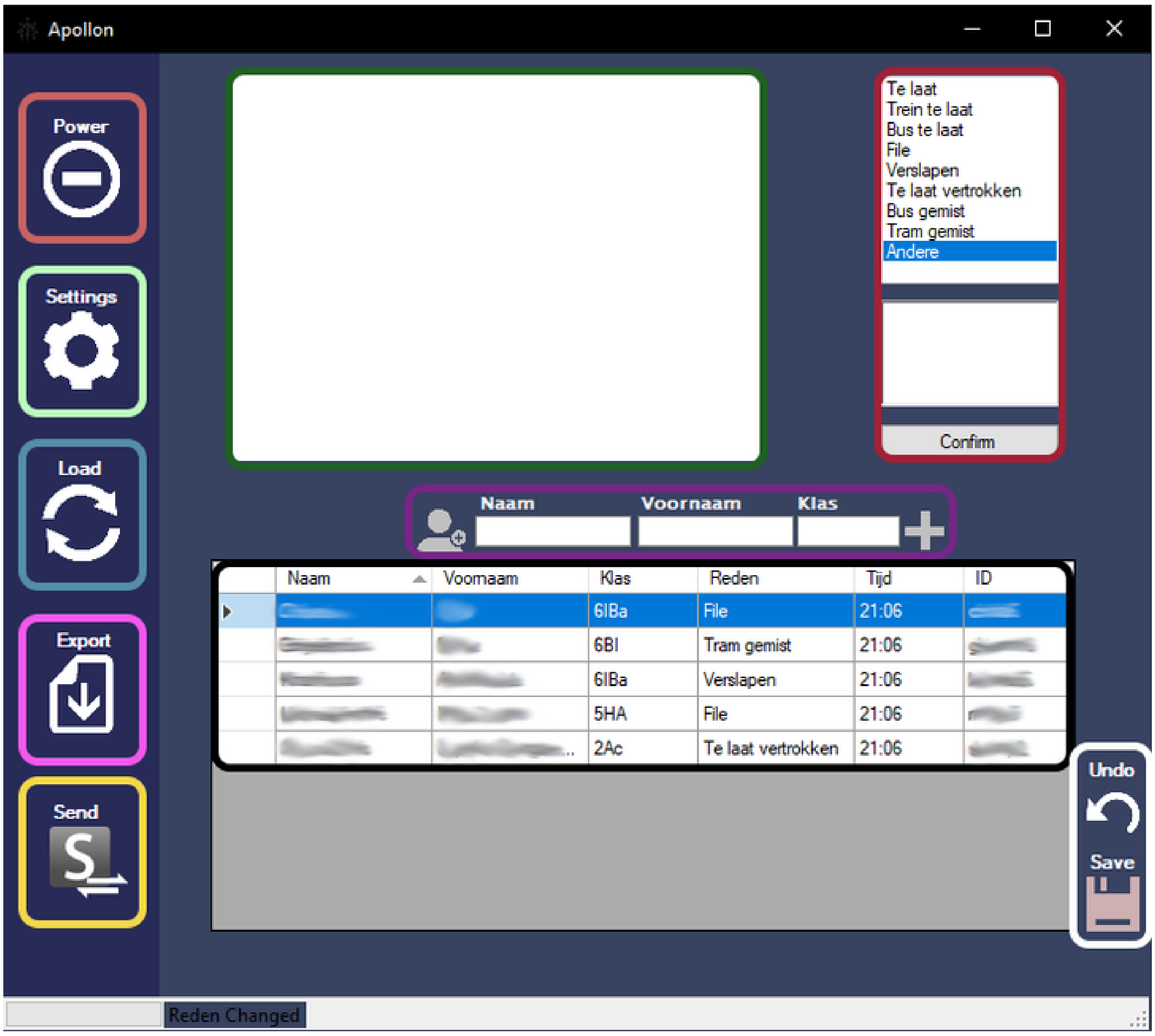
	Naam: -----_-----	Nr.: 8 - 4
	Datum: 2021-05-16	Klas: 6IBa
		Vak: GIP
	Technische handleiding	Punten
		/

How to use Apollon:

The main window



	Naam	Voornaam	Klas	Reden	Tijd	ID
▶	[redacted]	[redacted]	6IBa	File	21:06	[redacted]
	[redacted]	[redacted]	6BI	Tram gemist	21:06	[redacted]
	[redacted]	[redacted]	6IBa	Verslapen	21:06	[redacted]
	[redacted]	[redacted]	5HA	File	21:06	[redacted]
	[redacted]	[redacted]	2Ac	Te laat vertrokken	21:06	[redacted]

Buttons and what they do

Power:

This button starts the application.

Settings:

This button opens the Settings-page.

Load:

This button let you import already saved data from a JSON file.

Export:

This button let you export data in a JSON file.

Send:

This button opens the Smartschool-page.

Reasons-list:

This list changes the “Reden” from the Students-list.

The text block under it can be used to send personalized reasons.

DataGridView:

This list shows all the students that are scanned.

By clicking on the arrow you can select a student.

You can edit a student by double-clicking on the part that you want to change.

CameraFrame:

This box shows a real-time view of the webcam.

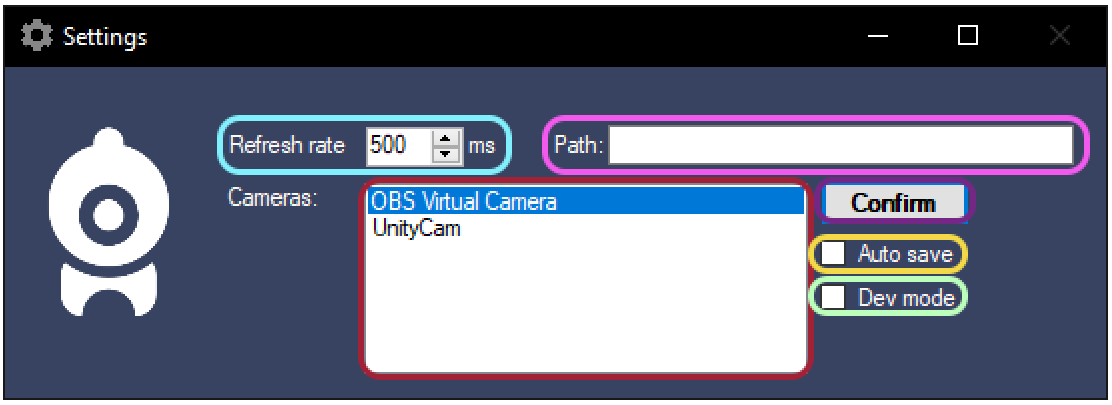
Add student:

This buttons opens three text blocks that can be used to manually add students.

Edit:

These buttons can be used to undo or save a just edited student.

The settings window



Buttons and what they do

Confirm:

This buttons confirm the camera/auto-save/Dev-mode and close the settings-page.

Path:

This text block can be used to chose where you want to export the save-files.

Dev mode:

This button enables the dev mode that shows special labels for the developers.

Auto save:

This buttons saves the students list every time something changes.

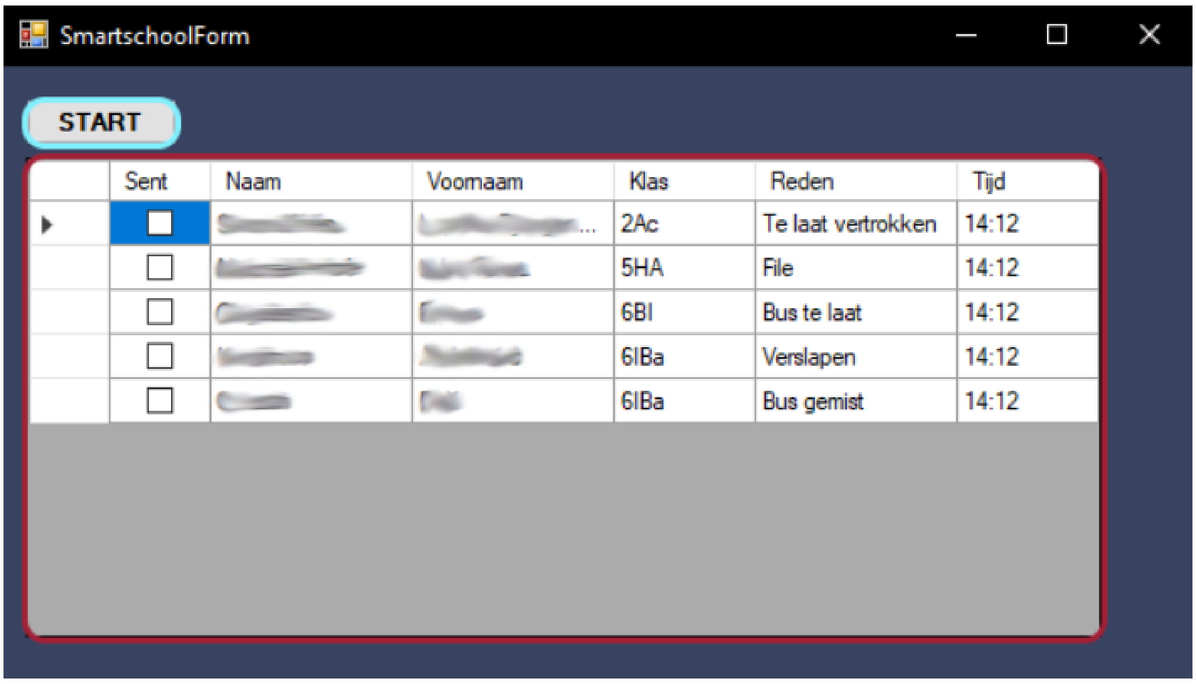
Camera list:

This list must be used to select which camera you are going to use.

Refresh rate:

This text block can be used to change the time between each scan from the camera. (Only use it if the application is laggy)

The smartschool window



	Sent	Naam	Voornaam	Klas	Reden	Tijd
▸	☐	[redacted]	[redacted]	2Ac	Te laat vertrokken	14:12
	☐	[redacted]	[redacted]	5HA	File	14:12
	☐	[redacted]	[redacted]	6BI	Bus te laat	14:12
	☐	[redacted]	[redacted]	6IBa	Verslapen	14:12
	☐	[redacted]	[redacted]	6IBa	Bus gemist	14:12

This window main purpose is to register all the students on Smartschool. It will launch a web browser and go through all the pages to register all the students one by one.

<u>Buttons</u>

Students list:

This list shows which students will be sent and when they will be sent the checkbox will be checked.

Start:

This button opens a Firefox window with smartschool on it and starts registering the students on Smartschool.

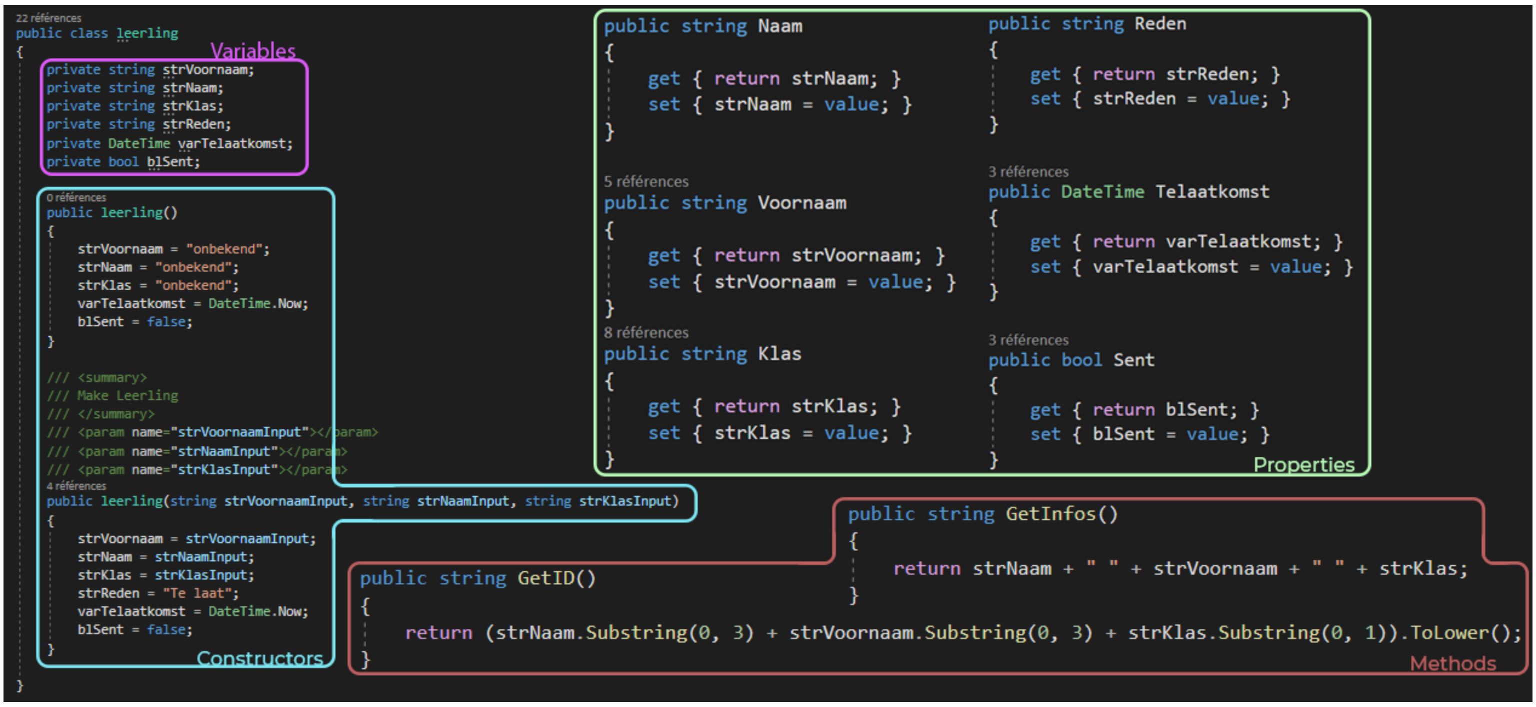
For Developers

We have open sourced our code and is hosted on GitHub if you want to read through.

Leerling.cs

github.com/Kaminiix/Apollon/blob/Apollon-Winform/TestCam/leerling.cs

shorten: **bit.ly/3fqgdv2**



```
22 références
public class leerling
{
    private string strVoornaam;
    private string strNaam;
    private string strKlas;
    private string strReden;
    private DateTime varTelaatkomst;
    private bool blSent;

    0 références
    public leerling()
    {
        strVoornaam = "onbekend";
        strNaam = "onbekend";
        strKlas = "onbekend";
        varTelaatkomst = DateTime.Now;
        blSent = false;
    }

    /// <summary>
    /// Make Leerling
    /// </summary>
    /// <param name="strVoornaamInput"></param>
    /// <param name="strNaamInput"></param>
    /// <param name="strKlasInput"></param>
    4 références
    public leerling(string strVoornaamInput, string strNaamInput, string strKlasInput)
    {
        strVoornaam = strVoornaamInput;
        strNaam = strNaamInput;
        strKlas = strKlasInput;
        strReden = "Te laat";
        varTelaatkomst = DateTime.Now;
        blSent = false;
    }
}
```

```
public string Naam
{
    get { return strNaam; }
    set { strNaam = value; }
}

5 références
public string Voornaam
{
    get { return strVoornaam; }
    set { strVoornaam = value; }
}
8 références
public string Klas
{
    get { return strKlas; }
    set { strKlas = value; }
}

public string Reden
{
    get { return strReden; }
    set { strReden = value; }
}

3 références
public DateTime Telaatkomst
{
    get { return varTelaatkomst; }
    set { varTelaatkomst = value; }
}

3 références
public bool Sent
{
    get { return blSent; }
    set { blSent = value; }
}
```

```
public string GetInfos()
{
    return strNaam + " " + strVoornaam + " " + strKlas;
}

public string GetID()
{
    return (strNaam.Substring(0, 3) + strVoornaam.Substring(0, 3) + strKlas.Substring(0, 1)).ToLower();
}
```

Variables

strVoornaam: is a string that contains the first name of the student.

strNaam: is a string that contains the last name of the student.

strKlas: is a string that contains the class in which the student is.

strReden: is a string that contains the reason why the student is being late

varTelaatkomst: is a Time variable that contains the moment student was registered

blSent: is a boolean variable that is used to know if the student has been sent to Smartschool.

Constructors

A constructor is a piece of code that will be executed when an object (here a student) is being made. We use this to give the students variables a default value.

```
leerling(string strVoornaamInput, string strNaamInput, string strKlasnput)
```

Example on how to use:

```
leerling Me = new leerling(“Alan”, “Smithee”, “6IBa”);
```

Methods

Methods are functions that process something and returns a value. Each student have 2 of them.

GetInfos() it returns the information (first and lastname and class) of the given student. Example Me.GetInfos() will return the following line.

```
Me.GetInfos()
   - Alan Smithee 6IBa
```

GetID() This returns the ID of the student. We use this function to quickly identify a student and make sure it’s not already registered. Its ID is made with the 2 first letters of his first and last name and the number of his class. This should in theory make them all unique. Example Me.GetID() will return the following line.

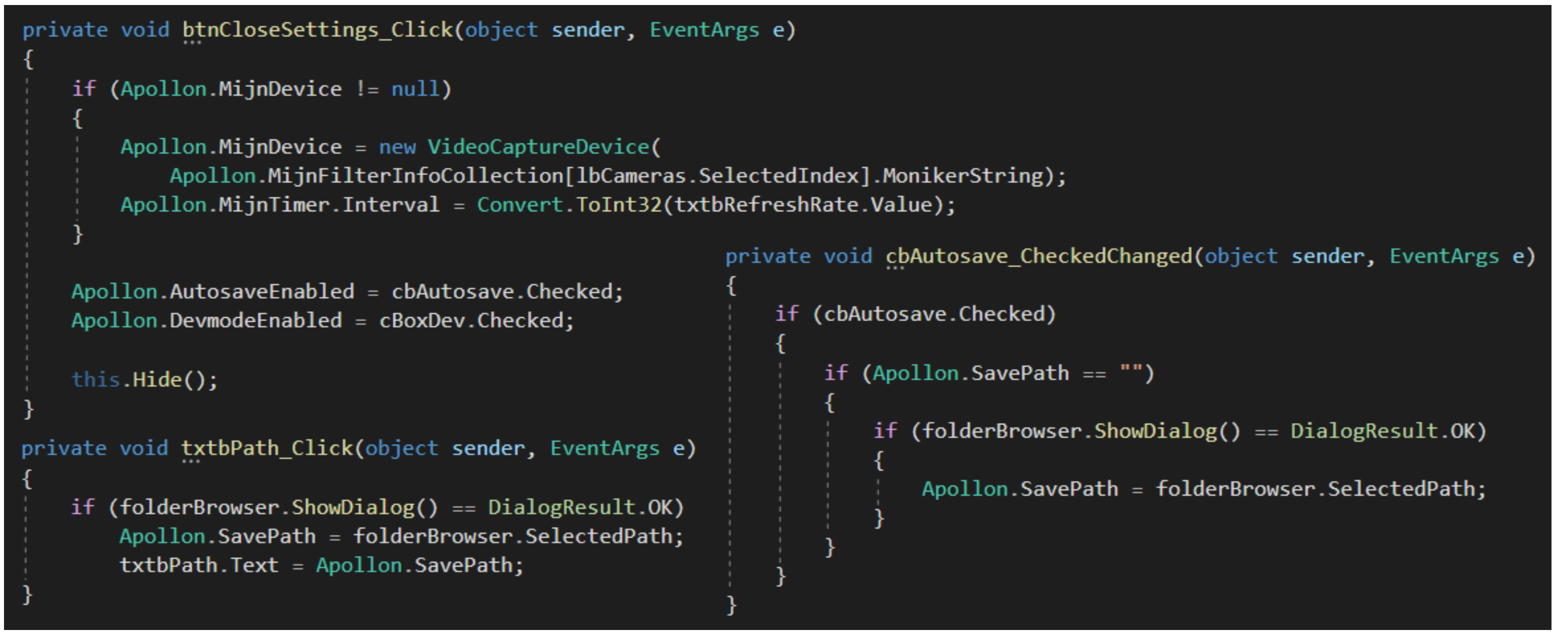
```
Me.GetID()
   - alsm6
```

Properties

Those are the properties every student have for example every student has a first name a class and such. Those can be called like this.

```
Me.Naam
   - Smithee
Me.Klas
   - 6IBa
```

SettingsForm.cs



```
private void btnCloseSettings_Click(object sender, EventArgs e)
{
    if (Apollon.MijnDevice != null)
    {
        Apollon.MijnDevice = new VideoCaptureDevice(
            Apollon.MijnFilterInfoCollection[lbCameras.SelectedIndex].MonikerString);
        Apollon.MijnTimer.Interval = Convert.ToInt32(txtbRefreshRate.Value);
    }

    Apollon.AutosaveEnabled = cbAutosave.Checked;
    Apollon.DevmodeEnabled = cBoxDev.Checked;

    this.Hide();
}
private void txtbPath_Click(object sender, EventArgs e)
{
    if (folderBrowser.ShowDialog() == DialogResult.OK)
        Apollon.SavePath = folderBrowser.SelectedPath;
        txtbPath.Text = Apollon.SavePath;
}
```

```
private void cbAutosave_CheckedChanged(object sender, EventArgs e)
{
    if (cbAutosave.Checked)
    {
        if (Apollon.SavePath == "")
        {
            if (folderBrowser.ShowDialog() == DialogResult.OK)
            {
                Apollon.SavePath = folderBrowser.SelectedPath;
            }
        }
    }
}
```

github.com/Kaminiix/Apollon/blob/Apollon-Winform/TestCam/SettingsForm.cs

shorten: ***bit.ly/3w7Lee2***

`SettingsForm.cs` is the settings window the code of this window only made of events. Events are pieces of code that are executed when an event occurs. For example, `txtbPath_Click()` that is executed when the textbox is clicked.

Functions

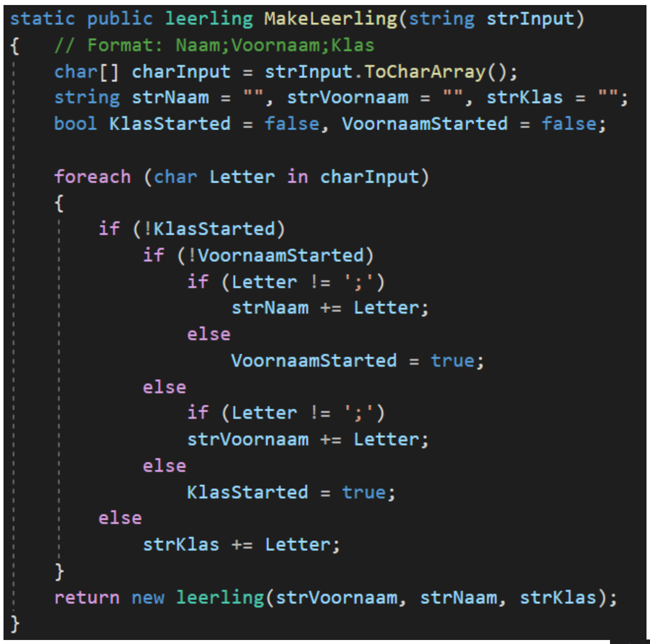
`btnCloseSettings_Click()` is executed when the confirm button is clicked. It will save the selected video capture device and hide the window.
`cbAutosave_CheckedChanged()` is executed when the checkbox is clicked and if no path has been selected it will ask for one.
`txtbPath_Click()` is executed when the textbox is clicked and will open a folder window to ask what path the user wants to use and will save it.

Main.cs

github.com/Kaminiix/Apollon/blob/Apollon-Winform/TestCam/Main.cs

shorten: ***bit.ly/3yhweMH***

This is the main window of the application. Most of the code and work is in this file. It contains work from over 5 months, and it might be difficult to read for non-experienced people. We have written comments all over our code to explain and make it more comprehensible



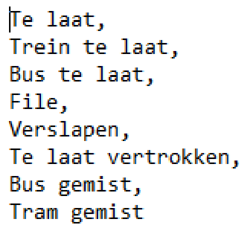
```
static public leerling MakeLeerling(string strInput)
{   // Format: Naam;Voornaam;Klas
    char[] charInput = strInput.ToCharArray();
    string strNaam = "", strVoornaam = "", strKlas = "";
    bool KlasStarted = false, VoornaamStarted = false;

    foreach (char Letter in charInput)
    {
        if (!KlasStarted)
            if (!VoornaamStarted)
                if (Letter != ';')
                    strNaam += Letter;
                else
                    VoornaamStarted = true;
            else
                if (Letter != ';')
                strVoornaam += Letter;
            else
                KlasStarted = true;
        else
            strKlas += Letter;
    }
    return new leerling(strVoornaam, strNaam, strKlas);
}
```

This function makes a student from a string. Like this `MakeLeerling("Smithee;Alan;6IBa")`

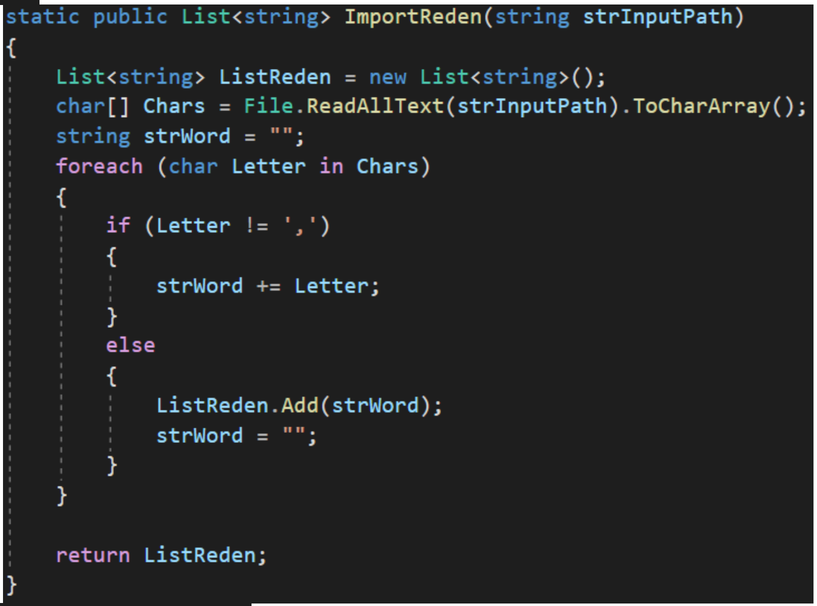
It's an algorithm we've written to recognize the first and last name and the class which have to be separated with a ';'.

`ImportReden()` this will make a list of reasons (reden) from a string in which every reason are separated by commas. The program will import it from a text file. Like this one



```
Te laat,
Trein te laat,
Bus te laat,
File,
Verslapen,
Te laat vertrokken,
Bus gemist,
Tram gemist
```

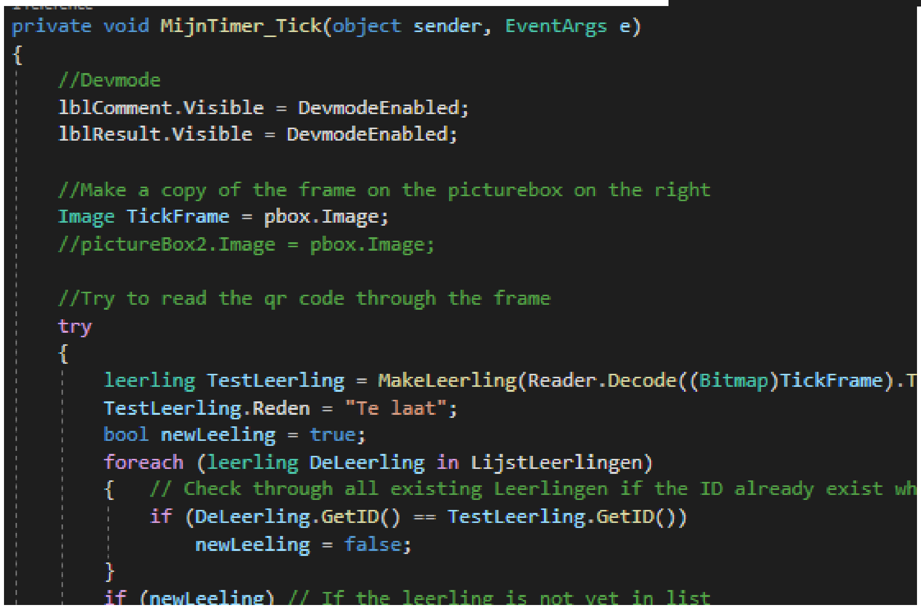
reden.txt



```
static public List<string> ImportReden(string strInputPath)
{

    List<string> ListReden = new List<string>();
    char[] Chars = File.ReadAllText(strInputPath).ToCharArray();
    string strWord = "";
    foreach (char Letter in Chars)
    {
        if (Letter != ',')
        {
            strWord += Letter;
        }
        else
        {
            ListReden.Add(strWord);
            strWord = "";
        }
    }

    return ListReden;
}
```

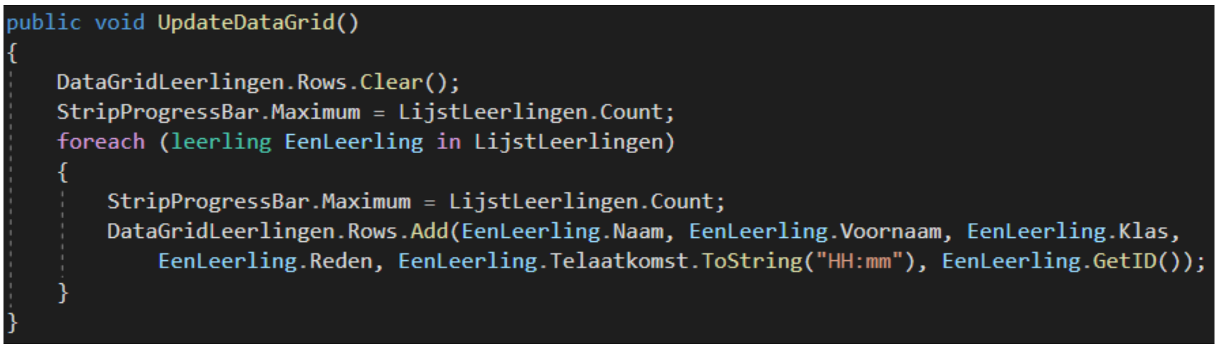


```
private void MijnTimer_Tick(object sender, EventArgs e)
{
    //Devmode
    lblComment.Visible = DevmodeEnabled;
    lblResult.Visible = DevmodeEnabled;

    //Make a copy of the frame on the picturebox on the right
    Image TickFrame = pbox.Image;
    //pictureBox2.Image = pbox.Image;

    //Try to read the qr code through the frame
    try
    {
        leerling TestLeerling = MakeLeerling(Reader.Decode((Bitmap)TickFrame).T
        TestLeerling.Reden = "Te laat";
        bool newLeeling = true;
        foreach (leerling DeLeerling in LijstLeerlingen)
        {   // Check through all existing Leerlingen if the ID already exist wh
            if (DeLeerling.GetID() == TestLeerling.GetID())
                newLeeling = false;
        }
        if (newLeeling) // If the leerling is not yet in list
```

`MijnTimer_Tick()` this is the function that is executed every tick. Most of the work happens here like the analysis of the webcam frame for searching a qr code. And if found read it and make a student of it and such.



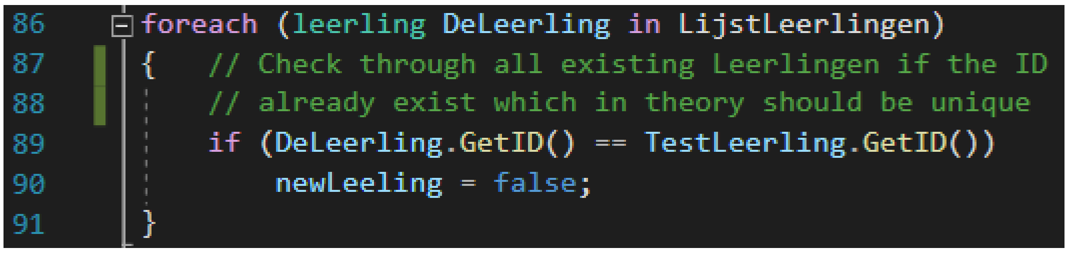
```
public void UpdateDataGrid()
{
    DataGridLeerlingen.Rows.Clear();
    StripProgressBar.Maximum = LijstLeerlingen.Count;
    foreach (leerling EenLeerling in LijstLeerlingen)
    {
        StripProgressBar.Maximum = LijstLeerlingen.Count;
        DataGridLeerlingen.Rows.Add(EenLeerling.Naam, EenLeerling.Voornaam, EenLeerling.Klas,
            EenLeerling.Reden, EenLeerling.Telaatkomst.ToString("HH:mm"), EenLeerling.GetID());
    }
}
```

`UpdateDataGrid()` is a function that when is called updates all the data in the DataGridView that is in the main window.

The main variables are the following:

- **LijstLeerlingen** is a list of every student.
- **LijstReden** is a list of all the reasons.
- **Reader** is the reader that reads through the webcam frames.
- **MijnDevice** is the capture device used.

Here is an example of how we check if a student is new. We go through every student in the list and compare the IDs.



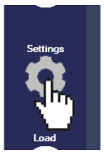
```
foreach (leerling DeLeerling in LijstLeerlingen)
{   // Check through all existing Leerlingen if the ID
    // already exist which in theory should be unique
    if (DeLeerling.GetID() == TestLeerling.GetID())
        newLeeling = false;
}
```

Step by step

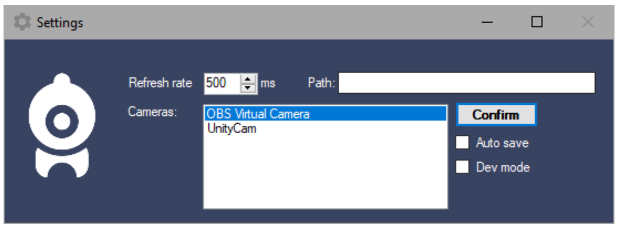
on how a regular use should go

1) Open the Apollon shortcut.

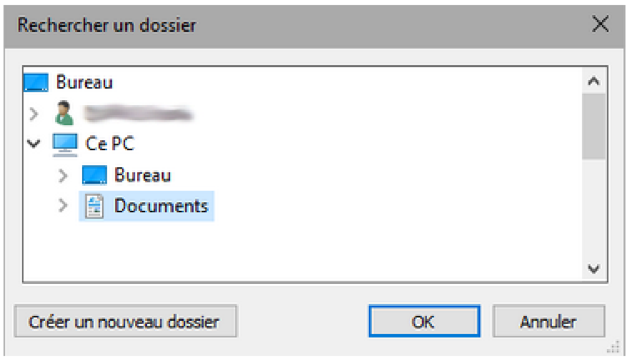
2) When the program has launched you should start by configuring it. Press the Settings button on the main page.



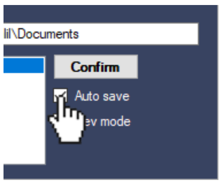
3) Chose your camera from the list. You can change the refresh rate if your computer is being slow.

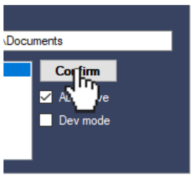


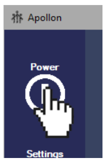
4) Click on "Path" and chose which folder you want to use as work folder. Your data will be exported there.



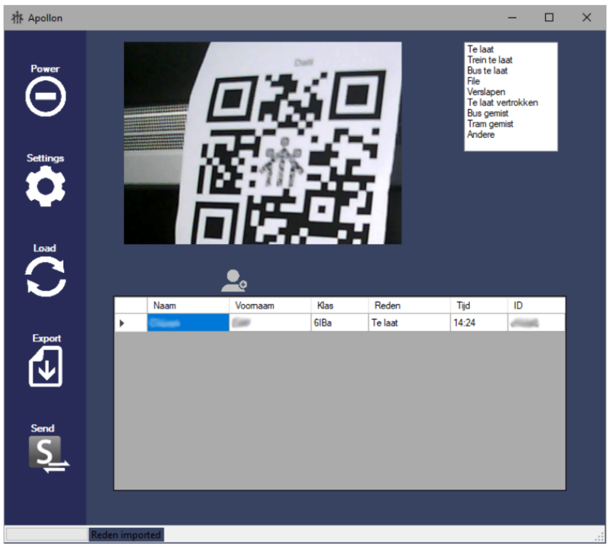
We highly recommend you checking the "Auto save" box. This will make a backup of your work which you could later on import, must the program be closed.





5) Once the configuration done you can start the program by clicking on this button. 

6) You can start scanning students QR-codes.

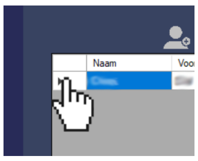


Te laat
Trein te laat
Bus te laat
File
Verslapen
Te laat vertrokken
Bus gemist
Tram gemist
Andere

	Naam	Voornaam	Klas	Reden	Tijd	ID
▸	[redacted]	[redacted]	6IBa	Te laat	14:24	[redacted]

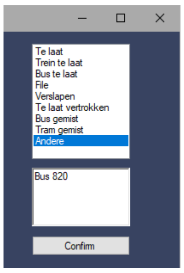
Reden imported

7) Press the little arrow next to the student to select him/her.
You can by the way select multiple student by holding **ctrl** and clicking on the wanted student



	Naam	Voo
	[redacted]	[redacted]

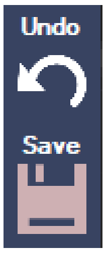
8) You can put personalized reasons for the student by clicking “Andere”.


Te laat
Trein te laat
Bus te laat
File
Verslapen
Te laat vertrokken
Bus gemist
Tram gemist
Andere

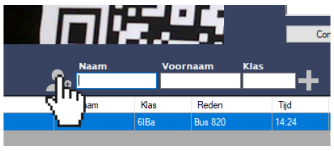
Bus 820

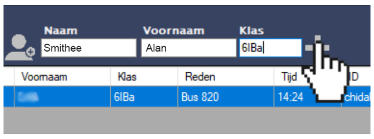
9) When you have changed something in the list of students like the reason of a student or his name you might have noticed those icons have appeared.

Those mean that you have unsaved work. You can save all your modification by clicking on the floppy disk.
Or undo all your work by clicking on the undo arrow.



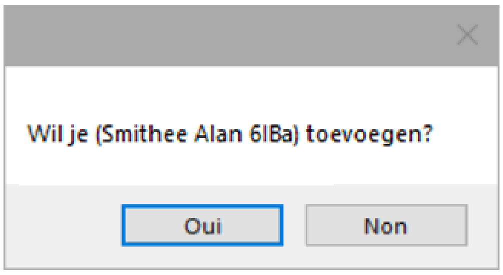
10) If you want to register a student without scanning his QR code you can add him manually to the list by clicking on the next button.





Naam	Voornaam	Klas
Smithee	Alan	6IBa

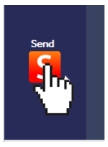
Voornaam	Klas	Reden	Tijd
[illegible]	6IBa	Bus 820	14:24



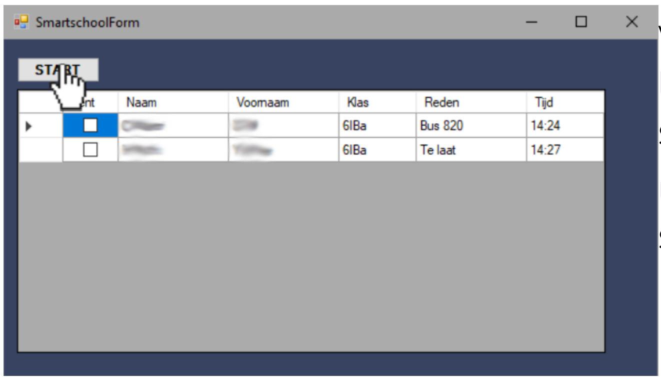
Wil je (Smithee Alan 6IBa) toevoegen?

You will have to confirm if you want to add it.

11) Now that you have a list of students and have assigned them all a reason for their late coming. You can send them to Smartschool. You can do that by clicking on this button.



12) When you have verified the list of students you can press the “Start” button. If the student has been successfully set as late the “sent” box next to him will be checked. Once the browser has opened the Smartschool page it will wait 2 minutes until you log into Smartschool.



nt	Naam	Voornaam	Klas	Reden	Tijd
☐	[illegible]	[illegible]	6IBa	Bus 820	14:24
☐	[illegible]	[illegible]	6IBa	Te laat	14:27

Once this is done you can check if every student has been registered. And close the program if you are done.

Thank you for using our creation. :)

Yassine and Dalil