Saw

Ryobi R18RS7-1 Bayonet Saw has been borrowed successfully

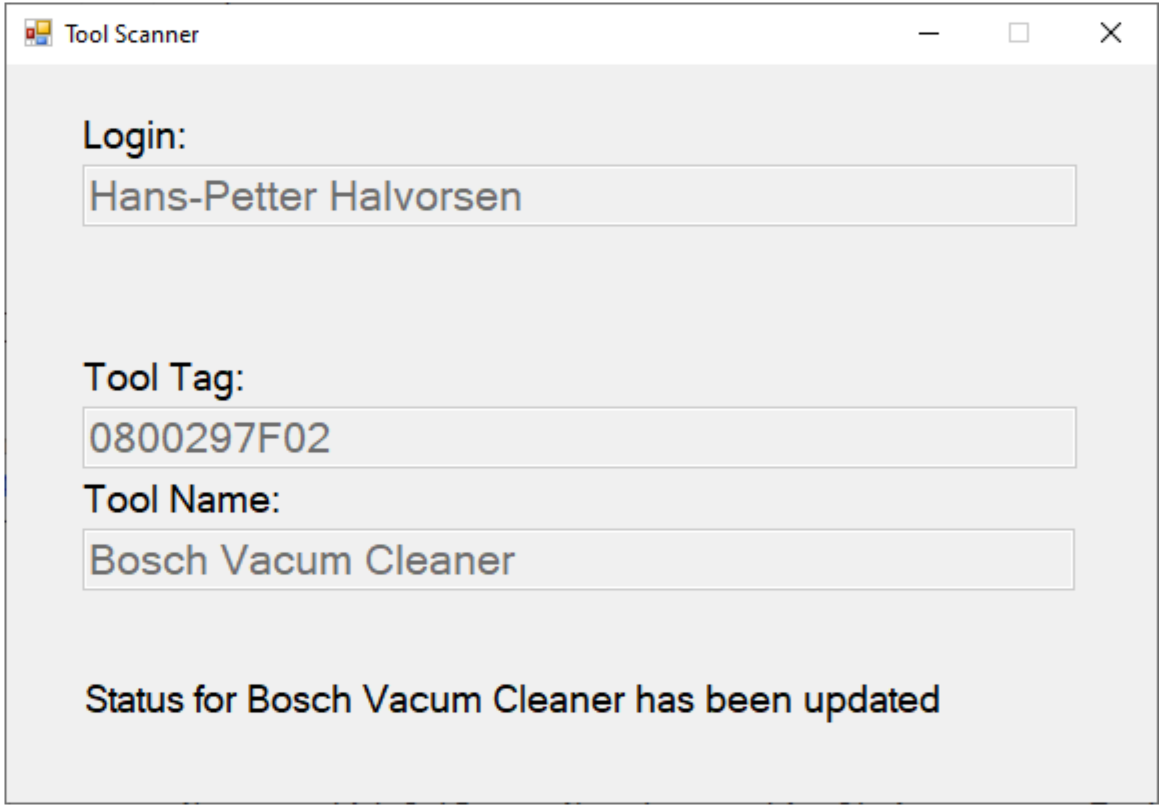
Tool Scanner (App #1) – Improved!

C# WinForm Desktop Application

The Tool Scanner Application automatically scans the User ID Card and the Tool he wants to borrow when the person is leaving the Inventory room through the door.

No manual steps are required by the user!

When the user return with the Tool, the system will be automatically updated



Login:
Hans-Petter Halvorsen

Tool Tag:
0800297F02

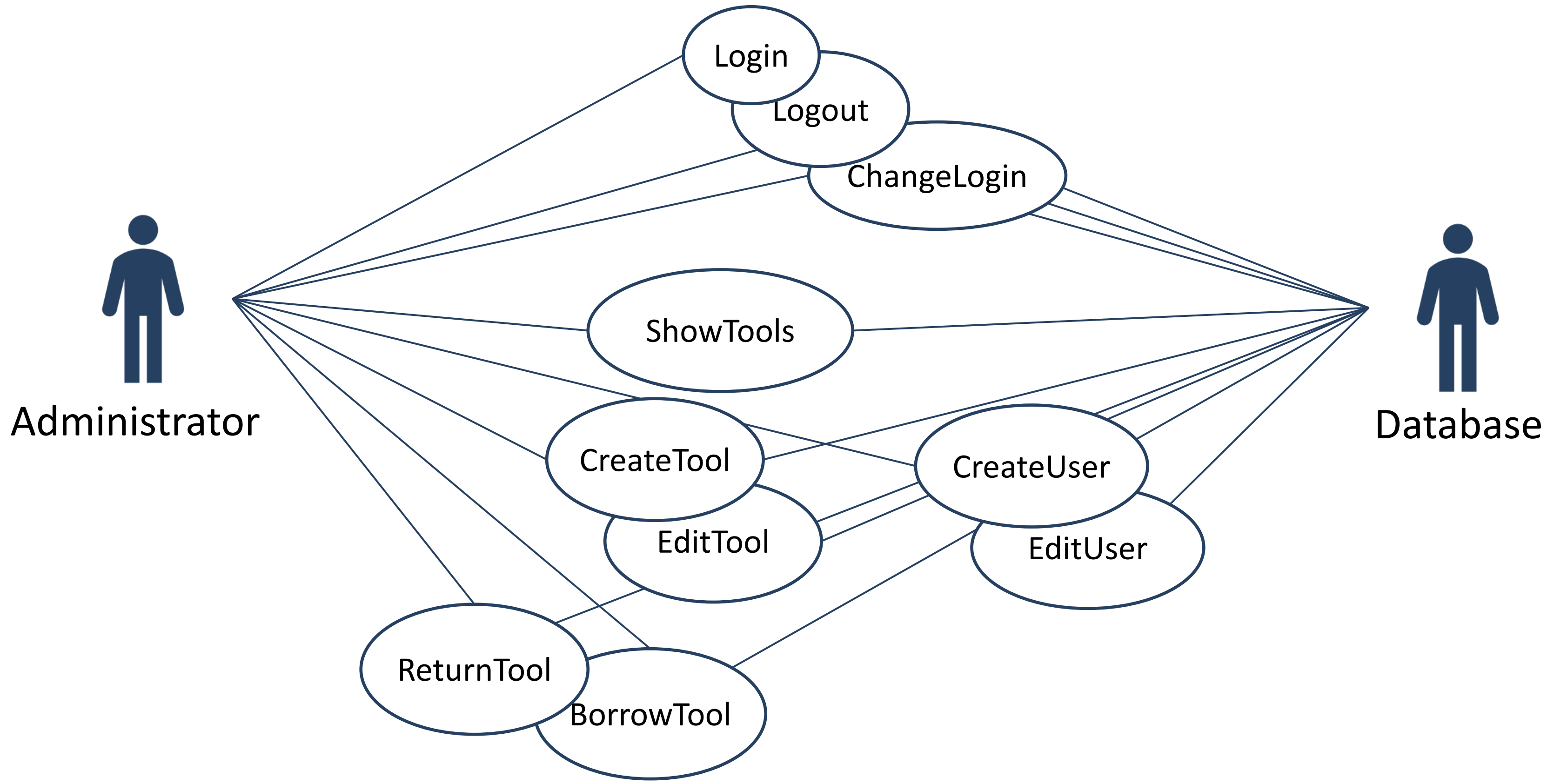
Tool Name:
Bosch Vacum Cleaner

Status for Bosch Vacum Cleaner has been updated

Tool Management (App #2)

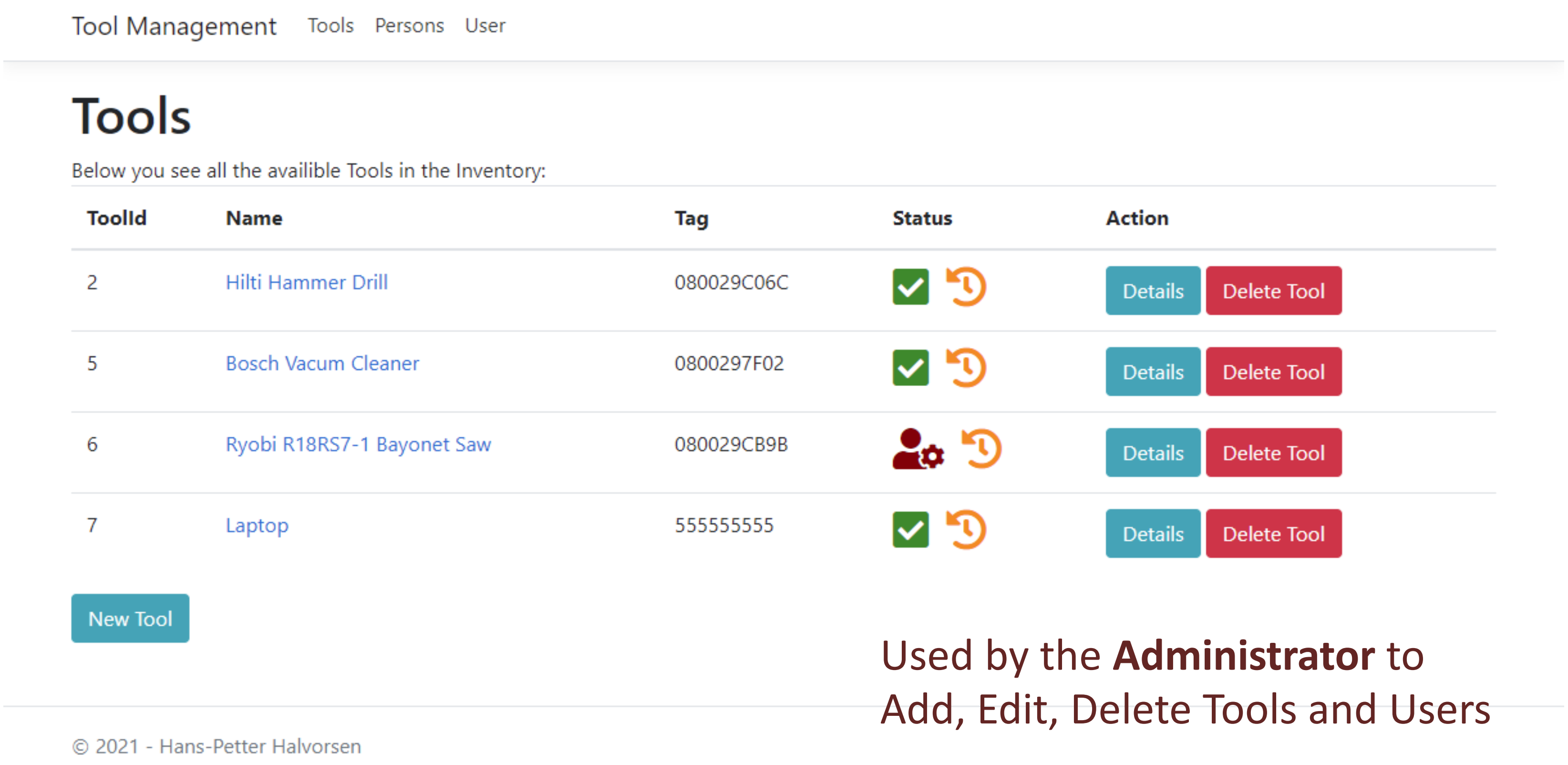
Hans-Petter Halvorsen

Use Case Diagram



Tool Management (App #2)

ASP.NET Core Web Application



Tool Management Tools Persons User

Tools

Below you see all the availible Tools in the Inventory:

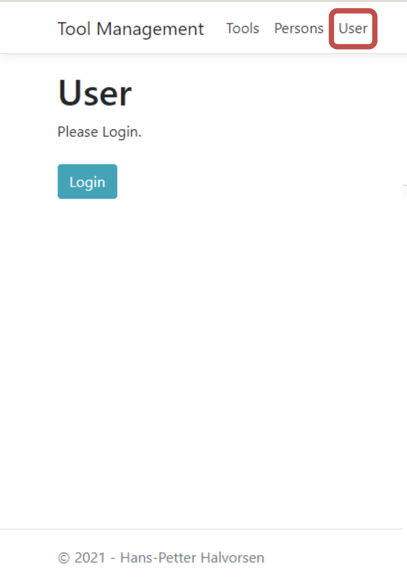
ToolId	Name	Tag	Status	Action
2	Hilti Hammer Drill	080029C06C	✓	Details Delete Tool
5	Bosch Vacum Cleaner	0800297F02	✓	Details Delete Tool
6	Ryobi R18RS7-1 Bayonet Saw	080029CB9B		Details Delete Tool
7	Laptop	555555555	✓	Details Delete Tool

New Tool

© 2021 - Hans-Petter Halvorsen

Used by the **Administrator** to Add, Edit, Delete Tools and Users

Login



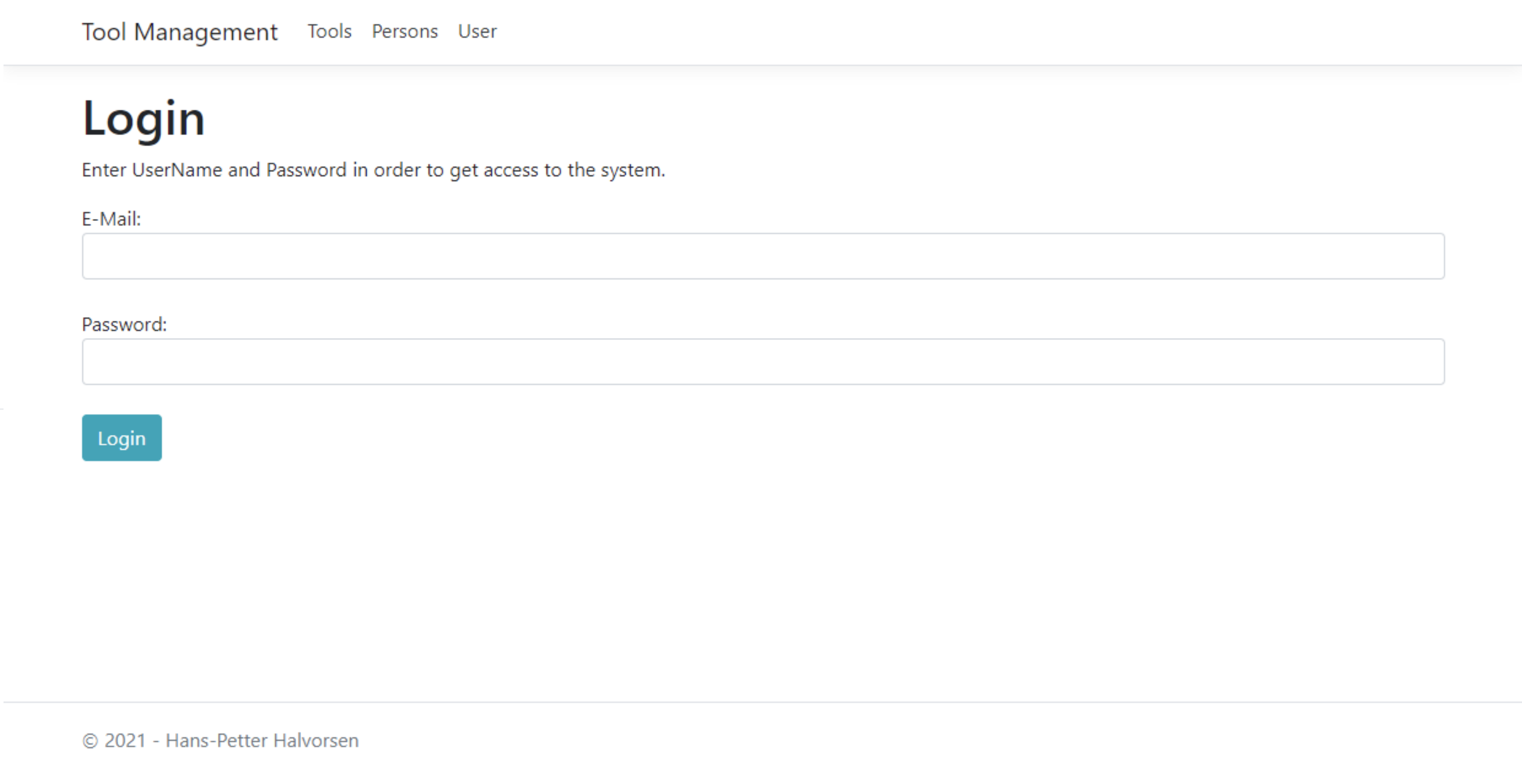
Tool Management Tools Persons User

User

Please Login.

Login

© 2021 - Hans-Petter Halvorsen



Tool Management Tools Persons User

Login

Enter UserName and Password in order to get access to the system.

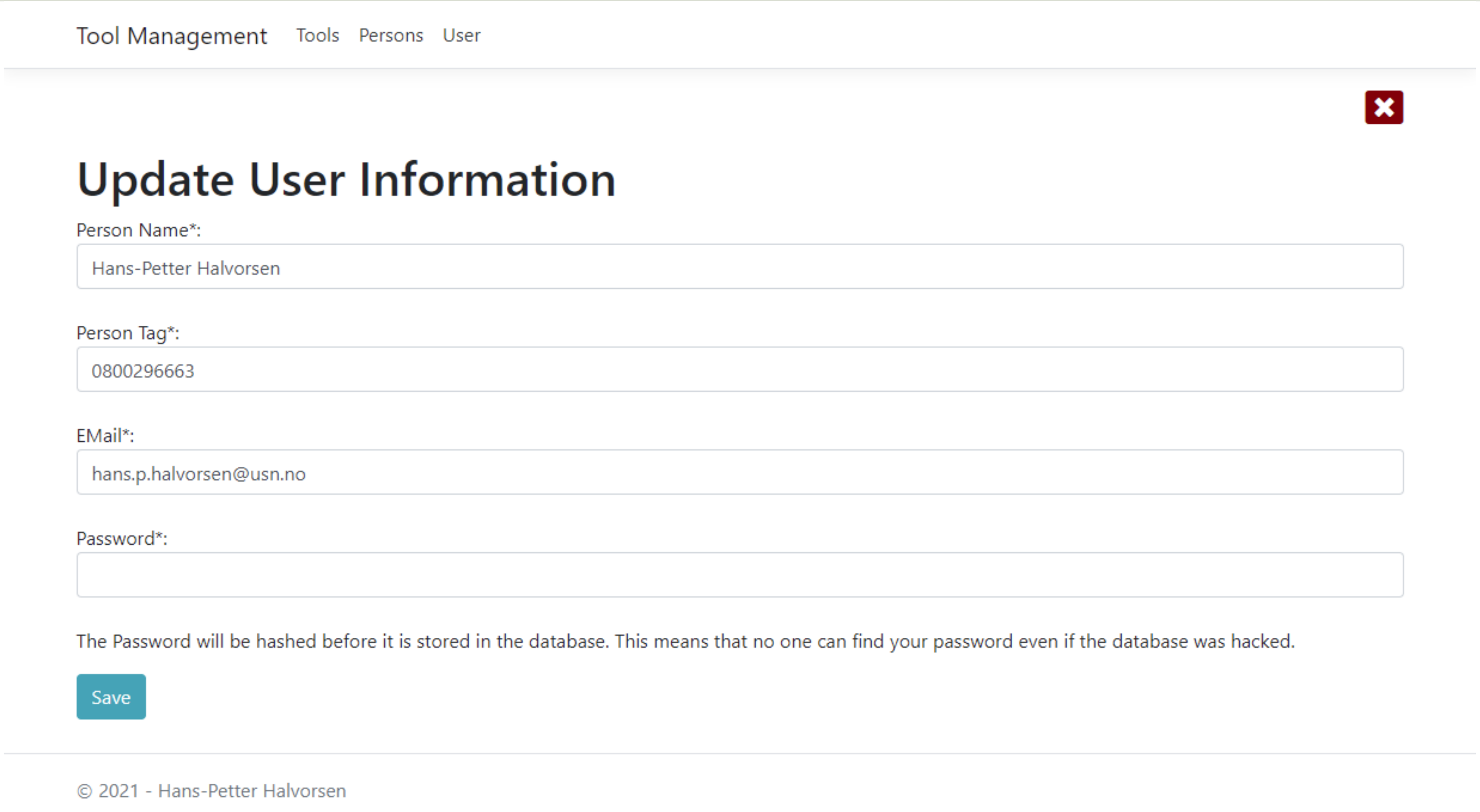
E-Mail:

Password:

Login

© 2021 - Hans-Petter Halvorsen

Update User Information



Tool Management Tools Persons User

Update User Information

Person Name*:

Hans-Petter Halvorsen

Person Tag*:

0800296663

EMail*:

hans.p.halvorsen@usn.no

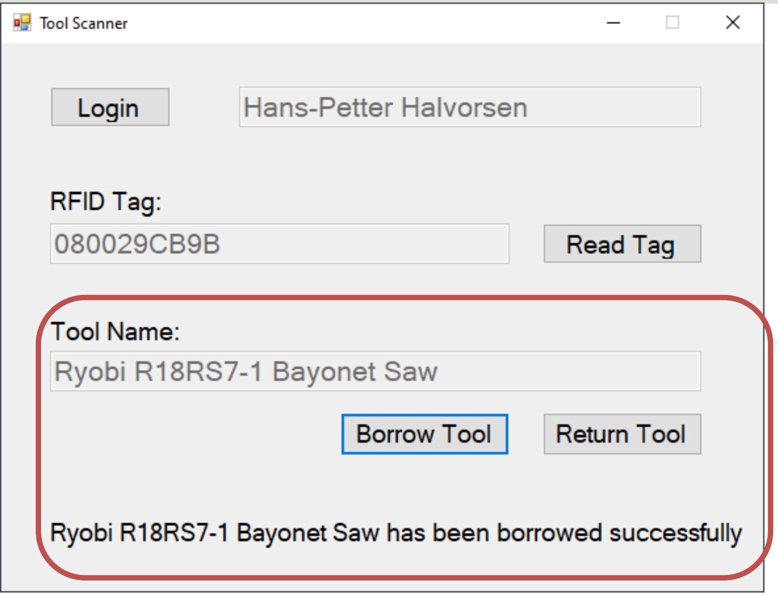
Password*:

The Password will be hashed before it is stored in the database. This means that no one can find your password even if the database was hacked.

Save

© 2021 - Hans-Petter Halvorsen

Borrow Tool



Hans-Petter Halvorsen

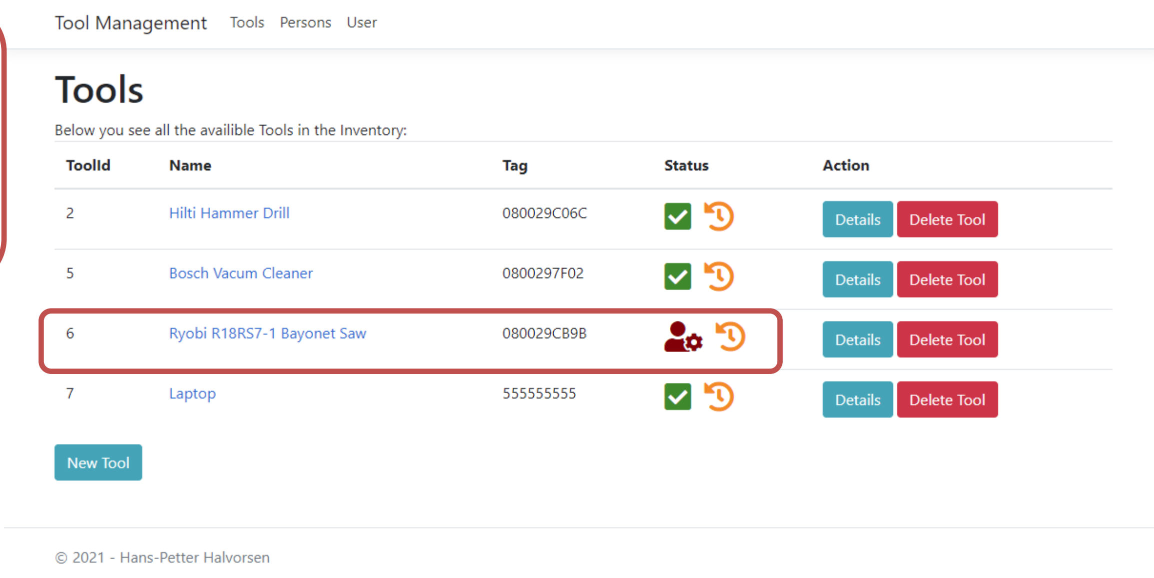
RFID Tag:

080029CB9B

Tool Name:

Ryobi R18RS7-1 Bayonet Saw

Ryobi R18RS7-1 Bayonet Saw has been borrowed successfully



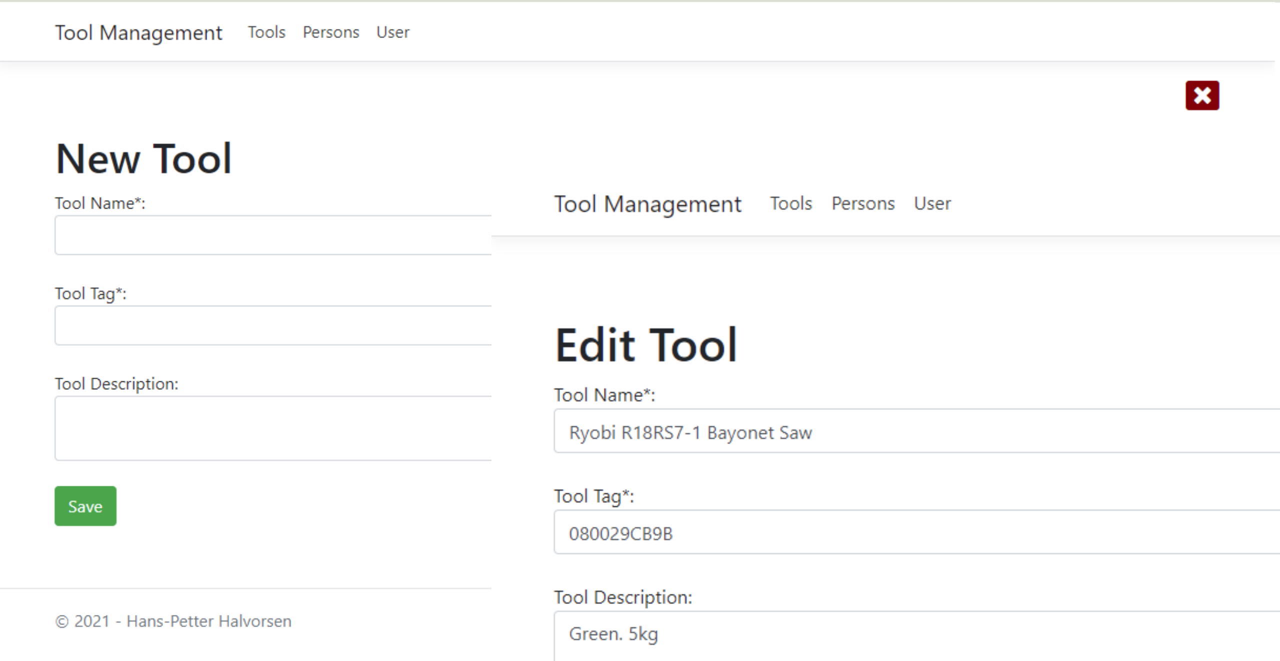
Tools

Below you see all the availible Tools in the Inventory:

ToolId	Name	Tag	Status	Action
2	Hilti Hammer Drill	080029C06C	✓	Details Delete Tool
5	Bosch Vacum Cleaner	0800297F02	✓	Details Delete Tool
6	Ryobi R18RS7-1 Bayonet Saw	080029CB9B		Details Delete Tool
7	Laptop	555555555	✓	Details Delete Tool

© 2021 - Hans-Petter Halvorsen

Tool Management



Tool Management Tools Persons User

New Tool

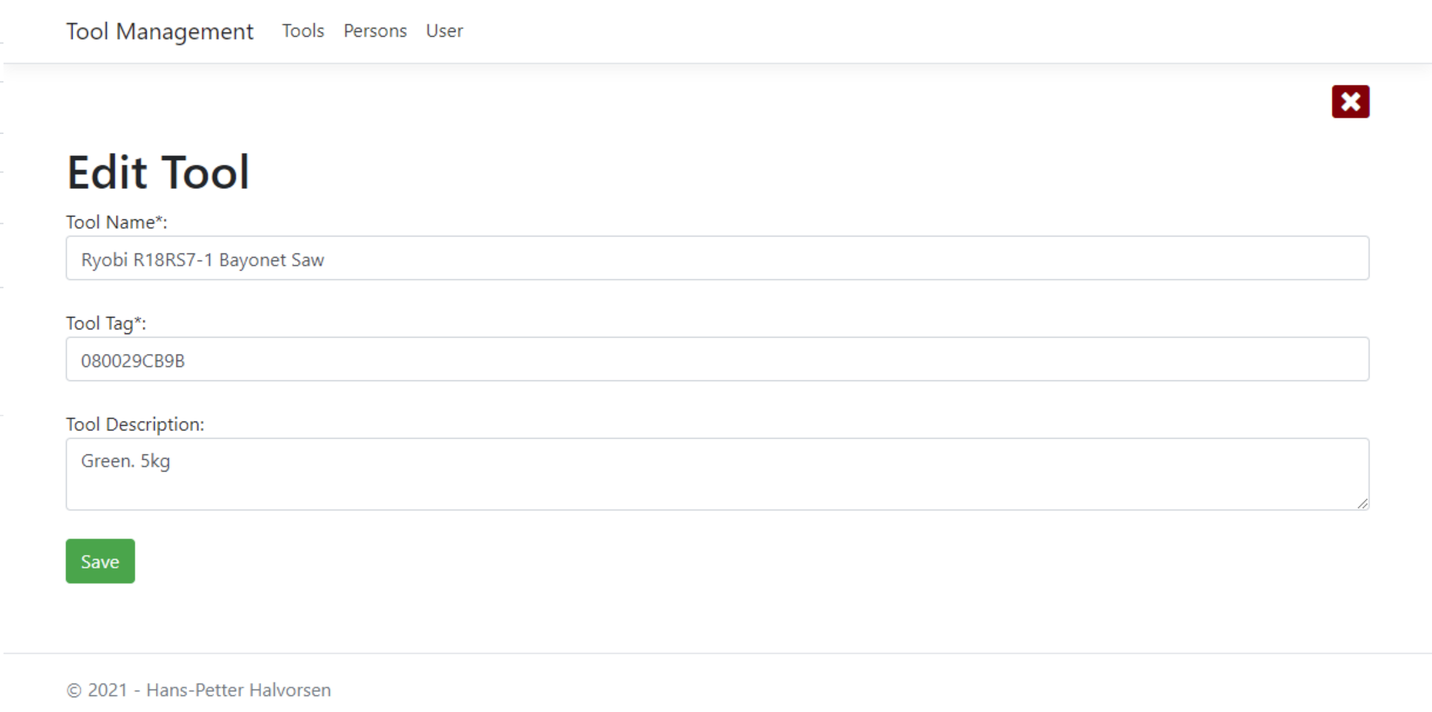
Tool Name*:

Tool Tag*:

Tool Description:

Save

© 2021 - Hans-Petter Halvorsen



Tool Management Tools Persons User

Edit Tool

Tool Name*: Ryobi R18RS7-1 Bayonet Saw

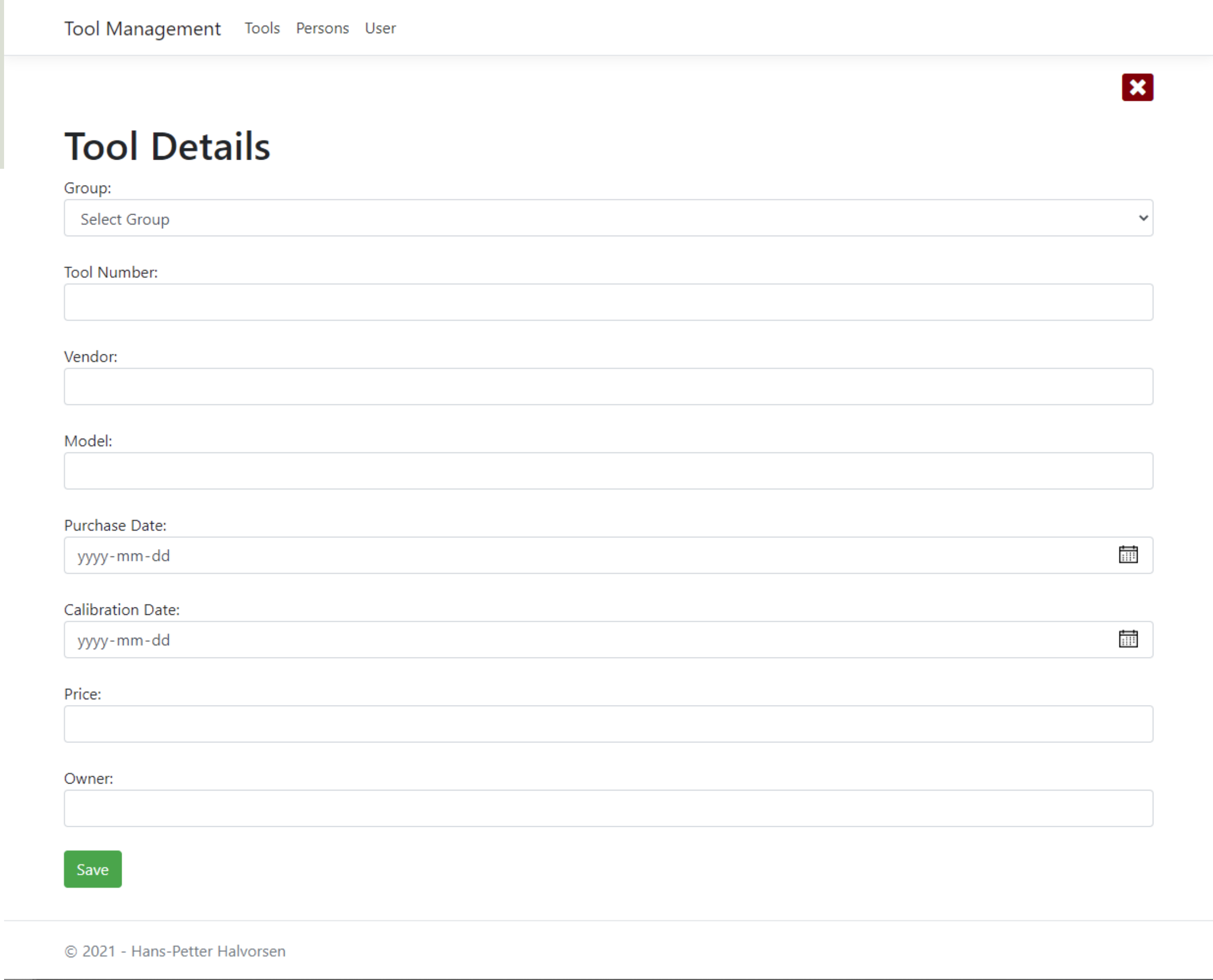
Tool Tag*: 080029CB9B

Tool Description: Green. 5kg

Save

© 2021 - Hans-Petter Halvorsen

Tool Details


Tool Management Tools Persons User

Tool Details

Group:
Select Group

Tool Number:

Vendor:

Model:

Purchase Date:
yyyy-mm-dd

Calibration Date:
yyyy-mm-dd

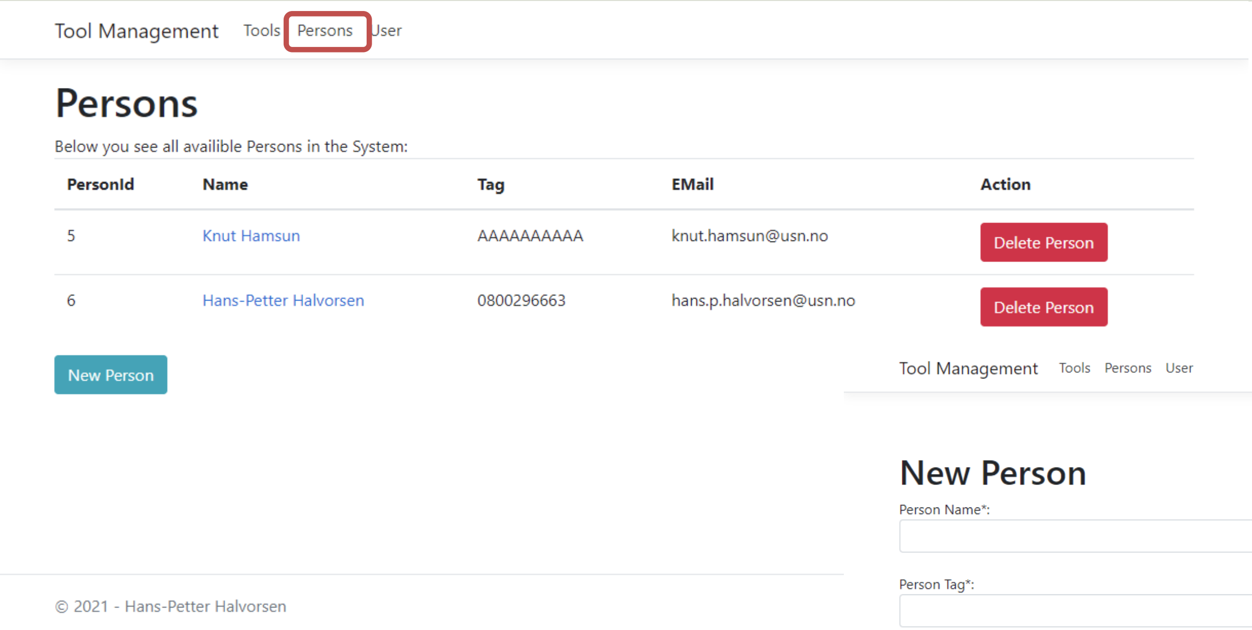
Price:

Owner:

Save

© 2021 - Hans-Petter Halvorsen

User/Person Management



Tool Management Tools Persons User

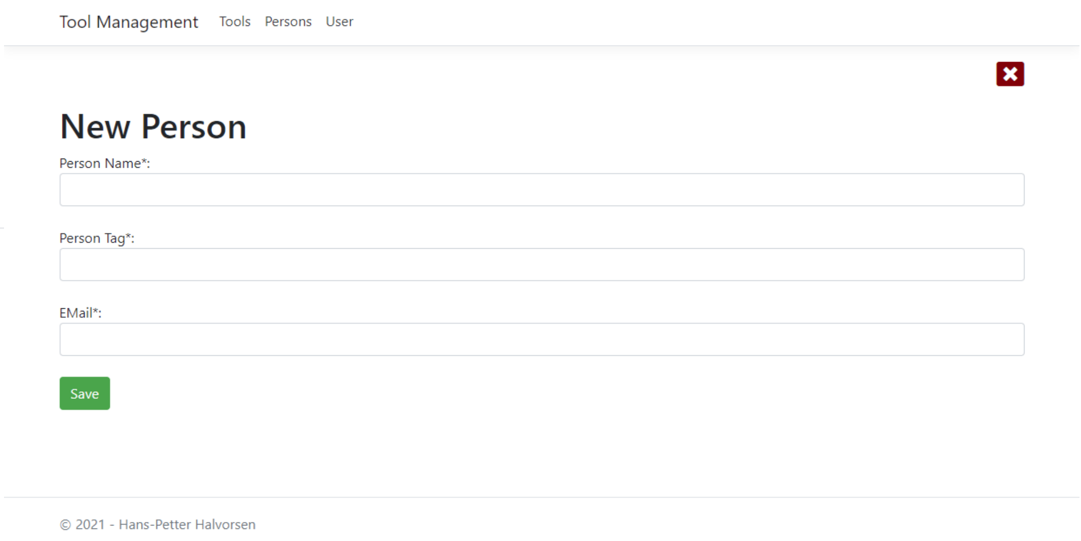
Persons

Below you see all availible Persons in the System:

PersonId	Name	Tag	EMail	Action
5	Knut Hamsun	AAAAAAAAAA	knut.hamsun@usn.no	Delete Person
6	Hans-Petter Halvorsen	0800296663	hans.p.halvorsen@usn.no	Delete Person

New Person

© 2021 - Hans-Petter Halvorsen



Tool Management Tools Persons User

New Person

Person Name*:

Person Tag*:

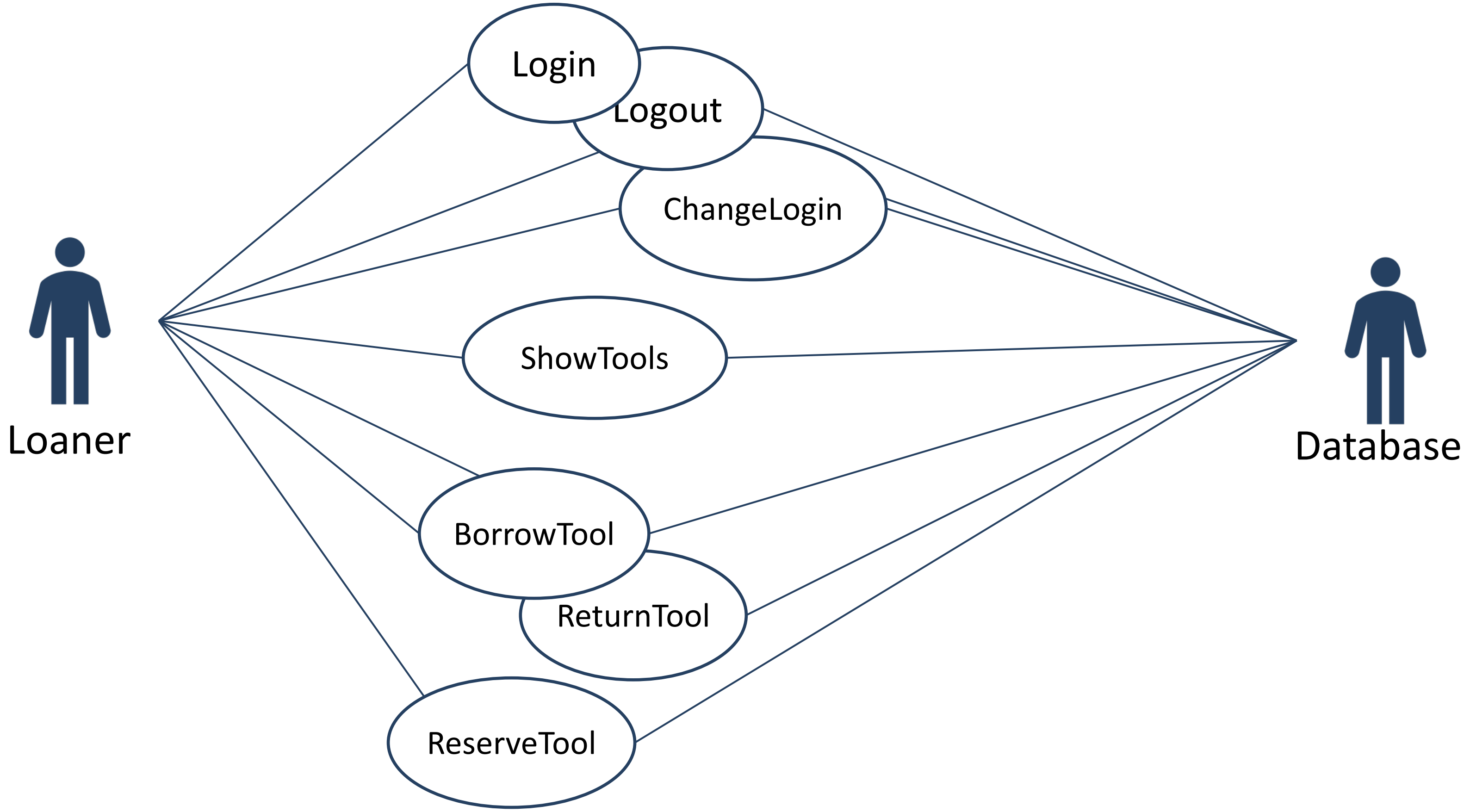
EMail*:

Save

© 2021 - Hans-Petter Halvorsen

Tools
(App #3)

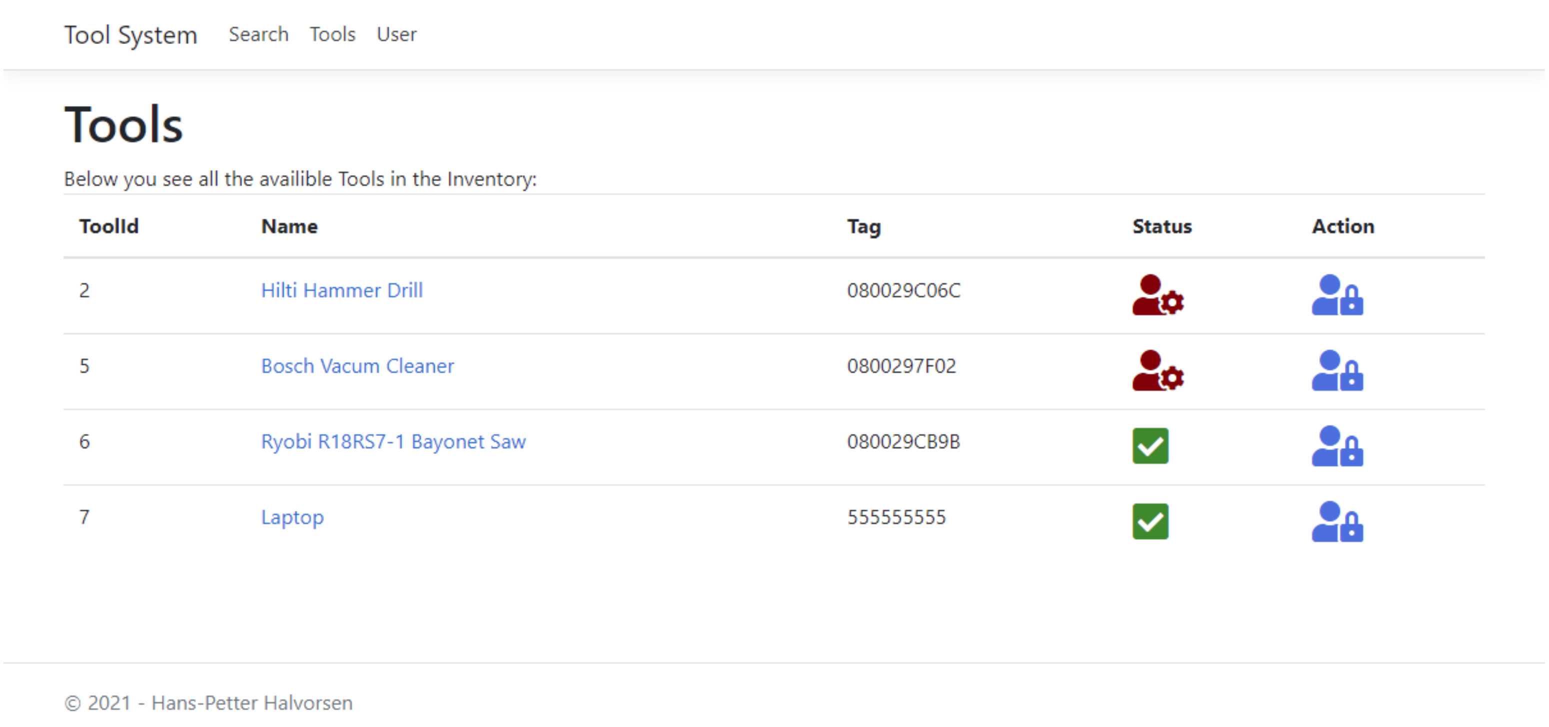
Use Case Diagram



Tools (App #3)

ASP.NET Core Web Application

The User can get an overview of available Tools in the Inventory



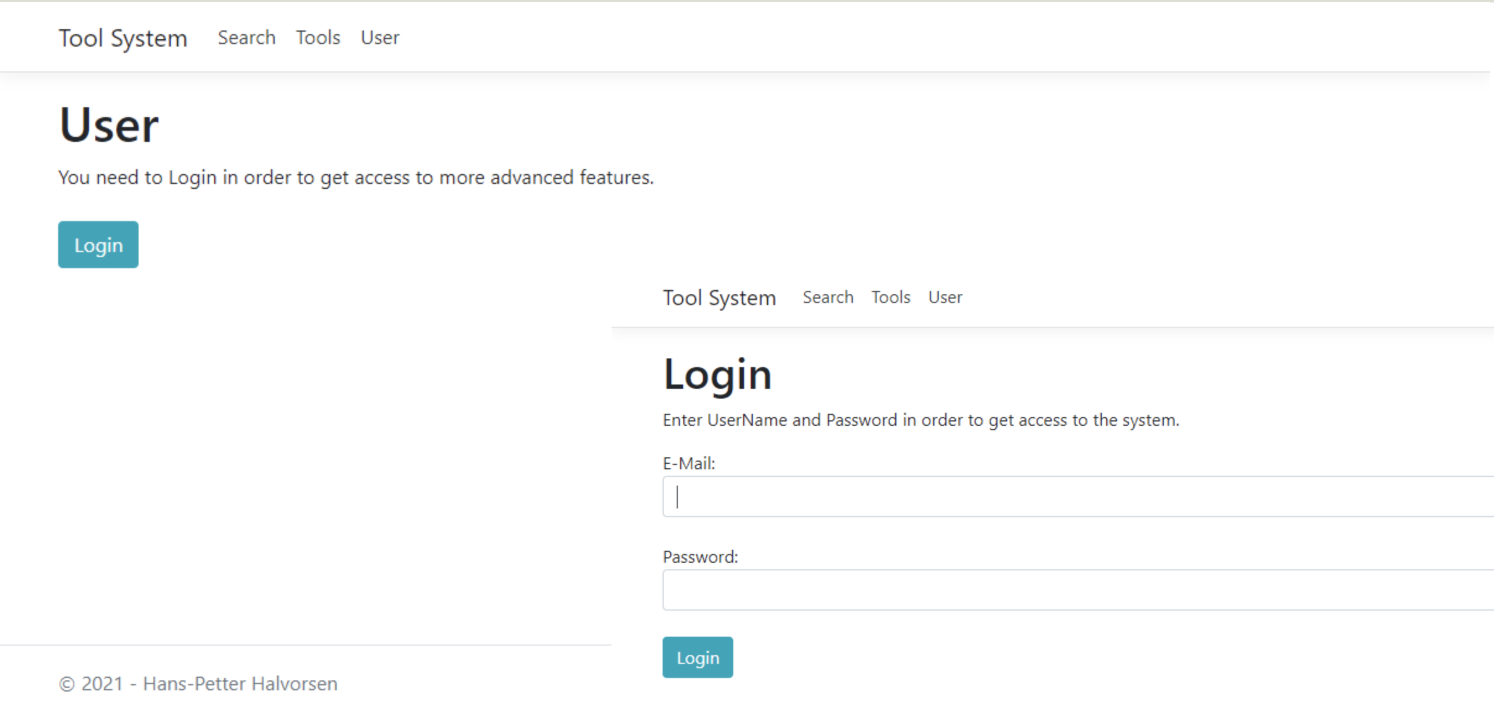
Tool System   Search   Tools   User

Tools

Below you see all the availible Tools in the Inventory:

ToolId	Name	Tag	Status	Action
2	Hilti Hammer Drill	080029C06C	👤⚙	👤🔒
5	Bosch Vacum Cleaner	0800297F02	👤⚙	👤🔒
6	Ryobi R18RS7-1 Bayonet Saw	080029CB9B	✓	👤🔒
7	Laptop	555555555	✓	👤🔒

Login



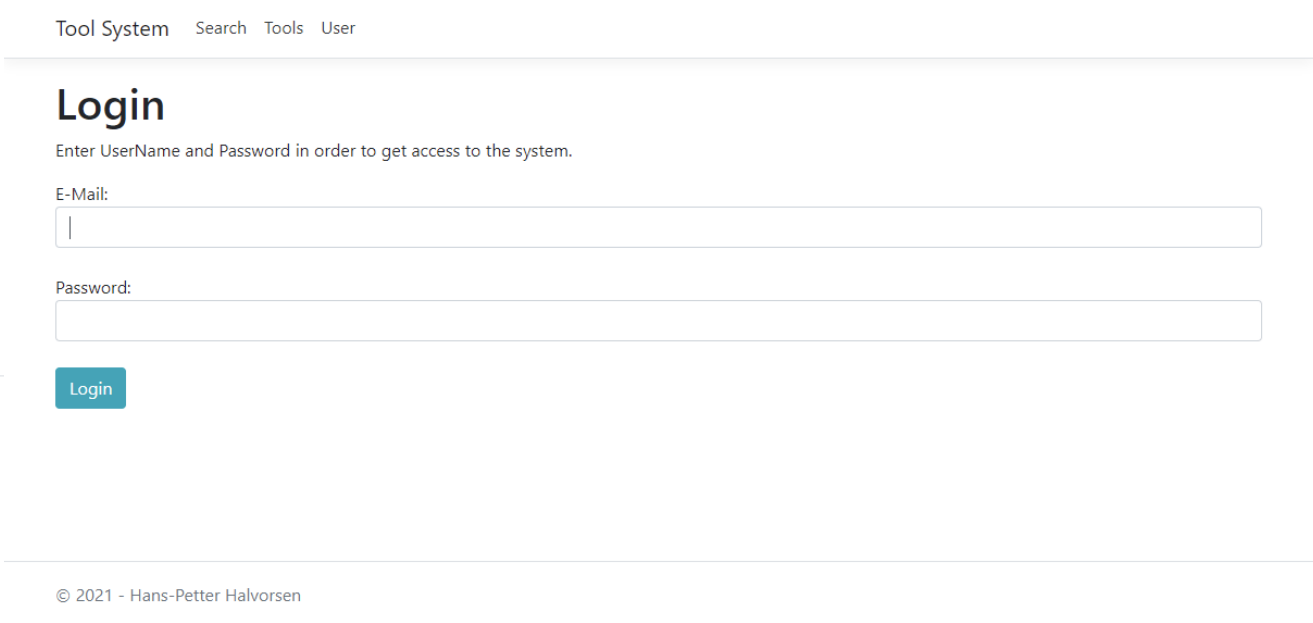
Tool System Search Tools User

User

You need to Login in order to get access to more advanced features.

Login

© 2021 - Hans-Petter Halvorsen



Tool System Search Tools User

Login

Enter UserName and Password in order to get access to the system.

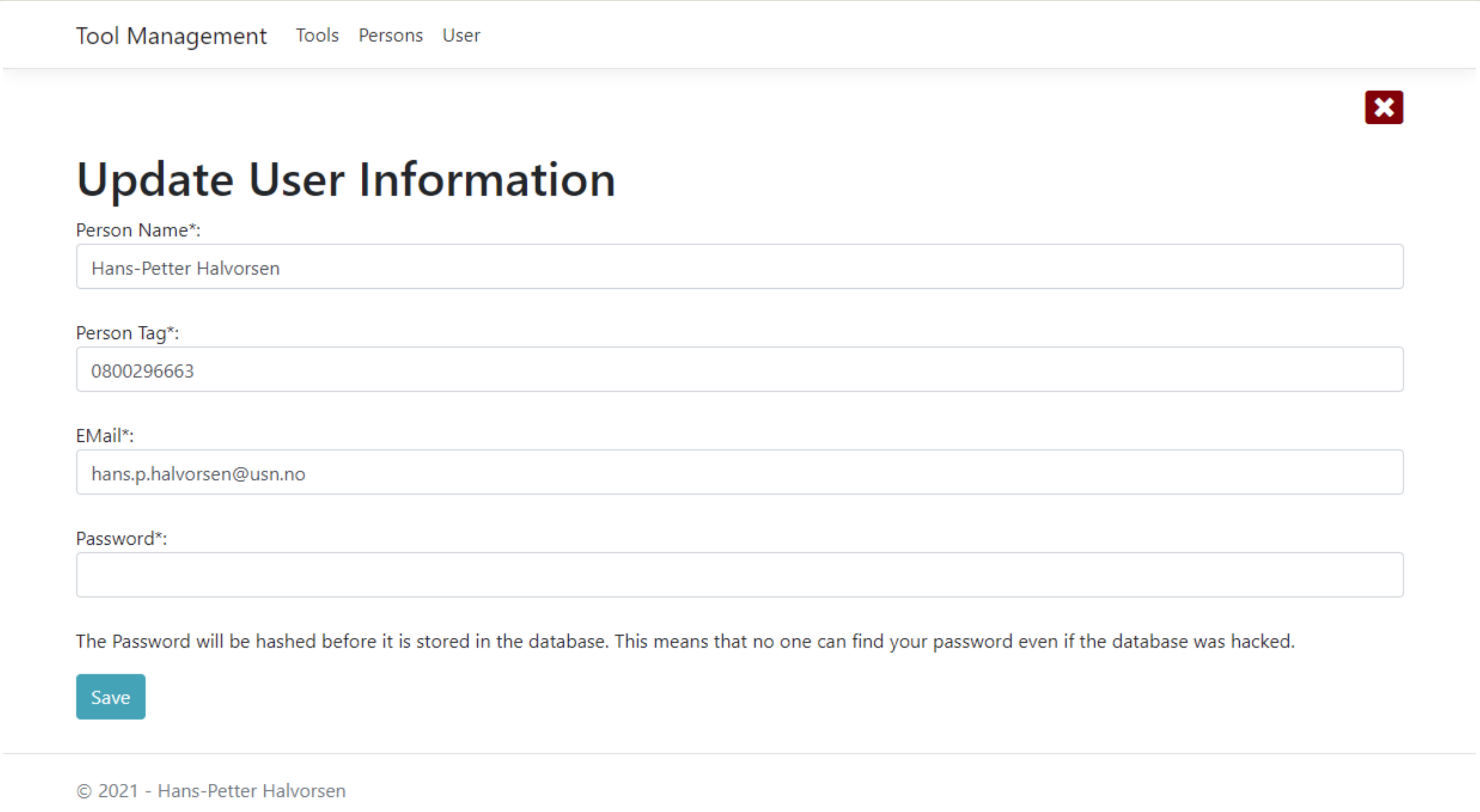
E-Mail:

Password:

Login

© 2021 - Hans-Petter Halvorsen

Update User Information



Update User Information

Person Name*:

Hans-Petter Halvorsen

Person Tag*:

0800296663

EMail*:

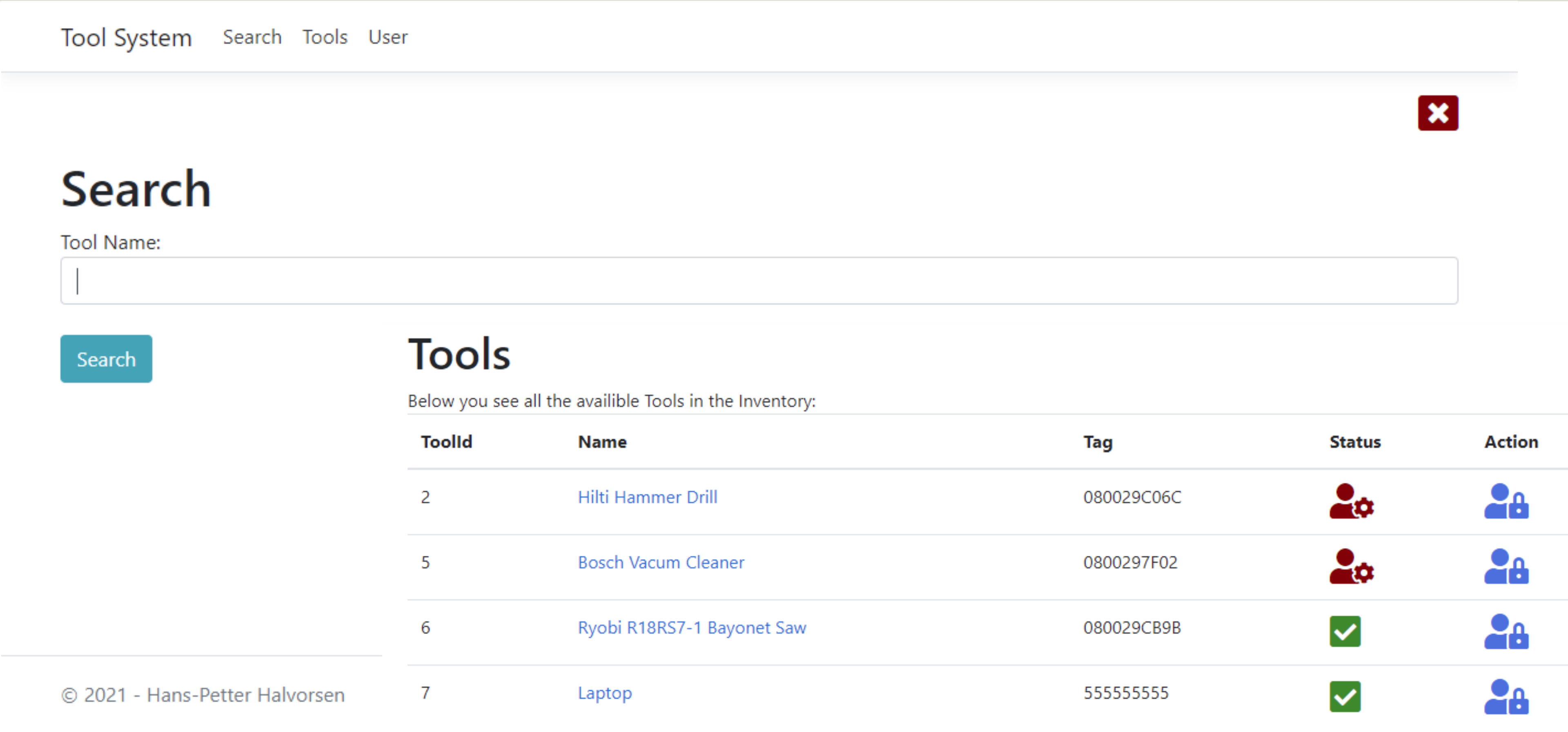
hans.p.halvorsen@usn.no

Password*:

The Password will be hashed before it is stored in the database. This means that no one can find your password even if the database was hacked.

© 2021 - Hans-Petter Halvorsen

Search



Search

Tool Name:

Search

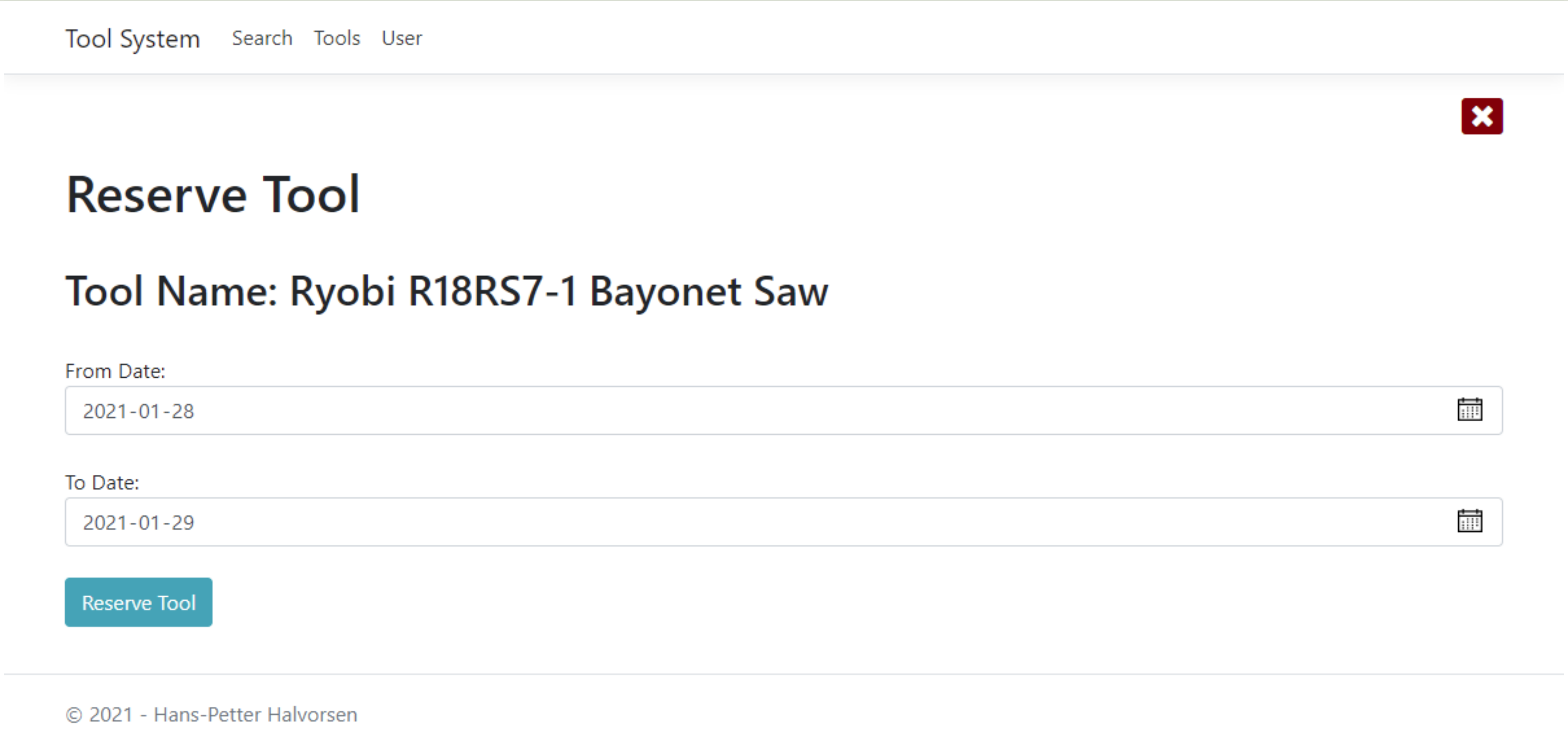
Tools

Below you see all the availible Tools in the Inventory:

ToolId	Name	Tag	Status	Action
2	Hilti Hammer Drill	080029C06C		
5	Bosch Vacum Cleaner	0800297F02		
6	Ryobi R18RS7-1 Bayonet Saw	080029CB9B	✓	
7	Laptop	555555555	✓	

© 2021 - Hans-Petter Halvorsen

Reserve Tool


Tool System Search Tools User

Reserve Tool

Tool Name: Ryobi R18RS7-1 Bayonet Saw

From Date:
2021-01-28

To Date:
2021-01-29

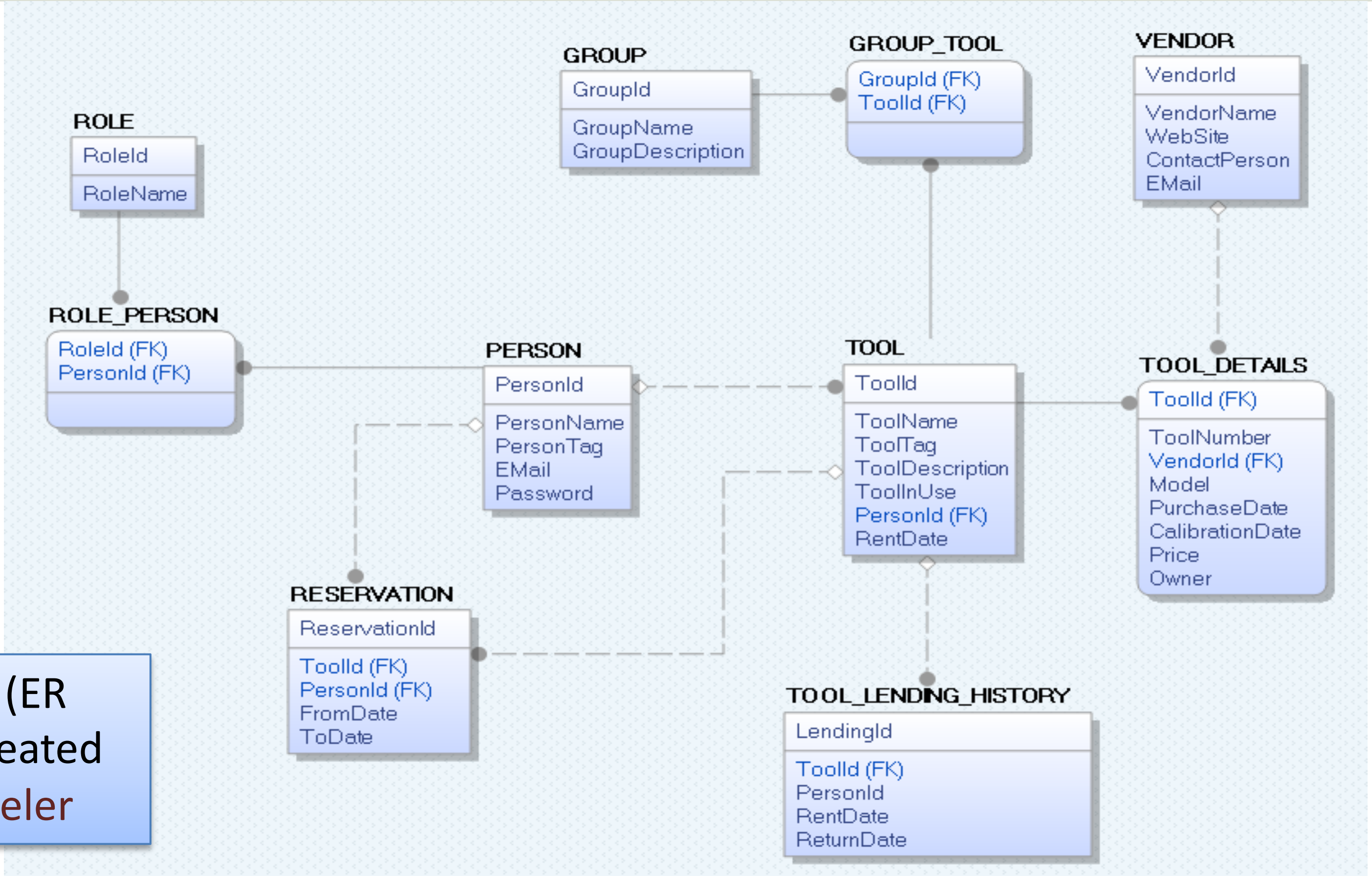
Reserve Tool

© 2021 - Hans-Petter Halvorsen

Database

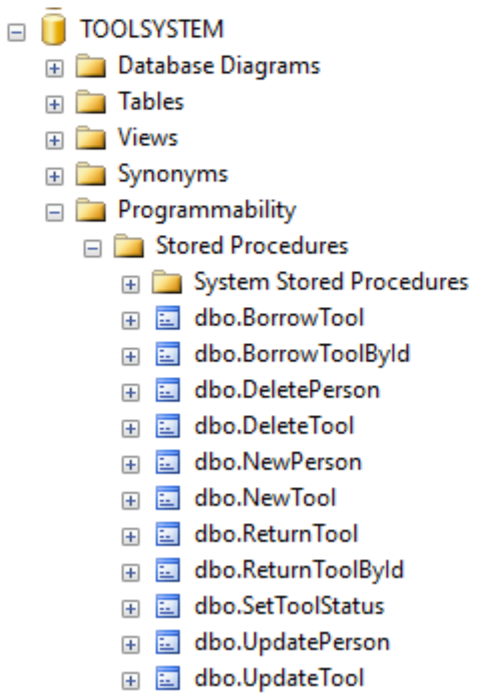
Database (ER Diagram)

SQL Server



The Database Design (ER diagram) has been created with erwin Data Modeler

Stored Procedures



TOOLSYSTEM
- Database Diagrams
- Tables
- Views
- Synonyms
- Programmability
  - Stored Procedures
    - System Stored Procedures
    - dbo.BorrowTool
    - dbo.BorrowToolById
    - dbo.DeletePerson
    - dbo.DeleteTool
    - dbo.NewPerson
    - dbo.NewTool
    - dbo.ReturnTool
    - dbo.ReturnToolById
    - dbo.SetToolStatus
    - dbo.UpdatePerson
    - dbo.UpdateTool

- NewPerson
- UpdatePerson
- DeletePerson

- NewTool
- DeleteTool
- UpdateTool

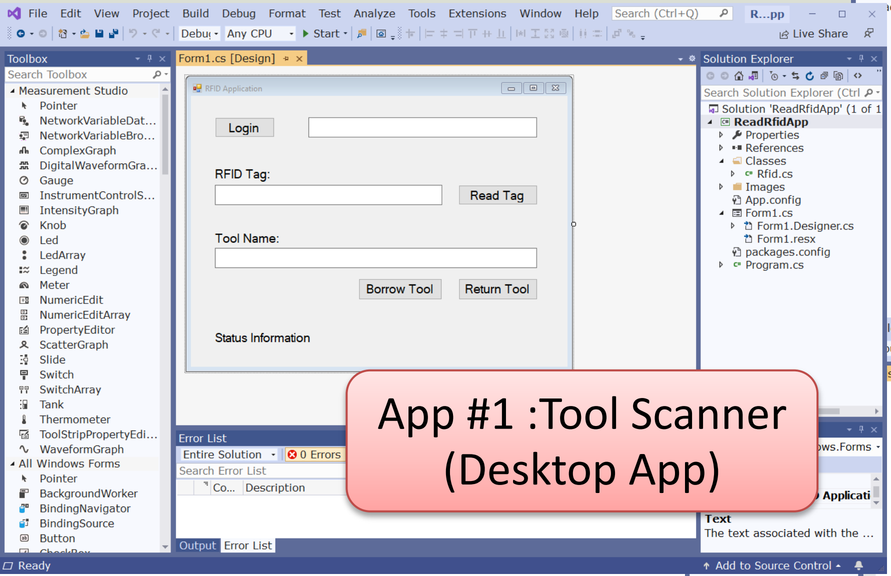
- BorrowTool
- BorrowToolById
- ReturnTool
- ReturnToolById
- SetToolStatus

Example

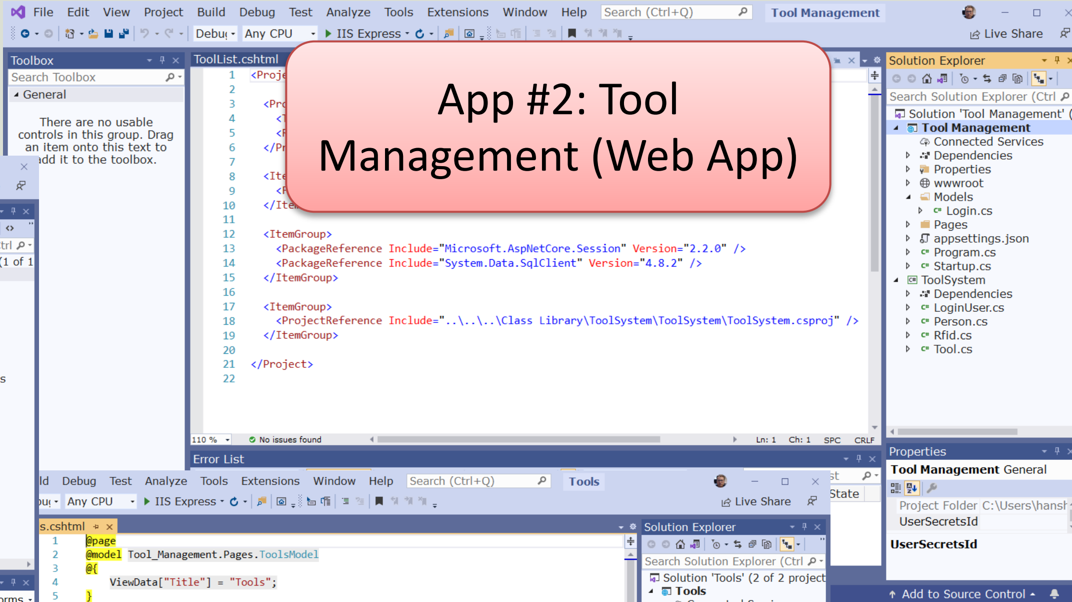
```
IF EXISTS (SELECT name
FROM sysobjects
WHERE name = 'NewTool'
AND type = 'P')
DROP PROCEDURE NewTool
GO

CREATE PROCEDURE NewTool
@ToolName varchar(100),
@ToolTag varchar(100),
@ToolDescription varchar(1000)
AS

if not exists (select * from TOOL where ToolName = @ToolName)
INSERT INTO TOOL (ToolName, ToolTag, ToolDescription, ToolInUse)
VALUES (@ToolName, @ToolTag, @ToolDescription, 0)

GO
```

Visual Studio

Visual Studio

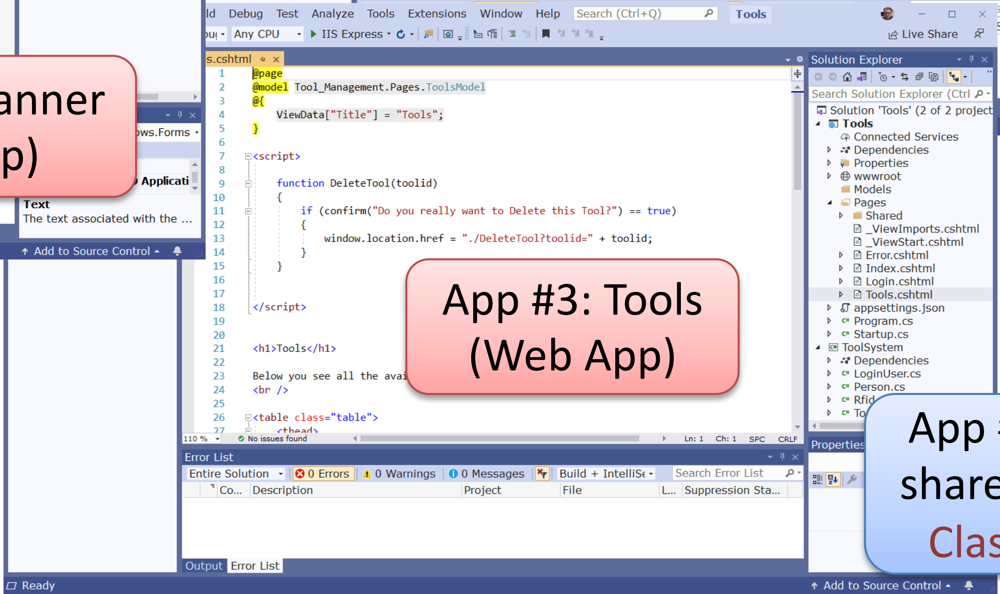


App #1 :Tool Scanner (Desktop App)



```
<Proje

  <Pr
    <
    <
  </Pr

  <Ite
    <
  </Ite

  <ItemGroup>
    <PackageReference Include="Microsoft.AspNetCore.Session" Version="2.2.0" />
    <PackageReference Include="System.Data.SqlClient" Version="4.8.2" />
  </ItemGroup>

  <ItemGroup>
    <ProjectReference Include="..\..\..\Class Library\ToolSystem\ToolSystem\ToolSystem.csproj" />
  </ItemGroup>

</Project>
```

App #2: Tool Management (Web App)



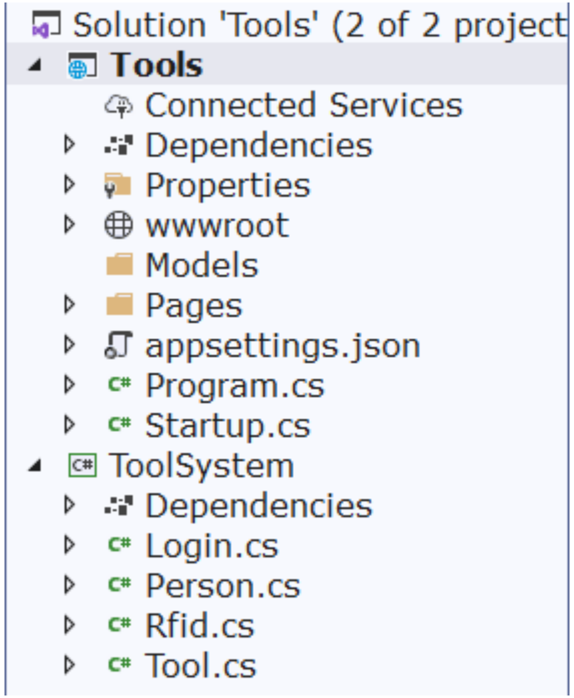
```
@page
@model Tool_Management.Pages.ToolsModel
@{
    ViewData["Title"] = "Tools";
}

<script>

    function DeleteTool(toolid)
    {
        if (confirm("Do you really want to Delete this Tool?") == true)
        {
            window.location.href = "./DeleteTool?toolid=" + toolid;
        }
    }

</script>


<h1>Tools</h1>

Below you see all the avai
<br />

<table class="table">
    <thead>
```

App #3: Tools (Web App)

These 3 applications share much of the same code. That's why a shared Class Library has been used.

App #2 and #3 share the same Class Library

Classes



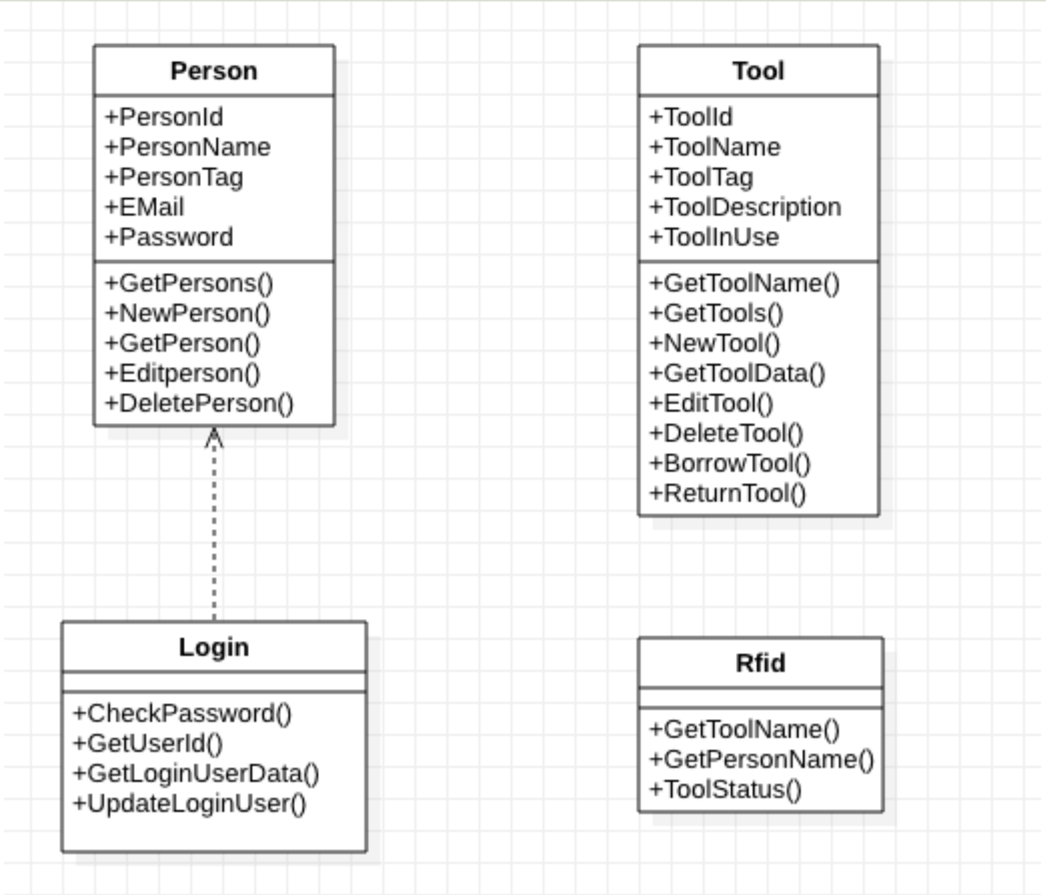
Solution 'Tools' (2 of 2 project
- Tools
  - Connected Services
  - Dependencies
  - Properties
  - wwwroot
  - Models
  - Pages
  - appsettings.json
  - Program.cs
  - Startup.cs
- ToolSystem
  - Dependencies
  - Login.cs
  - Person.cs
  - Rfid.cs
  - Tool.cs

The following Classes have been implemented in a shared Class Library:

- Person
- Tool
- Login
- Rfid

The different Classes are dealing with separate functionality of the applications. All these classes communicate with the database

Class Diagram



This is the Initial Classes for the Core Functionality.
More Classes and Methods may be added later when more functionality will be added.

ASP.NET Core

Web Page: https://halvorsen.blog/documents/programming/web/aspnet

Videos:

- ASP.NET Core – Introduction
  https://youtu.be/zkOtiBcwo8s
- ASP.NET Core – Database Communication
  https://youtu.be/0Ta3dQ3rxzs
- ASP.NET Core - Database CRUD Application
  https://youtu.be/k5TCZDwTYcE
- ASP.NET Core – Class Library
  https://youtu.be/emUiMd1zRrY
- ASP.NET Core – Charts
  https://youtu.be/mksUls9fx-Q
- ASP.NET Core – Session Data
  https://youtu.be/I0SQ_XAoFvA

Web Programming
ASP.NET Core

Hans-Petter Halvorsen

https://www.halvorsen.blog

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog