Wireless Communication

TP

Wallet Application

Done by

Christian Hasbani

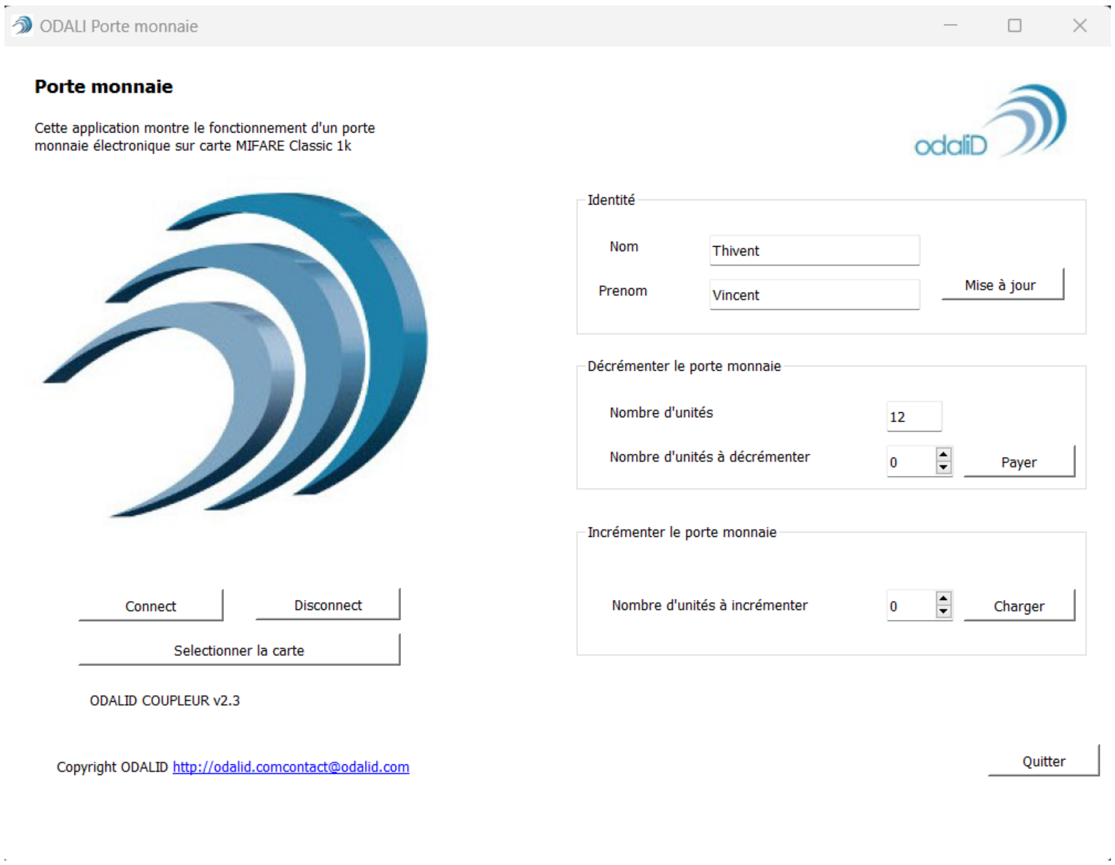
Presented to

Mr. Thevint Vincent

Contents

I. GUI Presentation

In this TP we are working on developing a wallet application with the following GUI



ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité

Nom: Thivent

Prenom: Vincent

Décrémenter le porte monnaie

Nombre d'unités: 12

Nombre d'unités à décrémenter: 0

Incrémenter le porte monnaie

Nombre d'unités à incrémenter: 0

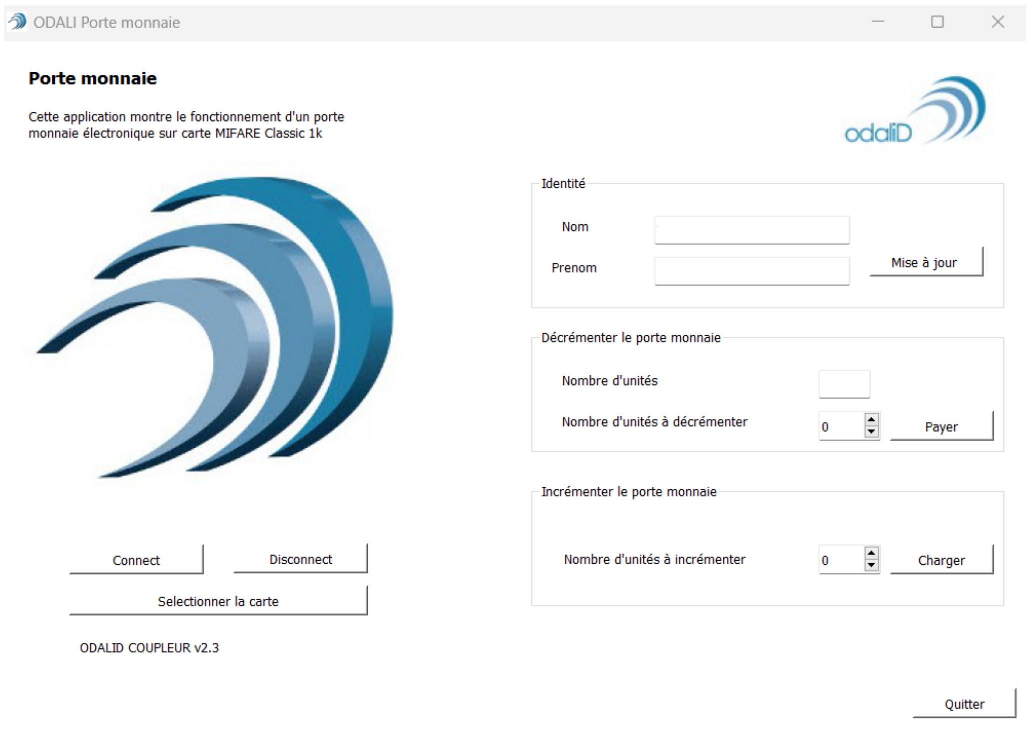
ODALID COUPLEUR v2.3

Copyright ODALID http://odalid.comcontact@odalid.com

Figure 1-GUI of the wallet application

II. Connecting the reader to the application

We start by pressing the connect button that connects the reader to our application


ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Figure 2-before connect GUI

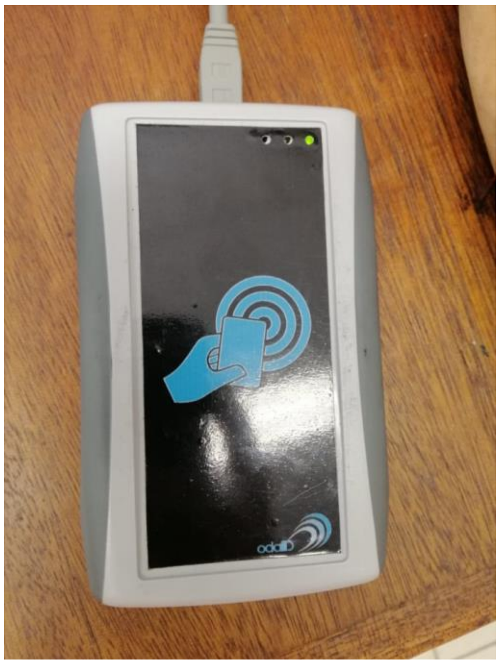
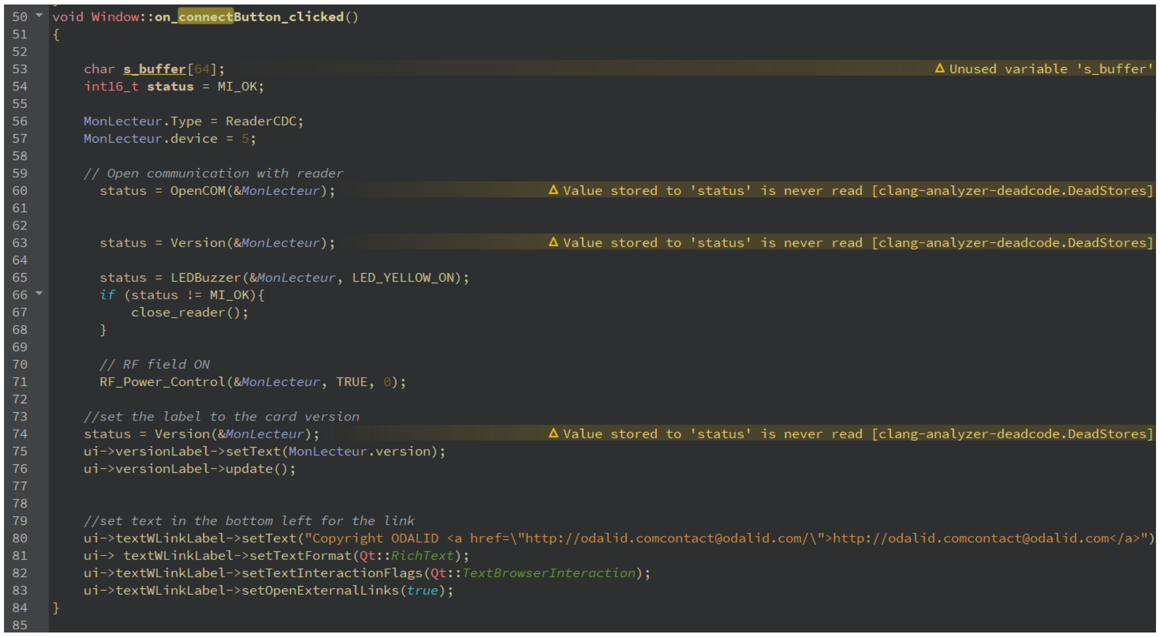


Figure 3-The reader before pressing connect

Once we press the connect button the following code will be run



```
void Window::on_connectButton_clicked()
{

    char s_buffer[64];                                          ⚠ Unused variable 's_buffer'
    int16_t status = MI_OK;

    MonLecteur.Type = ReaderCDC;
    MonLecteur.device = 5;

    // Open communication with reader
      status = OpenCOM(&MonLecteur);          ⚠ Value stored to 'status' is never read [clang-analyzer-deadcode.DeadStores]


      status = Version(&MonLecteur);          ⚠ Value stored to 'status' is never read [clang-analyzer-deadcode.DeadStores]

      status = LEDBuzzer(&MonLecteur, LED_YELLOW_ON);
      if (status != MI_OK){
          close_reader();
      }

      // RF field ON
      RF_Power_Control(&MonLecteur, TRUE, 0);

    //set the label to the card version
    status = Version(&MonLecteur);            ⚠ Value stored to 'status' is never read [clang-analyzer-deadcode.DeadStores]
    ui->versionLabel->setText(MonLecteur.version);
    ui->versionLabel->update();


    //set text in the bottom left for the link
    ui->textWLinkLabel->setText("Copyright ODALID <a href=\"http://odalid.comcontact@odalid.com/\">http://odalid.comcontact@odalid.com</a>")
    ui-> textWLinkLabel->setTextFormat(Qt::RichText);
    ui->textWLinkLabel->setTextInteractionFlags(Qt::TextBrowserInteraction);
    ui->textWLinkLabel->setOpenExternalLinks(true);
}
```

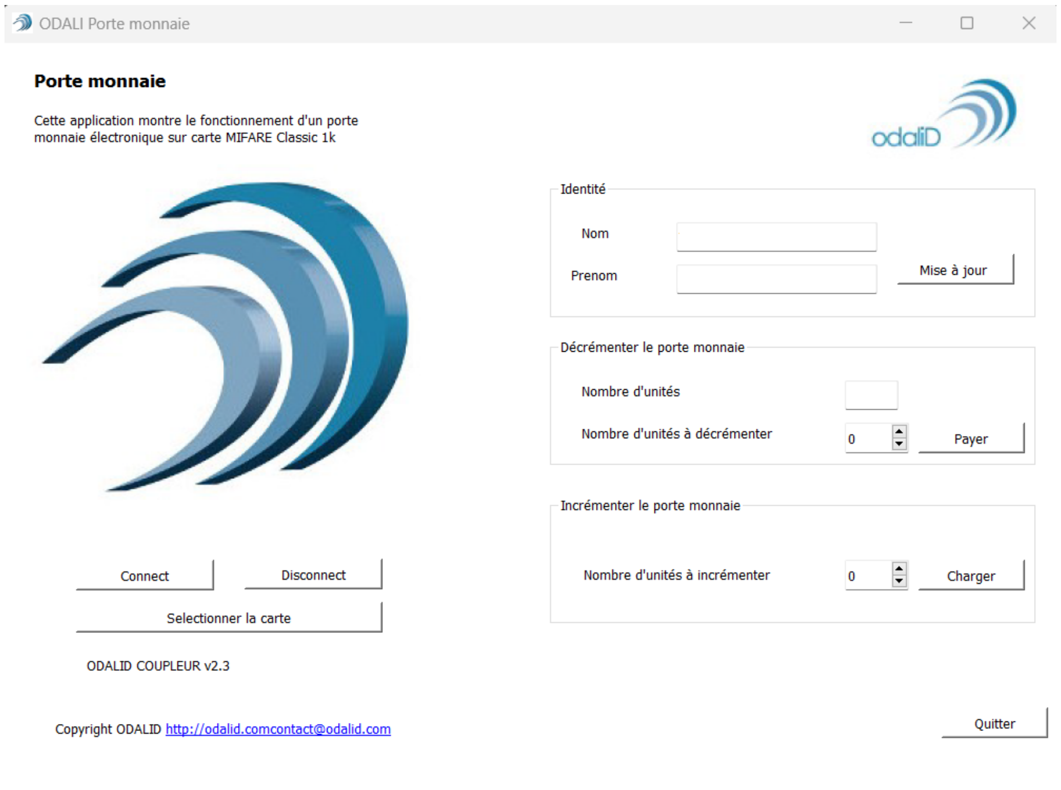
Figure 4-code for connect button

The main function of this code is to open communication between the application and the reader through functions like “OpenCOM” and get the version of the card “Version” and in case there was an error in the communication (if the version function returns a value different than “MI_OK” which is 0) then we close the communication through the “close_reader” function which will be shown later in the disconnect function.

After than we turn the RF field ON through the “RF_Power_control” function, then we show the version of the card (can be seen in figure 2) through the “setText” and “update” functions.

Finally, we add some text at the bottom of the GUI (seen in figure 4) and adding the hyperlink to the ODALID website.

So, we get this result after running the connect function


ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité

Nom

Prenom

Décrémenter le porte monnaie

Nombre d'unités

Nombre d'unités à décrémenter 0

Incrémenter le porte monnaie

Nombre d'unités à incrémenter 0

ODALID COUPLEUR v2.3

Copyright ODALID http://odalid.comcontact@odalid.com

Figure 5-after connect GUI

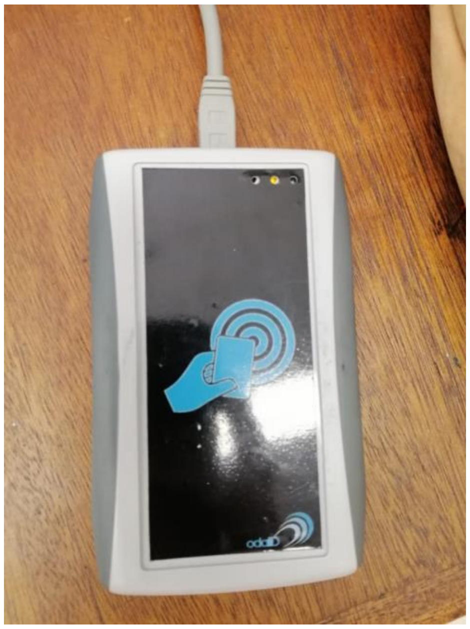
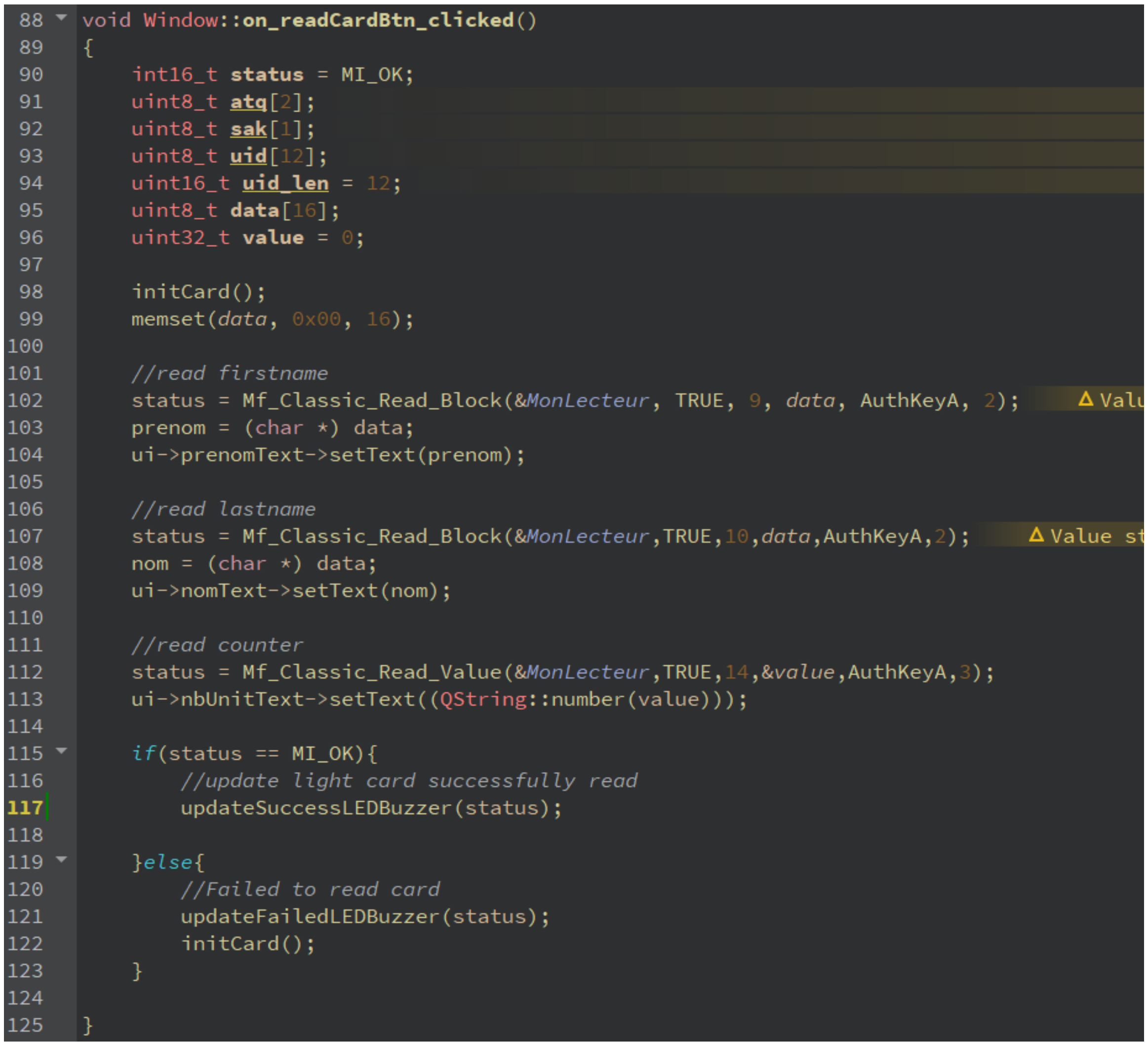


Figure 6-Reader after pressing connect (showing a yellow LED)

So now the reader is connected to our application and ready to read data from the MIFARE card.

III. Reading from the MIFARE card

Here we initiate the reading process once we press on the “selectionner la carte” button “figure 5”



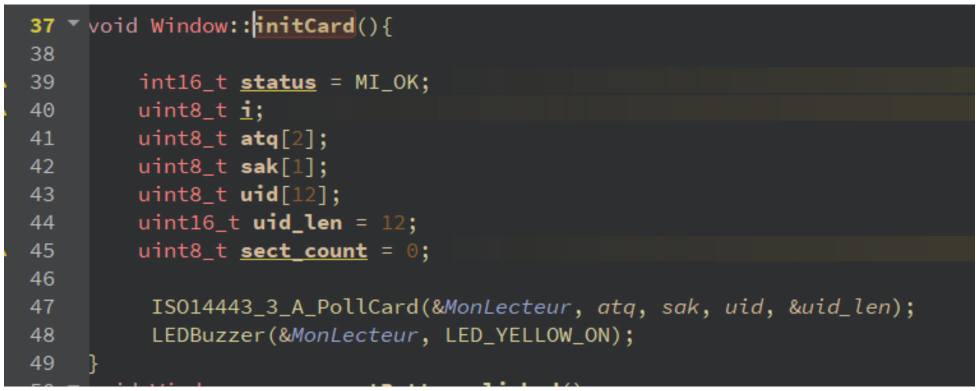
```
void Window::on_readCardBtn_clicked()
{
    int16_t status = MI_OK;
    uint8_t atq[2];
    uint8_t sak[1];
    uint8_t uid[12];
    uint16_t uid_len = 12;
    uint8_t data[16];
    uint32_t value = 0;

    initCard();
    memset(data, 0x00, 16);

    //read firstname
    status = Mf_Classic_Read_Block(&MonLecteur, TRUE, 9, data, AuthKeyA, 2);     ⚠ Valu
    prenom = (char *) data;
    ui->prenomText->setText(prenom);

    //read lastname
    status = Mf_Classic_Read_Block(&MonLecteur,TRUE,10,data,AuthKeyA,2);     ⚠ Value st
    nom = (char *) data;
    ui->nomText->setText(nom);

    //read counter
    status = Mf_Classic_Read_Value(&MonLecteur,TRUE,14,&value,AuthKeyA,3);
    ui->nbUnitText->setText((QString::number(value)));

    if(status == MI_OK){
        //update light card successfully read
        updateSuccessLEDBuzzer(status);

    }else{
        //Failed to read card
        updateFailedLEDBuzzer(status);
        initCard();
    }

}
```

Figure 7-Read card function code

The function starts by calling the “initCard” (figure 8) function which initializes the MIFARE card to be ready to read data from.

Then we call the “Mf_Classic_Read_Block” function to read data specifically the first name, last name and the “Mf_Classic_Read_Value” to read the counter to be displayed on the GUI.

At the end we check if the status is 0 (MI_OK is zero) then the reading of the card was successful so we call the “updateSuccessLEDBuzzer” to be explained later (figure 11) else we failed to read the card and call the “updateFailedLEDBuzzer” function to be shown later (figure 15).



```
void Window::initCard(){

    int16_t status = MI_OK;
    uint8_t i;
    uint8_t atq[2];
    uint8_t sak[1];
    uint8_t uid[12];
    uint16_t uid_len = 12;
    uint8_t sect_count = 0;

     ISO14443_3_A_PollCard(&MonLecteur, atq, sak, uid, &uid_len);
     LEDBuzzer(&MonLecteur, LED_YELLOW_ON);
}
```

Figure 8-initCard function

This function uses mainly the “ISO14443_3_A_Pollcard” function to initialize the card.

a) Card is present on the reader

If the card is present on the reader

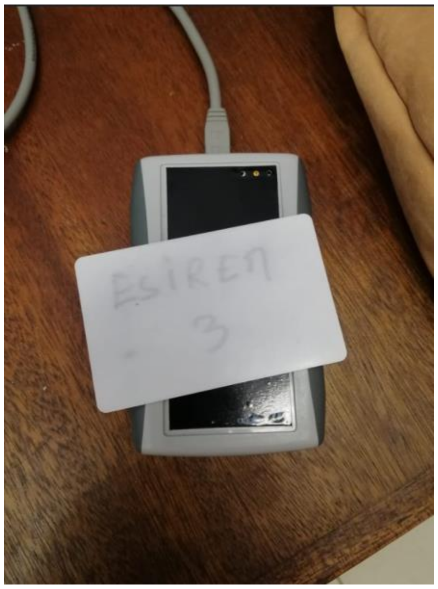
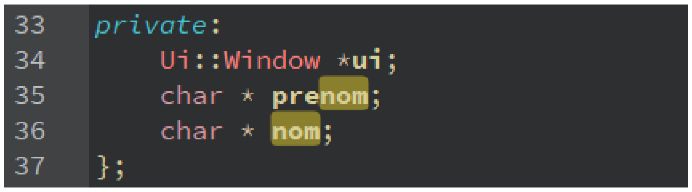


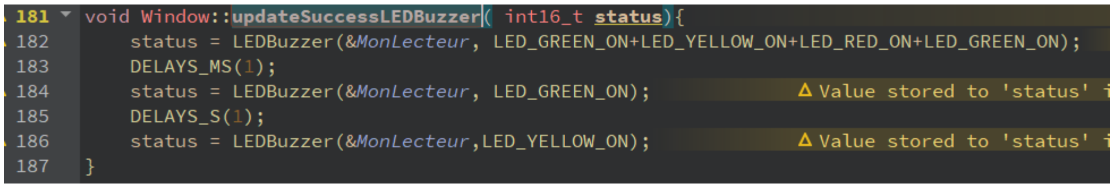
Figure 9-Reader with a MIFARE card before pressing the read card button

As we can see the color of the reader is still yellow from the initialization process. After pressing the read button on the GUI “selectionner la carte” since the process will be successful, we will call to a function called “updateSuccessLEDBuzzer” also the “prenom” and “nom” variables are global variable to be used later.



```
private:
    Ui::Window *ui;
    char * prenom;
    char * nom;
};
```

Figure 10-nom and prenom variable defined in the header file of the project



```
void Window::updateSuccessLEDBuzzer( int16_t status){
    status = LEDBuzzer(&MonLecteur, LED_GREEN_ON+LED_YELLOW_ON+LED_RED_ON+LED_GREEN_ON);
    DELAYS_MS(1);
    status = LEDBuzzer(&MonLecteur, LED_GREEN_ON);              ⚠ Value stored to 'status'
    DELAYS_S(1);
    status = LEDBuzzer(&MonLecteur,LED_YELLOW_ON);              ⚠ Value stored to 'status'
}
```

Figure 11-update LED and buzzer success function code

After compiling this code, the reader will change its LED color to green for 1 second and make a sound with the buzzer then it changes back to the yellow LED.

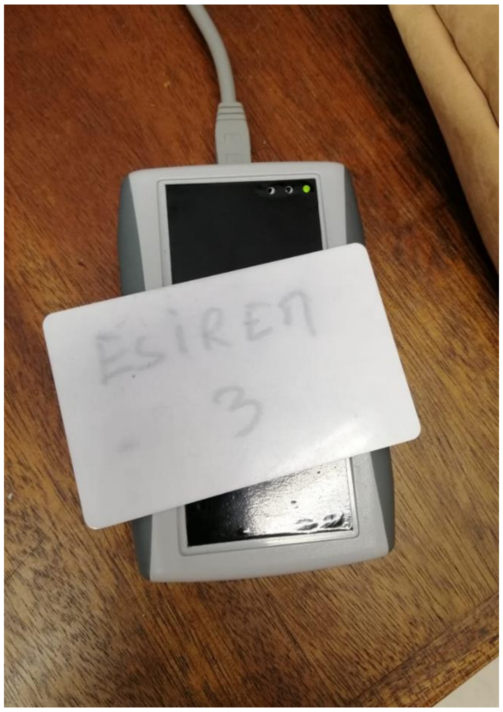
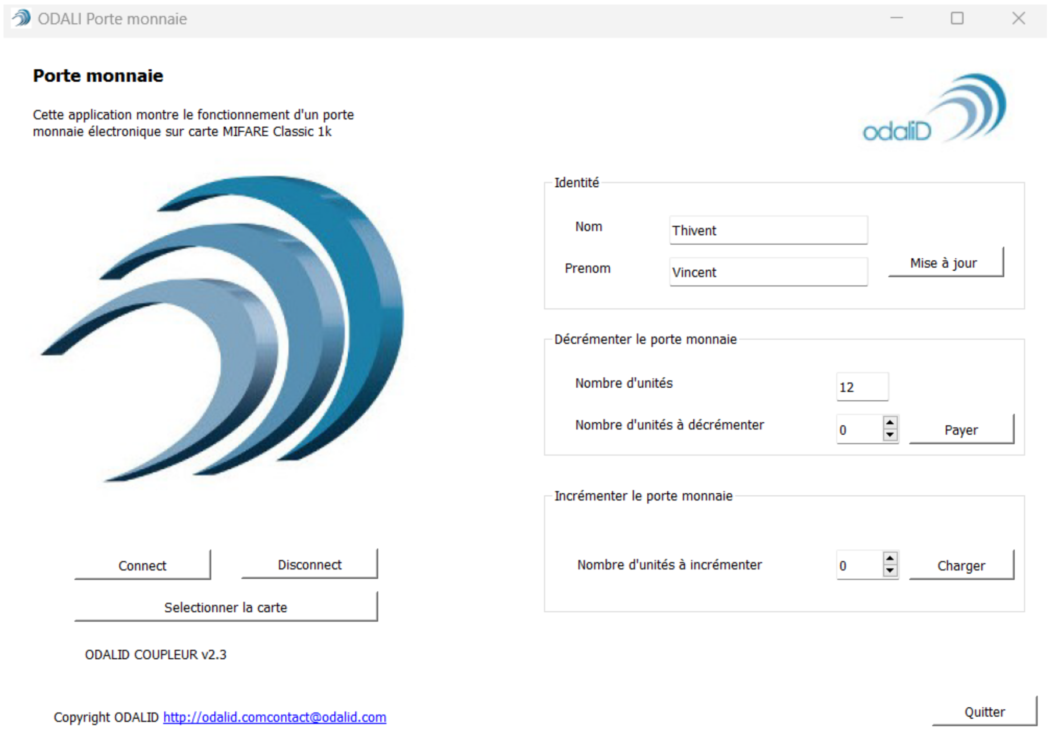


Figure 12-Reader after reading the card successfully

After reading the card data successfully we can see the following GUI.


ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité

Nom: Thivent

Prenom: Vincent

Décrémenter le porte monnaie

Nombre d'unités: 12

Nombre d'unités à décrémenter: 0

Incrémenter le porte monnaie

Nombre d'unités à incrémenter: 0

ODALID COUPLEUR v2.3

Copyright ODALID http://odalid.comcontact@odalid.com

Figure 13-After successfully reading the card GUI

b) Card is not present on the reader

In case we don't place a card on the reader and press the read button "selectionner la carte" like the following photo

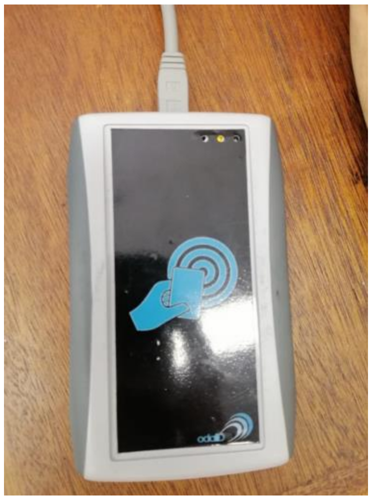
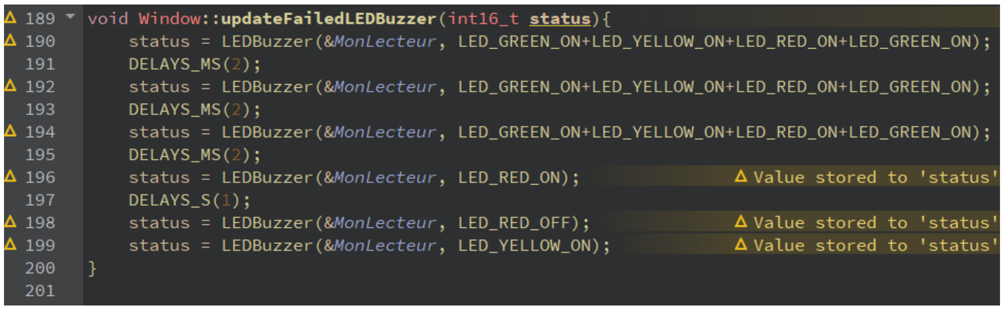


Figure 14-Reader before trying to read without a card

Once we press the read button and the read card function is called (Figure 7) the status returned from the “Mf_Classic_Read_Value” function will be different from “MI_OK” different from 0. So, the “updateFailedLEDBuzzer**”**



```
void Window::updateFailedLEDBuzzer(int16_t status){
    status = LEDBuzzer(&MonLecteur, LED_GREEN_ON+LED_YELLOW_ON+LED_RED_ON+LED_GREEN_ON);
    DELAYS_MS(2);
    status = LEDBuzzer(&MonLecteur, LED_GREEN_ON+LED_YELLOW_ON+LED_RED_ON+LED_GREEN_ON);
    DELAYS_MS(2);
    status = LEDBuzzer(&MonLecteur, LED_GREEN_ON+LED_YELLOW_ON+LED_RED_ON+LED_GREEN_ON);
    DELAYS_MS(2);
    status = LEDBuzzer(&MonLecteur, LED_RED_ON);             Δ Value stored to 'status'
    DELAYS_S(1);
    status = LEDBuzzer(&MonLecteur, LED_RED_OFF);            Δ Value stored to 'status'
    status = LEDBuzzer(&MonLecteur, LED_YELLOW_ON);          Δ Value stored to 'status'
}
```

Figure 15-udpate LED and buzzer failed

This function will make the buzzer make a sound 3 times in a row with a 2millisecond delay between each and then make the LED red for 1 sec after that it will be back to the yellow LED.

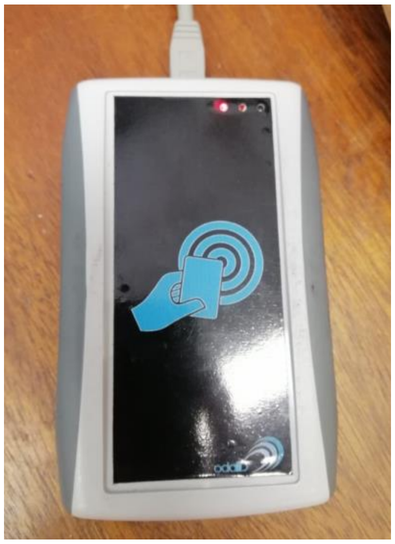
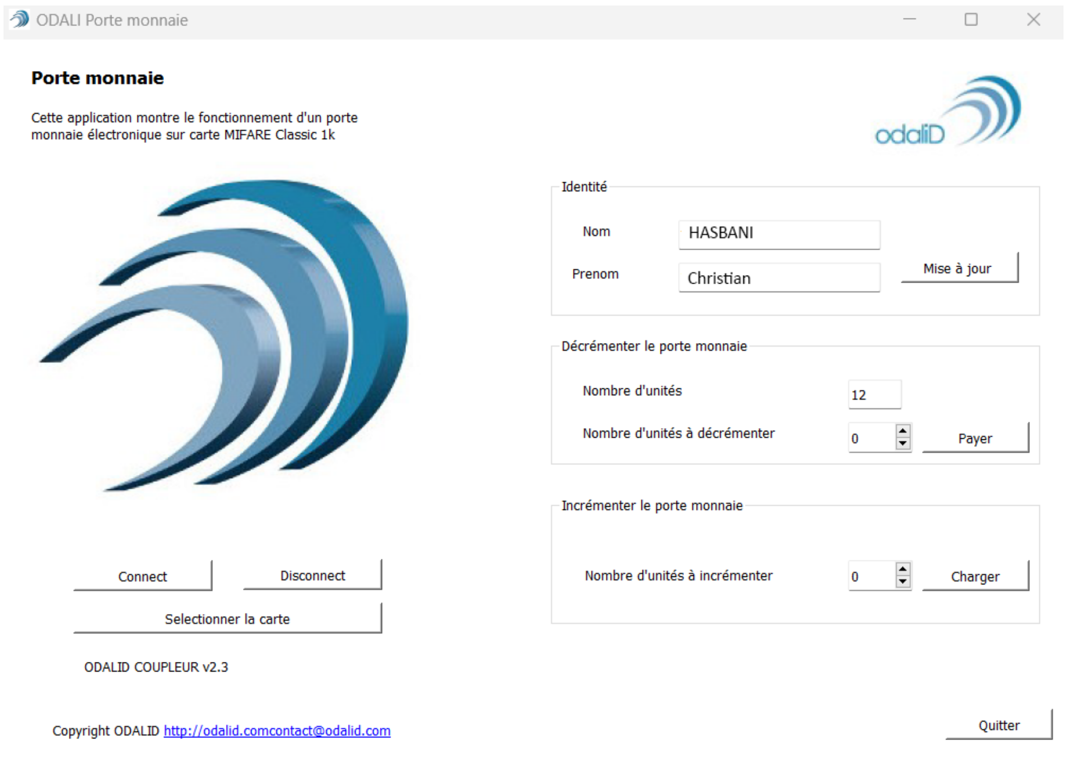


Figure 16-Reader after failing to read the card

After that we check the code of the read function in Figure 7 which recalls the initCard function to reinitialize the reader to try again and read the card.

IV. Update first name and last name

In this part we let the user update his/her first name and last name in the “Identité” section in the GUI



ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité

Nom: HASBANI

Prenom: Christian

Mise à jour

Décrémenter le porte monnaie

Nombre d'unités: 12

Nombre d'unités à décrémenter: 0

Payer

Incrémenter le porte monnaie

Nombre d'unités à incrémenter: 0

Charger

Connect

Disconnect

Selectionner la carte

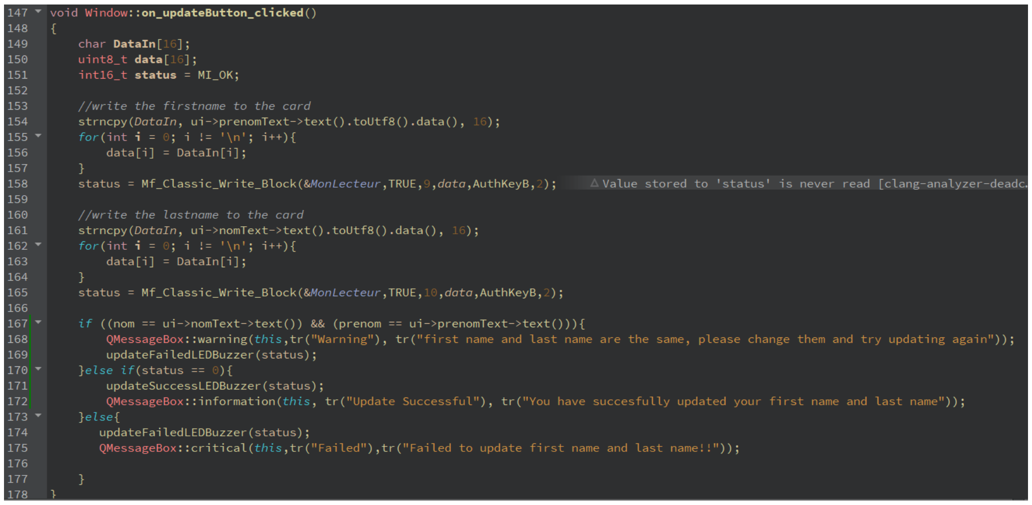
ODALID COUPLEUR v2.3

Copyright ODALID http://odalid.comcontact@odalid.com

Quitter

Figure 17-GUI before updating the first name and last name

Here I changed the first name to “Christian” and last name to “HASBANI” and pressed on the update button “Mise à jour” button in the GUI which will call the following function in the code



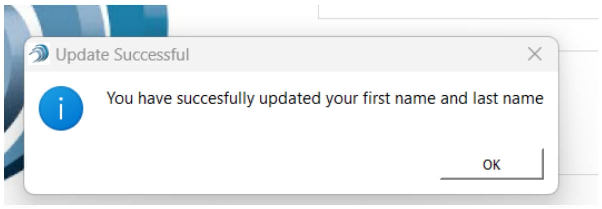
```
void Window::on_updateButton_clicked()
{
    char DataIn[16];
    uint8_t data[16];
    int16_t status = MI_OK;

    //write the firstname to the card
    strncpy(DataIn, ui->prenomText->text().toUtf8().data(), 16);
    for(int i = 0; i != '\n'; i++){
        data[i] = DataIn[i];
    }
    status = Mf_Classic_Write_Block(&MonLecteur,TRUE,9,data,AuthKeyB,2);    ⚠ Value stored to 'status' is never read [clang-analyzer-deadc

    //write the lastname to the card
    strncpy(DataIn, ui->nomText->text().toUtf8().data(), 16);
    for(int i = 0; i != '\n'; i++){
        data[i] = DataIn[i];
    }
    status = Mf_Classic_Write_Block(&MonLecteur,TRUE,10,data,AuthKeyB,2);

    if ((nom == ui->nomText->text()) && (prenom == ui->prenomText->text())){
        QMessageBox::warning(this,tr("Warning"), tr("first name and last name are the same, please change them and try updating again"));
        updateFailedLEDBuzzer(status);
    }else if(status == 0){
        updateSuccessLEDBuzzer(status);
        QMessageBox::information(this, tr("Update Successful"), tr("You have succesfully updated your first name and last name"));
    }else{
       updateFailedLEDBuzzer(status);
       QMessageBox::critical(this,tr("Failed"),tr("Failed to update first name and last name!!"));

    }
}
```

Figure 18-update first name and last name function code

we start by copying the text that was written in the GUI text field to a variable then using a for loop to get an int array containing the ASCII code for each character that was written after that this array is written on the card using a function called “Mf_Classic_Write_Block” using authentication key B for writing same thing for the last name except we change the sector we are writing to.

a) Writing to the card was successful

In this case the “Mf_Classic_Write_Block” function returned 0 indicating that the operation of writing data was successful on the card. So, we call the “updateSuccessLEDBuzzer” function that was explained earlier, and we display a message to let the user know that the update was successful.



You have succesfully updated your first name and last name

Figure 19-update first name and last name successful GUI

Also, since we called the “updateSuccessLEDBuzzer” function the reader will show the following

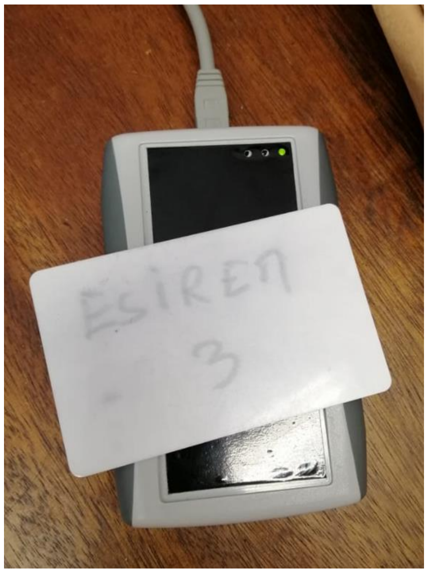


Figure 20-Reader after update first name and last name successful

b) The user didn’t change the first name and last name and pressed update

In this case the user pressed update even through his first name and last name are still the same on the card so a warning will pop up showing to let the user know he didn’t change neither his first name nor his last name.

Naturally because we call the “updateFailedLEDBuzzer” function the reader will display a red LED and multiple buzzer sounds indicating that the update didn’t happen.

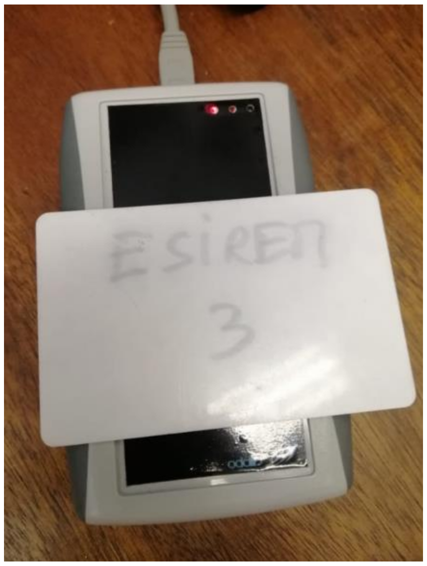


Figure 21-reader showing it failed up to update the card because of the user trying to rewrite existing data

c) Update failed

In the case that the update failed for whatever reason and the “Mf_Classic_Write_Block” returns a different value than “MI_OK” then a CRITICAL message will appear to the user to let him/her know that there was an unknown error during the update process and the “updateFailedLEDBuzzer” function is called also which result in red LED displayed on the reader and multiple buzzer sounds to let the user know that the update failed.

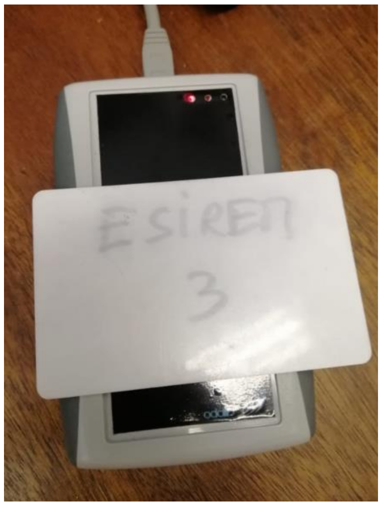
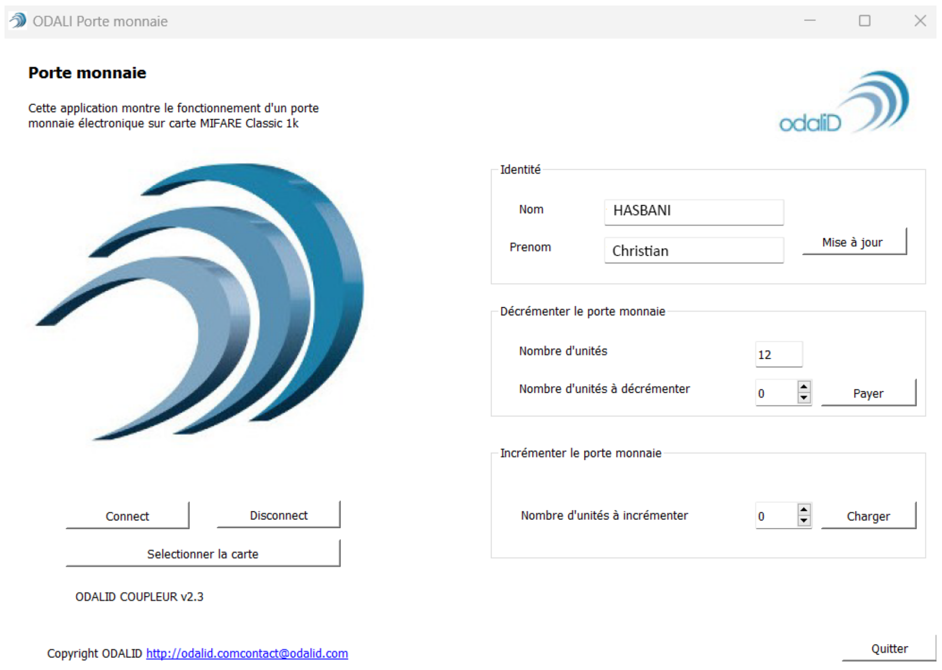


Figure 22-Reader showing it failed to update

V. Paying in the application

Here the user wants to pay, this can be initiated by pressing the “Payer” button in the GUI



Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité
Nom: HASBANI
Prenom: Christian

Décrémenter le porte monnaie
Nombre d'unités: 12
Nombre d'unités à décrémenter: 0

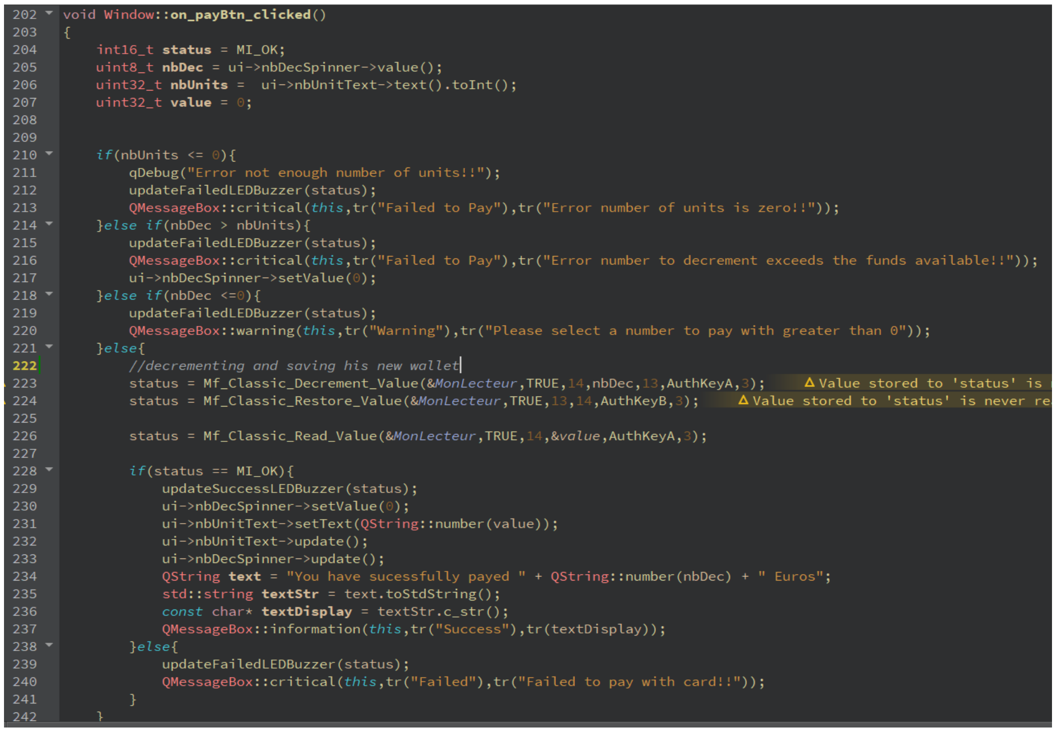
Incrémenter le porte monnaie
Nombre d'unités à incrémenter: 0

ODALID COUPLEUR v2.3

Copyright ODALID http://odalid.comcontact@odalid.com

Figure 23-Before pressing the pay button

Here we choose to pay 2 so we increase the spinner to the value of 2, then we press the “payer” button to call the pay function in the code.



```
202  void Window::on_payBtn_clicked()
203  {
204      int16_t status = MI_OK;
205      uint8_t nbDec = ui->nbDecSpinner->value();
206      uint32_t nbUnits =  ui->nbUnitText->text().toInt();
207      uint32_t value = 0;
208
209
210      if(nbUnits <= 0){
211          qDebug("Error not enough number of units!!");
212          updateFailedLEDBuzzer(status);
213          QMessageBox::critical(this,tr("Failed to Pay"),tr("Error number of units is zero!!"));
214      }else if(nbDec > nbUnits){
215          updateFailedLEDBuzzer(status);
216          QMessageBox::critical(this,tr("Failed to Pay"),tr("Error number to decrement exceeds the funds available!!"));
217          ui->nbDecSpinner->setValue(0);
218      }else if(nbDec <=0){
219          updateFailedLEDBuzzer(status);
220          QMessageBox::warning(this,tr("Warning"),tr("Please select a number to pay with greater than 0"));
221      }else{
222          //decrementing and saving his new wallet
223          status = Mf_Classic_Decrement_Value(&MonLecteur,TRUE,14,nbDec,13,AuthKeyA,3);     ⚠ Value stored to 'status' is
224          status = Mf_Classic_Restore_Value(&MonLecteur,TRUE,13,14,AuthKeyB,3);    ⚠ Value stored to 'status' is never re
225
226          status = Mf_Classic_Read_Value(&MonLecteur,TRUE,14,&value,AuthKeyA,3);
227
228          if(status == MI_OK){
229              updateSuccessLEDBuzzer(status);
230              ui->nbDecSpinner->setValue(0);
231              ui->nbUnitText->setText(QString::number(value));
232              ui->nbUnitText->update();
233              ui->nbDecSpinner->update();
234              QString text = "You have sucessfully payed " + QString::number(nbDec) + " Euros";
235              std::string textStr = text.toStdString();
236              const char* textDisplay = textStr.c_str();
237              QMessageBox::information(this,tr("Success"),tr(textDisplay));
238          }else{
239              updateFailedLEDBuzzer(status);
240              QMessageBox::critical(this,tr("Failed"),tr("Failed to pay with card!!"));
241          }
242      }
```

Figure 24-Pay button function code

a) The payment was successful

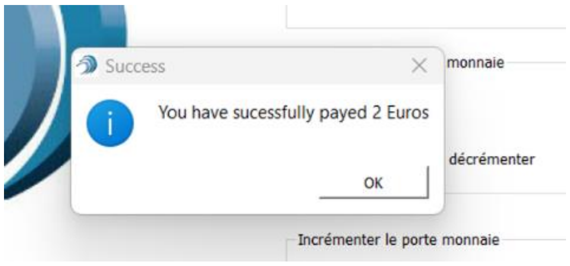
The payment was made which in this code is starting in the “else” at line 222 of figure 24 whereby we will use these functions.

In screenshot above we will use the “Mf_Classic_Decrement_Value” function to decrement the counter which is stored in block 14 sector 3 using the value stored in block 13 (memory) read from the GUI after the user enter it and saved in the “nbDec” variable using authentication key A. After that we use the “Mf_Classic_Restore_Value” function to copy the new value of the counter after using the decrement function earlier that is stored in block 13 (memory) to block 14 sector 3 using authentication key B. Finally, we use the “Mf_Classic_Read_Value” function to read the new saved counter in block 14 sector 3 using authentication key A to the variable “value” to be used in the GUI for displaying the new wallet.

If the returned values from the “Mf_Classic_Read_Value” function is “MI_OK” then we proceed to call the “updateSuccessLEDBuzzer” and refresh the values of the text fields in the GUI.

Else in case that the read failed of the compter then we will call the “updateFailedLEDBuzzer” and display a message to let the user know it failed.

Here we will let the user know the payment was successful and the amount that was paid in small message box.



You have sucessfully payed 2 Euros

Figure 25-Payment successful GUI

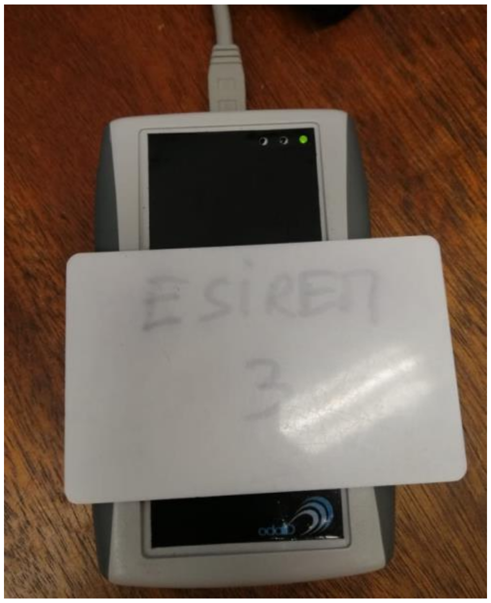
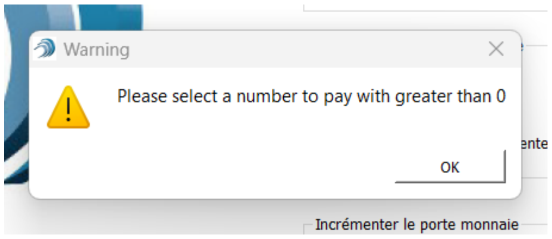


Figure 26-Reader showing that the payment was successful

And of course, since we called the "updateSuccessLEDBuzzer" function then the reader will make a buzzer sound and change the LED to green for a moment to let the user know the payment was successful.

b) The number to decrement is zero

Here if the user doesn't change number to decrement meaning he doesn't choose an amount to pay the user will get a warning telling him that there was no payment made since he didn't choose a number to pay with greater than zero.


Warning

Please select a number to pay with greater than 0

Figure 27-Warning showing that the user has not entered a value to pay with

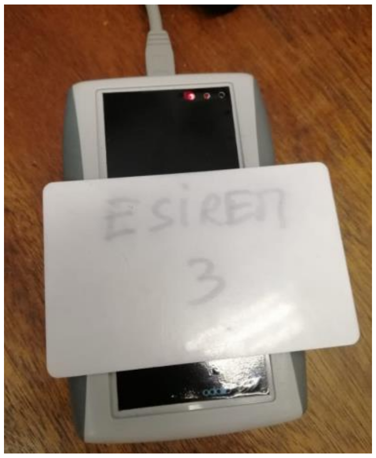
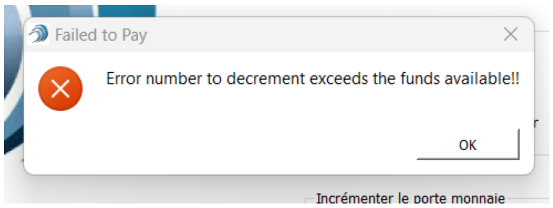


Figure 28-Reader showing the payment failed with a red LED

c) Trying to pay with an amount bigger than the user’s wallet or

In this case a CRITICAL error message will appear indicating that the payment will not take place and calling the “updateFailedLEDBuzzer” function that will show the reader with a red LED and multiple buzzer’s sounds to let the user know it failed to pay.


Failed to Pay

Error number to decrement exceeds the funds available!!

Figure 29-Failed to pay due to payment number exceeding the available number of units

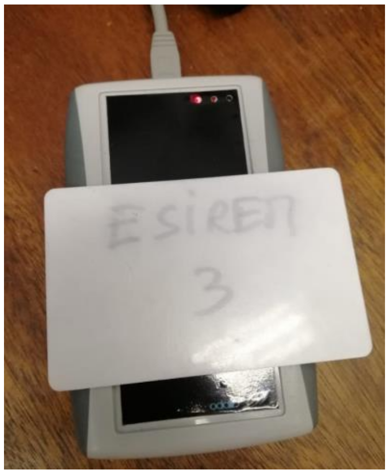
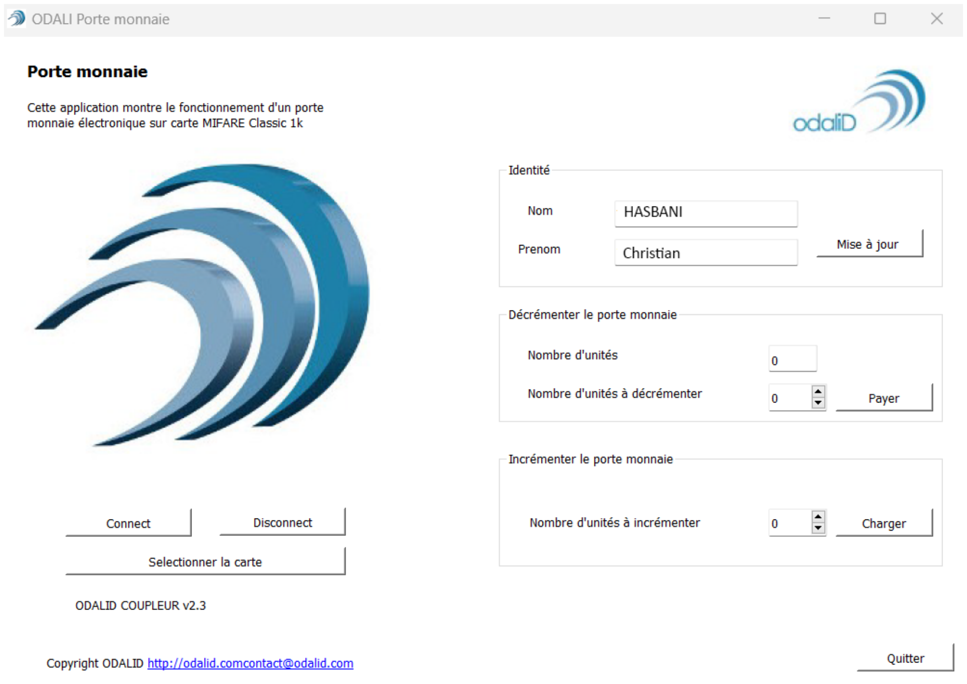


Figure 30-Reader showing that the payment failed

d) User tries to pay with an empty wallet

Here when the user tries to pay when he has an empty wallet like in this picture


ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité

Nom: HASBANI

Prenom: Christian

Mise à jour

Décrémenter le porte monnaie

Nombre d'unités: 0

Nombre d'unités à décrémenter: 0

Payer

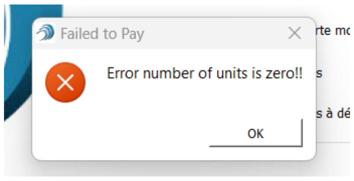
Incrémenter le porte monnaie

Nombre d'unités à incrémenter: 0

Charger

Figure 31-Trying to pay with an empty wallet GUI

This will go in the first if statement in figure 24 which checks the number of units if it's 0 or lower and the user will see the following in his application


Failed to Pay

Error number of units is zero!!

Figure 32-Trying to pay with an empty wallet after pressing payer GUI

And the user will see that the reader will also show a red LED with multiple buzzer sounds indicating that the payment failed

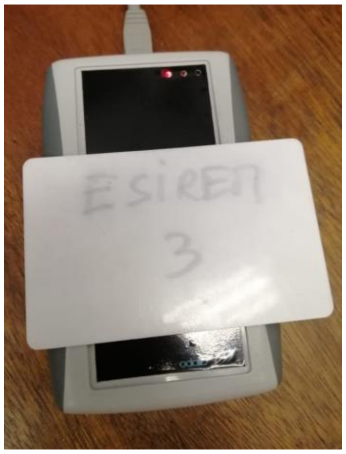
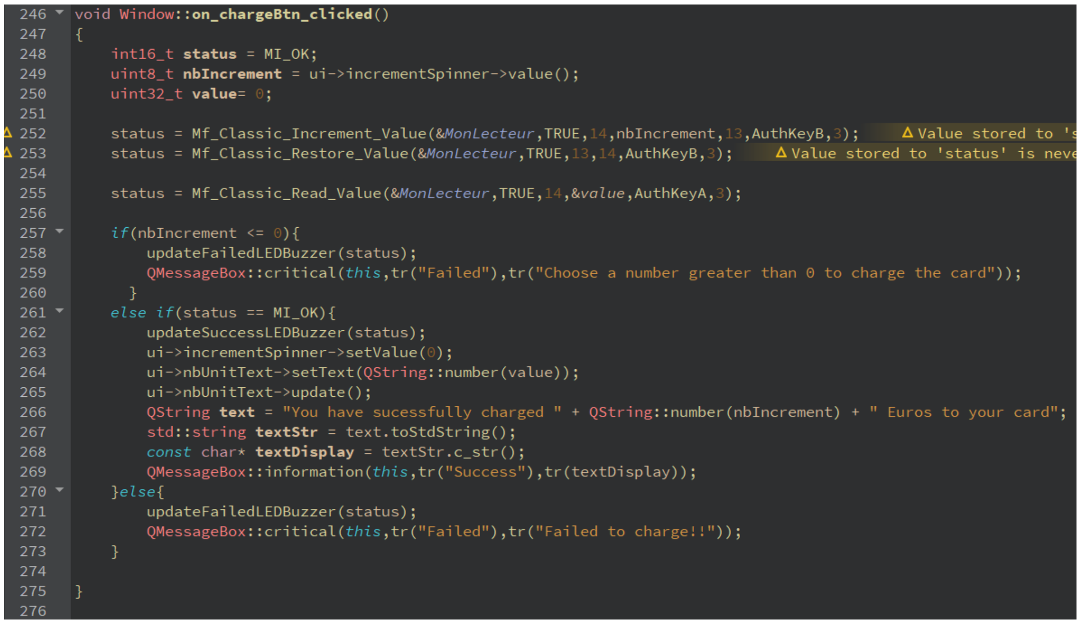


Figure 33-Trying to pay with an empty wallet after pressing payer reader

VI. Recharge wallet

To call the recharge function the user must press on the “charger” button in the GUI.



```cpp
void Window::on_chargeBtn_clicked()
{
    int16_t status = MI_OK;
    uint8_t nbIncrement = ui->incrementSpinner->value();
    uint32_t value= 0;

    status = Mf_Classic_Increment_Value(&MonLecteur,TRUE,14,nbIncrement,13,AuthKeyB,3);     Δ Value stored to 's
    status = Mf_Classic_Restore_Value(&MonLecteur,TRUE,13,14,AuthKeyB,3);     Δ Value stored to 'status' is neve

    status = Mf_Classic_Read_Value(&MonLecteur,TRUE,14,&value,AuthKeyA,3);

    if(nbIncrement <= 0){
        updateFailedLEDBuzzer(status);
        QMessageBox::critical(this,tr("Failed"),tr("Choose a number greater than 0 to charge the card"));
      }
    else if(status == MI_OK){
        updateSuccessLEDBuzzer(status);
        ui->incrementSpinner->setValue(0);
        ui->nbUnitText->setText(QString::number(value));
        ui->nbUnitText->update();
        QString text = "You have sucessfully charged " + QString::number(nbIncrement) + " Euros to your card";
        std::string textStr = text.toStdString();
        const char* textDisplay = textStr.c_str();
        QMessageBox::information(this,tr("Success"),tr(textDisplay));
    }else{
        updateFailedLEDBuzzer(status);
        QMessageBox::critical(this,tr("Failed"),tr("Failed to charge!!"));
    }

}
```

Figure 34-charger function code

In this code we start by read the value entered by the user in the Increment box in the GUI, then we use the function “Mf_Classic_Increment_Value” which makes it possible to increment the value of the counter in block 14 sector 3 by adding the number of units requested that are stored in block 13 (memory) after the user enter a number to be saved in the “nbIncrement” variable using authentication key B. After that we use the “Mf_Classic_Restore_Value” function to write the new value of the counter located in block 13 (memory) after the incrementation process in block 14 sector 3 using authentication key B. Finally, we read the value of the counter using “Mf_Classic_Read_Value” function into the variable “value” to be used later on in the GUI to let the user see his new wallet amount.

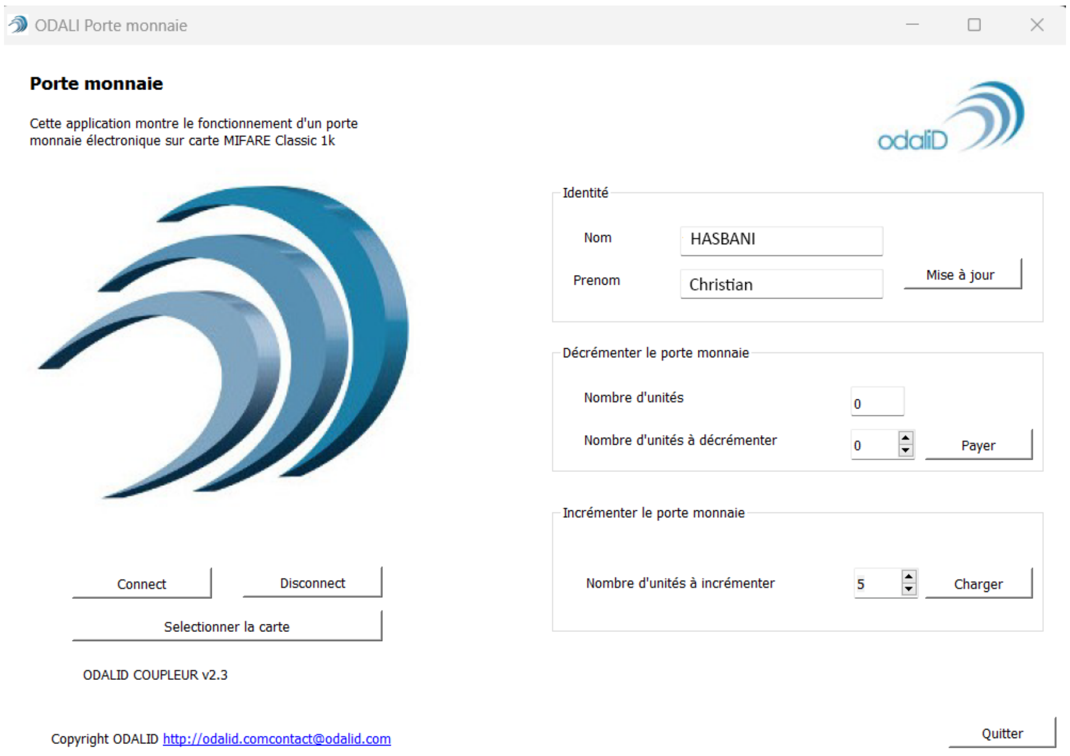
Then we check if the number of units is 0 or less than 0, we call the “updateFailedLEDBuzzer” and show a message that the recharging process failed.

Else we will check if the status from the “Mf_Classic_Read_Value” is “MI_OK” then we will call the “updateSuccessLEDBuzzer” then we update the GUI and refresh the inputs, finally we show a message that the recharging process was successful with the amount displayed in front of the user.

Else if faced any unknown error we will call the “updateFailedLEDBuzzer” and show a critical error message for the user.

a) The recharge process is successful

In this case let us suppose that the user enters 5 units to recharge and presses on the “charger” button in the GUI


ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité

Nom: HASBANI

Prenom: Christian

Mise à jour

Décrémenter le porte monnaie

Nombre d'unités: 0

Nombre d'unités à décrémenter: 0

Payer

Incrémenter le porte monnaie

Nombre d'unités à incrémenter: 5

Charger

Connect | Disconnect

Selectionner la carte

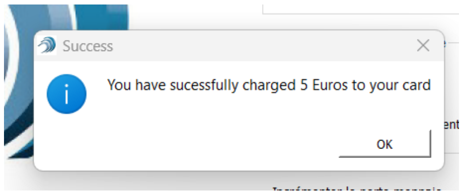
ODALID COUPLEUR v2.3

Copyright ODALID http://odalid.comcontact@odalid.com

Quitter

Figure 35-Before pressing charger GUI

Then once the user presses “charger” button the charge function will be called, and the function will go in the “else if” statement on line 261 in figure 34 and procced to recharge the wallet successfully as we can see in the following picture.


Success

You have sucessfully charged 5 Euros to your card

OK

Figure 36-After pressing the charger button GUI successful recharge

And the reader will show the green LED with one buzzer sound indicating that the recharging of his/her wallet was successful.

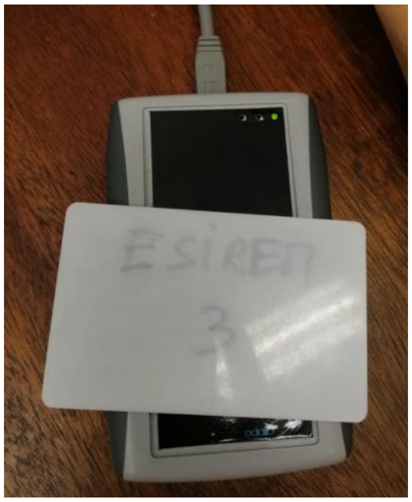
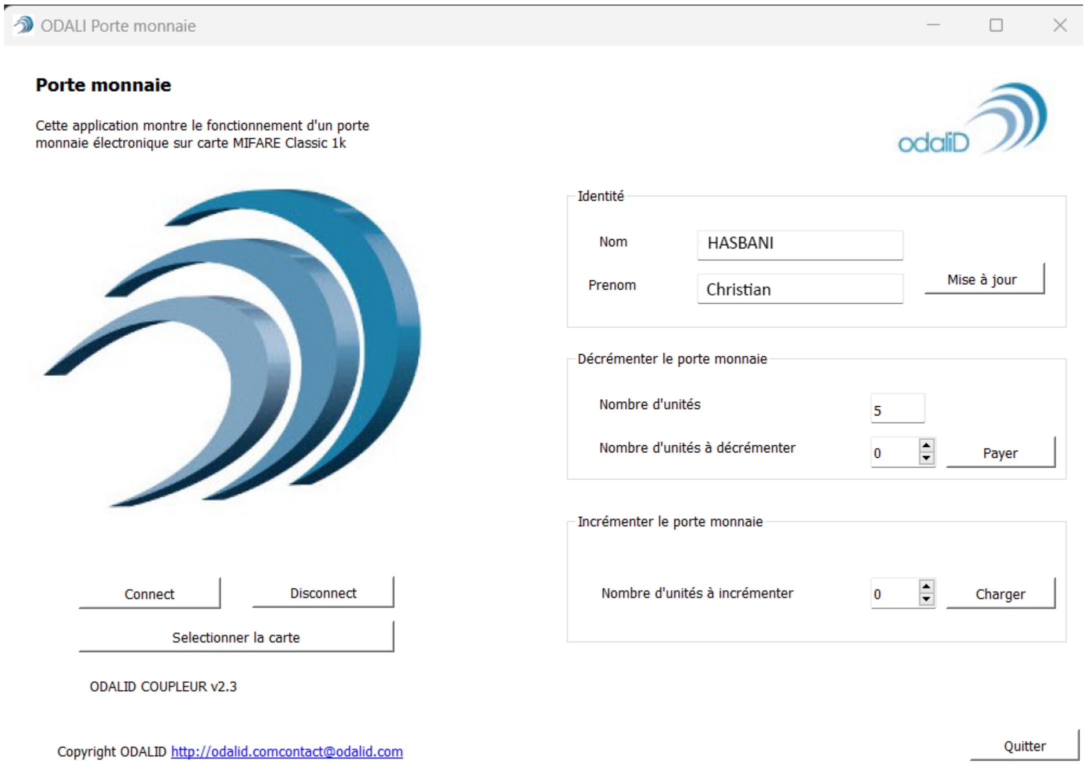


Figure 37-Reader after a successful wallet recharge

b) The recharge process failed

In this case the user tries to recharge with 0 units which doesn't make any sense



ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité

Nom: HASBANI

Prenom: Christian

Mise à jour

Décrémenter le porte monnaie

Nombre d'unités: 5

Nombre d'unités à décrémenter: 0

Payer

Incrémenter le porte monnaie

Nombre d'unités à incrémenter: 0

Charger

Connect

Disconnect

Selectionner la carte

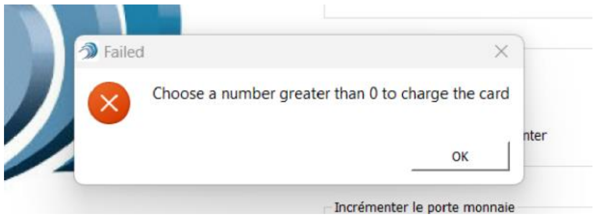
ODALID COUPLEUR v2.3

Copyright ODALID http://odalid.comcontact@odalid.com

Quitter

Figure 38-Trying to recharge with 0 units

Here the user will get a message to let him/her know that they can't recharge with 0 units they need to increment at least by 1 unit to do that.



Failed

Choose a number greater than 0 to charge the card

Figure 39-Trying to recharge with 0 units GUI

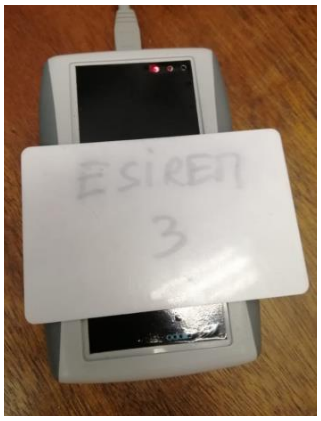
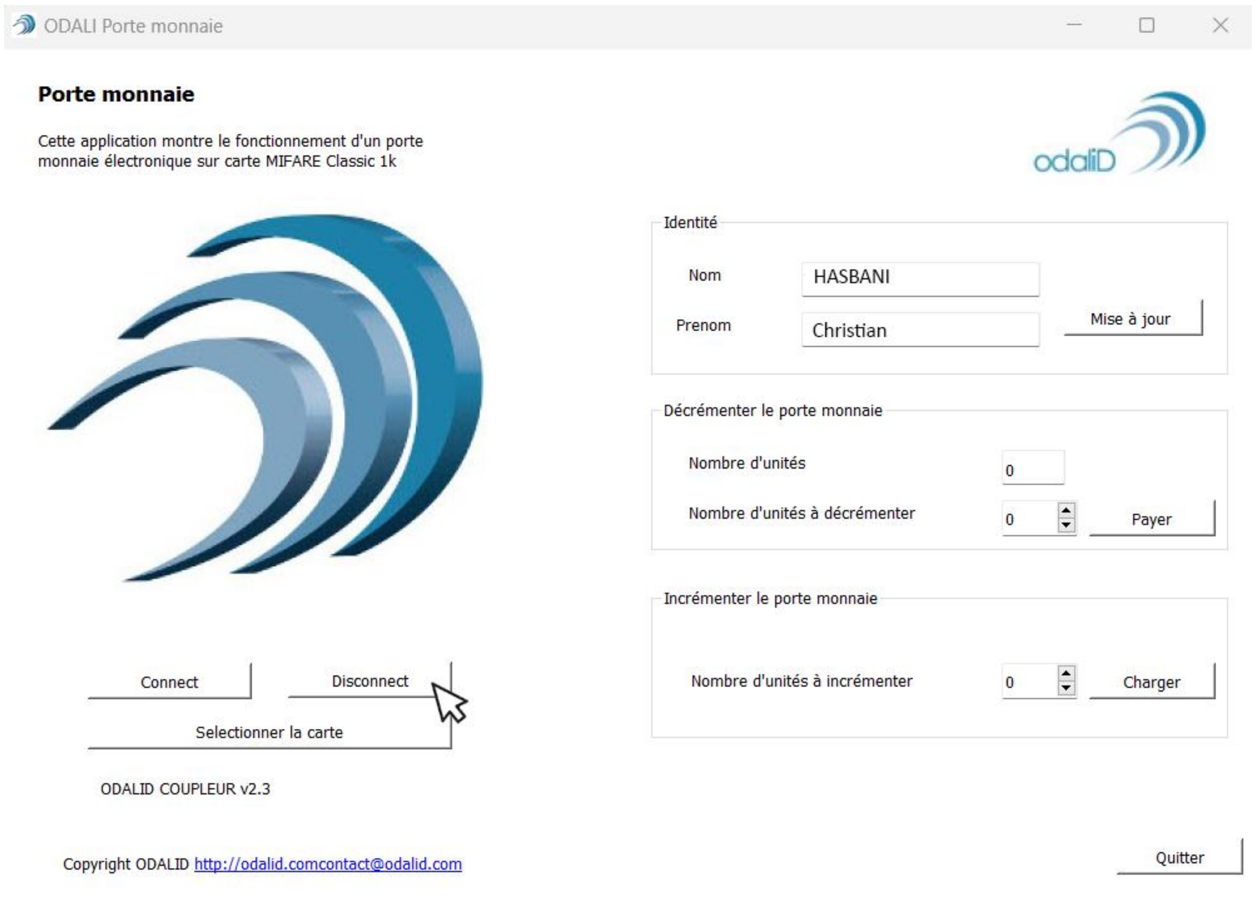


Figure 40-Trying to recharge with 0 units reader failed

And of course, here we called the "updateFailedLEDBuzzer" function which lets the reader turn red LED on and multiple buzzer sounds to let the user know that the charging process failed.

VII. Disconnecting the reader

To disconnect the reader from the laptop the user presses on the disconnect button


ODALI Porte monnaie

Porte monnaie

Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k

Identité

Nom: HASBANI

Prenom: Christian

Mise à jour

Décrémenter le porte monnaie

Nombre d'unités: 0

Nombre d'unités à décrémenter: 0

Payer

Incrémenter le porte monnaie

Nombre d'unités à incrémenter: 0

Charger

Connect | Disconnect

Selectionner la carte

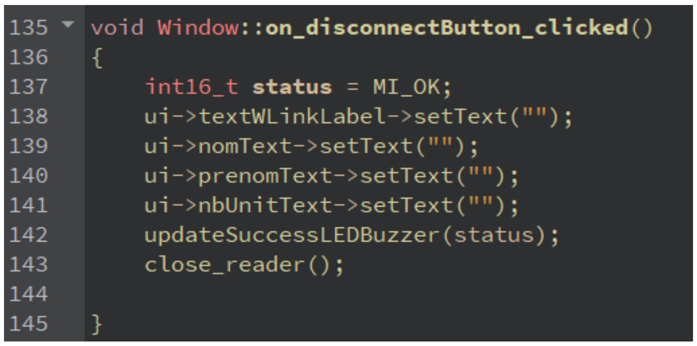
ODALID COUPLEUR v2.3

Copyright ODALID http://odalid.comcontact@odalid.com

Quitter

Figure 41-Pressing on the disconnect button GUI

Once the user presses this button the following code will be executed



```
void Window::on_disconnectButton_clicked()
{
    int16_t status = MI_OK;
    ui->textWLinkLabel->setText("");
    ui->nomText->setText("");
    ui->prenomText->setText("");
    ui->nbUnitText->setText("");
    updateSuccessLEDBuzzer(status);
    close_reader();

}
```

Figure 42-Disconnect function code

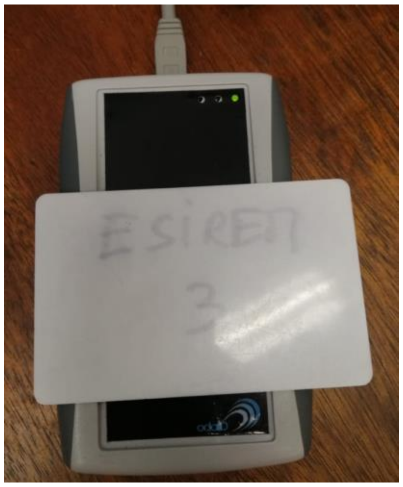
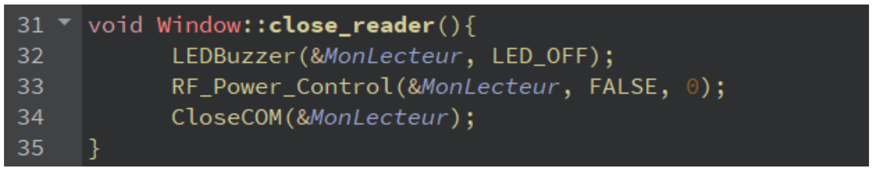


Figure 43-Disconnection successful reader

In this code we are resetting the GUI and calling the "updateSuccessLEDBuzzer" function to turn the LED green on the reader and make a sound with the buzzer and also "close_reader" function which can be seen in the next picture.



```
void Window::close_reader(){
        LEDBuzzer(&MonLecteur, LED_OFF);
        RF_Power_Control(&MonLecteur, FALSE, 0);
        CloseCOM(&MonLecteur);
}
```

Figure 44-close_reader function code

This function will turn the reader off and close all communication with the reader

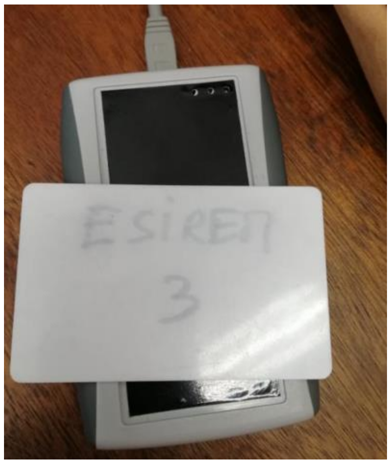


Figure 45-After closing all communication with the reader

VIII. Quitting the application

When the user presses the “Quitter” button in the GUI

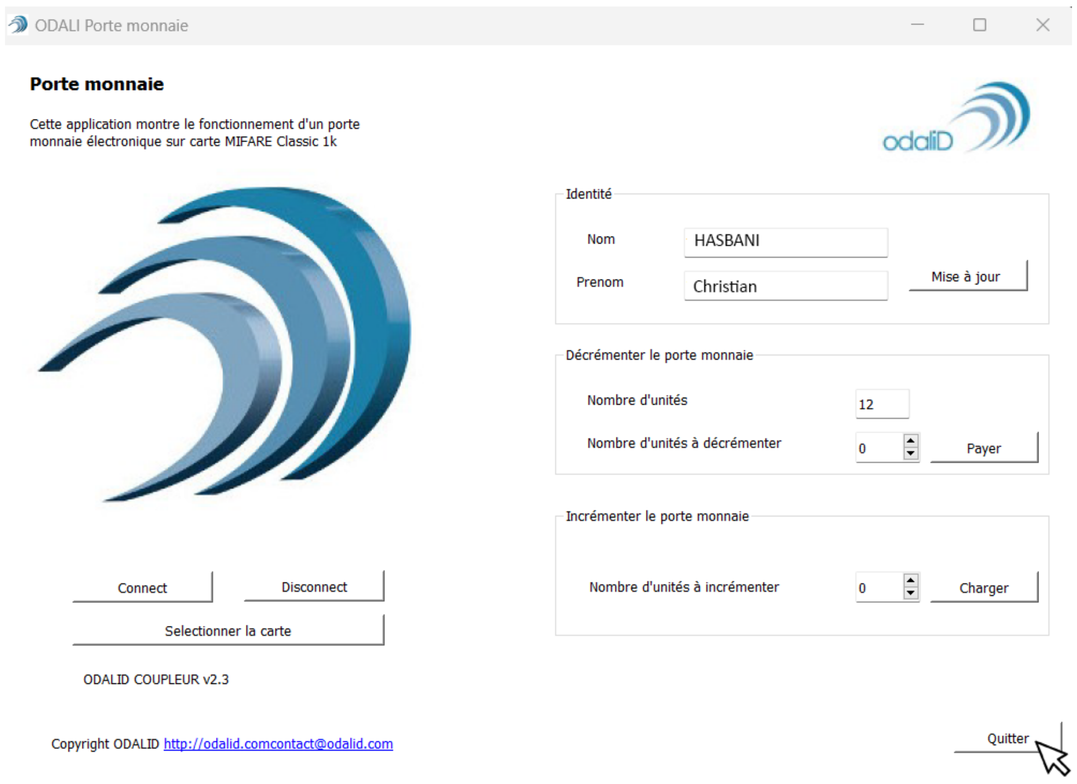
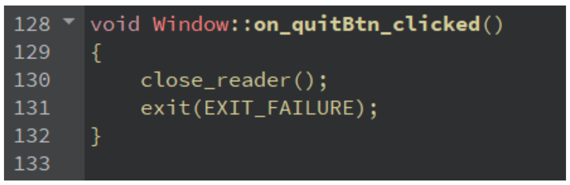

ODALI Porte monnaie
Porte monnaie
Cette application montre le fonctionnement d'un porte monnaie électronique sur carte MIFARE Classic 1k
odaliD
Identité
Nom
HASBANI
Prenom
Christian
Mise à jour
Décrémenter le porte monnaie
Nombre d'unités
12
Nombre d'unités à décrémenter
0
Payer
Incrémenter le porte monnaie
Nombre d'unités à incrémenter
0
Charger
Connect
Disconnect
Selectionner la carte
ODALID COUPLEUR v2.3
Copyright ODALID http://odalid.comcontact@odalid.com
Quitter

Figure 46-User pressing the quit button on the GUI

This button will call the quit function in the code



```
void Window::on_quitBtn_clicked()
{
    close_reader();
    exit(EXIT_FAILURE);
}
```

Figure 47-Quit function code

Here we call the “close_reader” function which was explained earlier to close the communication with the reader and then call the “exit” function to close the application completely.

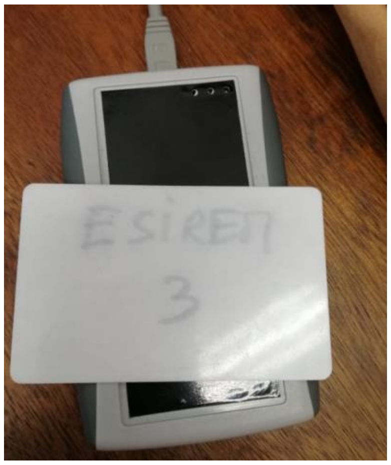


Figure 48-Reader after quiting the application

After we call the "close_reader" function this will turn off all LED lights in the reader.

Table of Figures