

# **Development of a Test Environment for Evaluation of NFC Devices**

Aoife Coady

K00242185

A Final Year Project submitted in partial fulfilment of  
the requirements of Limerick Institute of Technology  
for the degree of Bachelor of Science (Honours) in  
Software Development

Supervised by:

Mr Gerry Guinane

May 2022


# FYP DECLARATION



**TUS**  
**Midwest**  
Department of  
Information Technology

<b>Student Name:</b>  <b>Aoife Coady</b>	<b>Date:</b>  <b>04/05/2022</b>
<b>Department:</b>  <b>Information Technology</b>	<b>Award Title</b>  <b>B. Sc. in Software Development</b>
<b>Title:</b>  <b>Development of a Test Environment for Evaluation of NFC Devices</b>	<b>Supervisor:</b>  <b>Gerry Guinane</b>

I wish to confirm that my FYP submission document and software is my own unaided effort and that I have acknowledged all sources of information and ideas used in this submission.

Signed:  \_\_\_\_\_ Date: 04/05/2022

## **Acknowledgements**

I would like to acknowledge my lecturers that helped me through my 4 years of college, I wouldn't think I'd get this far without them. I'd also like to acknowledge my supervisor for helping me get through my project.

I'd like to acknowledge my family and friends that were kind and who encouraged me into finishing my 4<sup>th</sup> year of college.

## **Abstract**

Developing a program in C# that goes through different test environments to understand NFC devices and their limits. The limits that are tested are the device distances and what material would block the transfer of data. I made my own equipment for this project that goes through different distances in centimetres, I also made material that covers the devices so I could test for connection blockage. There were changes made during the process that would lead to the final program. A few ideas had to be taken out because of lack of time to solve the problems that I discovered. It was a big learning experience for me, under stressful conditions and being able to learn about NFC devices.

## Contents

Figures.....	7
1 Literature Review Chapter.....	10
1.1 Introduction .....	10
1.1.1 Research.....	10
1.1.2 Technical.....	10
1.2 Near Field Communicator (NFC) .....	10
1.3 NFC vs Radio Frequency Identification (RFID).....	10
1.4 Uses for NFC.....	11
1.5 NFC Modes .....	11
1.6 Encoding NFCs .....	12
1.6.1 NFC Data Exchange Format (NDEF).....	12
1.6.2 NFC Record and Payload.....	13
1.7 NFC Application Programme Library.....	13
1.8 Securing data transfer.....	13
1.9 Conclusion.....	14
2 Analysis & Design Chapter .....	15
2.1 Target .....	15
2.2 Requirements.....	15
2.2.1 Use Case Diagram.....	15
2.2.2 List of Actors .....	15
2.2.3 List of Use Cases .....	15
2.3 Application .....	15
2.4 Data Storage .....	16
2.4.1 Database.....	16
2.5 Detailed Specification .....	16
2.5.1 NFC Connection .....	17

2.5.2	Test Function .....	17
3	Implementation Chapter.....	22
3.1	Introduction .....	22
3.2	Technical Information .....	22
3.3	Project Management.....	22
3.3.1	Deadlines.....	23
3.3.2	Material .....	23
3.4	Design Decisions.....	25
3.4.1	ER Diagram .....	26
3.4.2	Main window .....	26
3.5	System Structure .....	26
3.5.1	NFC Library.....	27
3.5.2	NuGet NFC-ACR122U Library.....	28
3.5.3	Management Object .....	28
3.5.4	Database Structure .....	28
3.6	Completed System (how it works).....	29
3.7	Discussion .....	31
4	Testing Chapter.....	32
4.1	Experimental Testing .....	32
4.1.1	NuGet NFC-ACR122U Library.....	32
4.1.2	NFC Connection Window Prototype .....	33
4.1.3	Gathering Information .....	33
4.2	Verification/Validation.....	34
4.3	Software Testing .....	35
4.3.1	Unit Testing .....	35
5	Conclusion .....	36
	References.....	38

## Figures

FIGURE 1: NDEF .....	13
FIGURE 2: USER CASE.....	15
FIGURE 3: ER DIAGRAM .....	16
FIGURE 4: NFC CONNECTION WIREFRAME.....	17
FIGURE 5: DISTANCE WIREFRAME.....	18
FIGURE 6: MATERIAL WIREFRAME.....	19
FIGURE 7: INTERCEPTION WIREFRAME.....	20
FIGURE 8: RESULT DISTANCE WIREFRAME.....	20
FIGURE 9: RESULTS MATERIAL WIREFRAME.....	21
FIGURE 10: RESULTS INTERCEPTION WIREFRAME .....	21
FIGURE 11: NFC CARD READER .....	23
FIGURE 12: STUDENT CARD.....	24
FIGURE 13: LEAP CARD .....	24
FIGURE 14: DISTANCE TESTING EQUIPMENT .....	24
FIGURE 15: CARDBOARD.....	24
FIGURE 16: TIN FOIL .....	25
FIGURE 17: PLASTIC.....	25
FIGURE 18: USER CASE CHANGES.....	25
FIGURE 19: CHANGES ER DIAGRAM.....	26
FIGURE 20: MAIN WINDOW .....	26
FIGURE 21: SYSTEM STRUCTURE .....	27
FIGURE 22: MT LICENSE.....	27
FIGURE 23: NFC CONNECTING CODE .....	27
FIGURE 24: NUGET LIBRARY.....	28
FIGURE 25: WIN32_PNP CODE .....	28
FIGURE 26: DATABASE CODE.....	29
FIGURE 27: COMPLETED MAIN WINDOW .....	29
FIGURE 28: TABLE OF TAG TYPES .....	30
FIGURE 29:TAG TYPE SELECTION .....	30
FIGURE 30: DISTANCE WINDOW .....	30
FIGURE 31: MATERIAL WINDOW .....	31
FIGURE 32:EXPERIMENTAL TESTING NUGET LIBRARY .....	32
FIGURE 33: EXPERIMENTAL TESTING NUGET LIBRARY .....	32
FIGURE 34: EXPERIMENTAL TESTING NFC CONNECTION .....	33
FIGURE 35: EXPERIMENTAL TESTING TEST WINDOW .....	33
FIGURE 36: TABLE OF SUCCESS RATE FOR DISTANCE .....	34

FIGURE 37: GRAPH FOR SUCCESS RATE OF DISTANCES.....	34
FIGURE 38: TABLE OF SUCCESS RATE FOR MATERIAL BLOCKAGE.....	34
FIGURE 39: TRACEABILITY MATRIX .....	34
FIGURE 40: NUNIT INSTALLATION .....	35
FIGURE 41: DATABASE CONNECTION .....	35
FIGURE 42: TESTING DATABASE COMPONENTS .....	35



### List of Abbreviations

TUS	Technological University of the Shannon
NFC	Near Field Communication
RFID	Radio Frequency Identification
WPF	Window Presentation Foundation
ISO	International Standards Organization
NDEF	NFC Data Exchange Format
SE	Secure Element
TSM	Trusted Service Manager
UML	Unified Modelling Language
ER	Entity Relationship
UID	Unique Identification Number

# **1 Literature Review Chapter**

## **1.1 Introduction**

### **1.1.1 Research**

Research and investigation into the safety and security of Near Field Communication (NFC) and how it is used with today's technology. This is getting a feeling of how safe it is. What cryptographic method does it use – can someone just easily steal data from it or hence your phone? Types of ways it's used – credit card machines, leap cards, and student id cards.

Understanding what NFC is, what are the advantages that it holds, why it is useful, and see what future projects there are being developed with it. Comparing NFC with radio frequency identification (RFID) and learning how they are both used.

### **1.1.2 Technical**

Making a Desktop Windows Presentation Foundation (WPF) application (C#) that will be connecting to a read/write NFC USB stick that will collect the data from. This will help with finding out the limits of NFC first-hand, testing the capacity, and studying the concept of Near-Field Communication.

Using SQL to store the logs for this project – will help with testing and shows the user where the project is failing and hopefully why. Use it to log data that is retrieved, if it can.

Friendly User Interface, user/application/NFC (Reader/Writer)/NFC Tags, NFC library.

Demonstration of the application working and being able to hit the necessary challenges that will be faced.

## **1.2 Near Field Communicator (NFC)**

Near Field Communicator was invented in 2002 but didn't come into acceptance until 2003 by the International Standards Organization (ISO). It is a peer-to-peer communicator that uses radio frequencies to transfer or receive information. NFC chip is made up of some small memory storage, radio chip, and an antenna. This chip does not need a power source, when it is in range of an NFC reading device, it picks up the electromagnetic waves and it transfers the data that is embedded on the chip to the phone or device used. The max range for the transaction to take place is 10cm/4in. (Blue Bite, 2020) If the NFC tag or NFC enabled device is not in this proximity it would remain in sleep mode.

## **1.3 NFC vs Radio Frequency Identification (RFID)**

RFID was invented in the 1940s, the inventor Mario W. Cardullo didn't get a patent until 1973. RFID readers tend to be expensive. The RFIDs are used for tracking and logistics. RFID has

three frequency ranges: Low 125 to 134kHz, High 13.56MHz, Ultra High 856 MHz to 960 MHz.

There are 2 types of RFID – Active and Passive. Active tags have their own power source which can broadcast a read range up to 100 meters. Passive tags get their power from electromagnetic energy transmitters from an RFID reader. Their signal can go up to 25 meters (2500 cm).

NFC operates at the same frequency as the High-Frequency RFID which operates at 13.56 MHz frequency. The NFC device can act as the reader and as the tag making it peer-to-peer communication. (Thrasher, 2013)

The NFC-enabled device sends power and commands to the tag. Which response with data. NFC- enabled devices (Smartphones) can write data to the tags using a special command. You can update tags with new data.

#### **1.4 Uses for NFC**

The Card payment phone generates a single-use transaction key that can only be used once and expires within a second. – making it hard for anyone to give unauthorized payments to the account.

Pairing with devices by keeping passwords on the device that help pair the two devices, keeps the passwords safe so the user doesn't forget them or write them down and forget about them. Connecting Smartphones with headphones, cameras, and other equipment that uses Bluetooth to connect them with – makes it easier and faster.

Gaming – The Nintendo Switch has an NFC touchpoint located on the right stick – this connects an 'amiibo figure' to the Nintendo. Amiibo figure allows the user to get new characters, game modes, or other perks.

Exchanging contact information, business cards and in 2006 it was used with Smart Posters. These smart posters were a way for the customer to get more information i.e., brings them to the website.

When interacting between the smartphones, the first phone becomes a reader and the other becomes a card, this switch once the information has been transferred.

#### **1.5 NFC Modes**

Three NFC modes

1. Card emulation: NFC enabled devices that can mimic bank cards, loyalty cards, hotel pass cards, travel cards (Leap cards), etc.

2. Read and write interaction with various information sources i.e. smart Posters. Receiving information from these also can be written to them using an NFC-enabled device.
3. Peer-to-peer communication: connectivity between two NFC-enabled devices.

## **1.6 Encoding NFCs**

Encoding is the process of adding data to NFC – a variety of types that can be encoded

- Encoding URL
- Telephone number or a small ID number

### **1.6.1 NFC Data Exchange Format (NDEF)**

NDEF encoding as it's known, helps with communication. The universal way of storing NFC information means that almost any NFC-enabled device will be able to read and understand the data and what type of data it is. Some only can store a sentence of text. NTAG210 NFC chip can only store a URL up to 40 characters long (Seritag, 2020)

To be compatible with NFC devices, the NFC Forum has specified the compatibility of five different tag types. (Christiansen, et al., 2021)

#### **1.6.1.1 Tag Type Technical Specifications**

**Type 1:** The ISO/IEC 14443-3A standard is used to create this tag (NFC-A). These tags can be converted to become read-only or rewritable. The memory size ranges from 96 bytes to 2 kilobytes. The data transfer rate is 106 kbit/s. These tags, unlike all others, have no anti-collision protection when dealing with numerous tags in the NFC field. (Christiansen, et al., 2021)

**Type 2:** The ISO/IEC 14443-3A standard is used to generate this tag (NFC-A). The tags can be adjusted to become read-only or rewritable. The memory size ranges from 48 bytes to 2 kilobytes. The data transfer rate is 106 kbit/s. (Christiansen, et al., 2021)

**Type 3:** This tag is based on the FeliCa-based Japanese Industrial Standard (JIS) X 6319-4 (ISO/IEC 18092). The tags are either rewritable or read-only by default. A total of 2 kilobytes of memory are available. The communication speed is 212 kbits per second or 424 kbits per second. (Christiansen, et al., 2021)

**Type 4:** This tag is based on ISO/IEC 14443-4 A/B (NFC A, NFC B) and can communicate using either NFC-A or NFC-B. Furthermore, the tag may enable ISO-DEP (Data Exchange Protocol defined in ISO/IEC 14443 (ISO/IEC 14443-4:2008 Part 4: Transmission Protocol) as an optional feature. The tags are either rewritable or read-only by default. Up to 32 kilobytes

of variable memory. Three alternative communication speeds are supported: 106, 212, and 424 kbit/s. (Christiansen, et al., 2021)

**Type 5:** This tag is based on ISO/IEC 15693 (NFC-V), and it allows you to read and write an NDEF message on an ISO/IEC 15693 RF tag which can also be read by long-range RFID readers. NFC transmission is confined to short distances and may employ ISO/IEC 18092's Active Communication Mode, in which the sending peer creates the field, balancing power usage and improving connection stability. Up to 64 kilobytes of variable memory. The data transfer rate is 26.48 kbit/s. (Christiansen, et al., 2021)

### 1.6.2 NFC Record and Payload

An NDEF message contains an NDEF record. Each record is a binary structure that includes both a data payload and type information. It also contains information on the data's structure, such as payload size and if the data is chunked across many records.

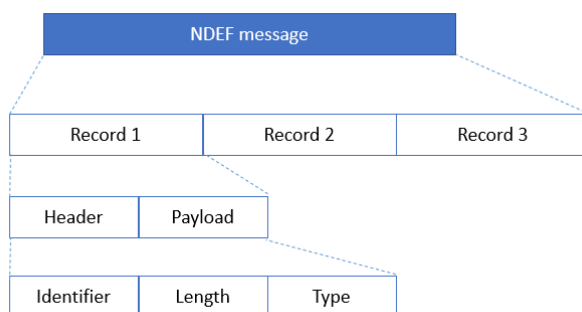


Figure 1: NDEF

#### 1.6.2.1 Payload

Payload is where the data is stored

ID Length – length of the ID field in bytes

Type – tells us the kind of data the payload contains

ID- this is the field where applications need to identify the whole payload carried within an NDEF record.

### 1.7 NFC Application Programme Library

NFC-ACR122U library is a read and write library in the NuGet extensions

### 1.8 Securing data transfer

Secure Element (SE) is enabled on smartphones. This protects the data when it is on the phone like an on-Device attack.

Trusted Service Manager (TSM) encrypts the data that's received from the bank.

There are three main security features:

1 – Distance: hard to receive any information from an NFC device only if in proximity. It would be very hard to get close to a person or object without being noticed.

2 – User input: for credit cards the NFC function doesn't work until a PIN or face recognition is used.

3. Validation element: there's a secure element chip within the device that validates the purchase or use of the NFC chip. A unique signature is assigned to every purchase making it secured. (WORLDPAY EDITORIAL TEAM, 2019)

Types of ways of making the NFC secure on phones are by turning off the NFC section on the smartphone and having a secure PIN/password to access the ability to use the NFC.

## **1.9 Conclusion**

To conclude NFC is different than RFID, it uses the same frequency to the High-Frequency RFID. There are different tag types for NFC, five in total. These would have different speed of data transfer between the two connections. There are three main security features: distance, user input and validation. These factors would lead to a safe secured tag, but without it would lead to data being stolen.

## 2 Analysis & Design Chapter

### 2.1 Target

To make an application that tests the limits of the NFC reader/writers. To analyse the different distances each NFC tag has. To see if material can block the tags from reading from a reader/writer. To identify if the tags can be intercepted if there are 2 tags on top of each other. These would help understand the types of tags there are and their limits.

### 2.2 Requirements

#### 2.2.1 Use Case Diagram

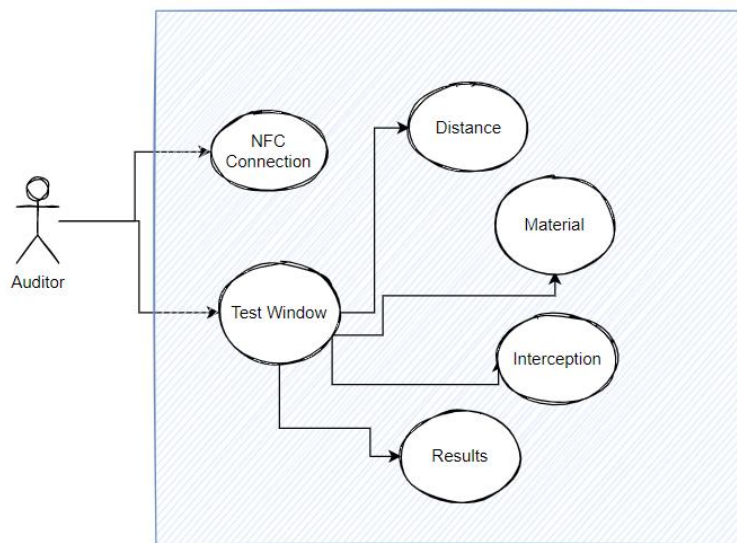


Figure 2: User Case

#### 2.2.2 List of Actors

The auditor – has access to the application to test the procedures.

#### 2.2.3 List of Use Cases

- NFC Connection
- Test Window
- Distance Test
- Material Test
- Interception Test
- Results

### 2.3 Application

C# Desktop application that tests NFC functionality and security.

1. Testing distance (2cm, 5cm, 10cm, 15cm, 20cm, etc.)

2. Testing if there's anything blocking the connection (cotton, metal, wood, cardboard, etc.)
3. Testing for interception i.e. two cards together, which one will connect, will it work?
4. Testing for breakage of encoding
5. Results page – this will display the latest results that were made – will be able to filter.

All the Wireframes in this chapter were made from [Wireframe.cc | The go-to free, online wireframing tool.](https://wireframe.cc/) website.

## 2.4 Data Storage

1. Device version of device, date & time, distance, succeed/failed
2. Device version of device, date & time, material used, succeed/failed
3. Device version of device, date & time, succeed/failed
4. Still thinking
5. Grabbing the data from all these and displaying them.

### 2.4.1 Database

#### 2.4.1.1 Entity Relationship Diagram (ER)

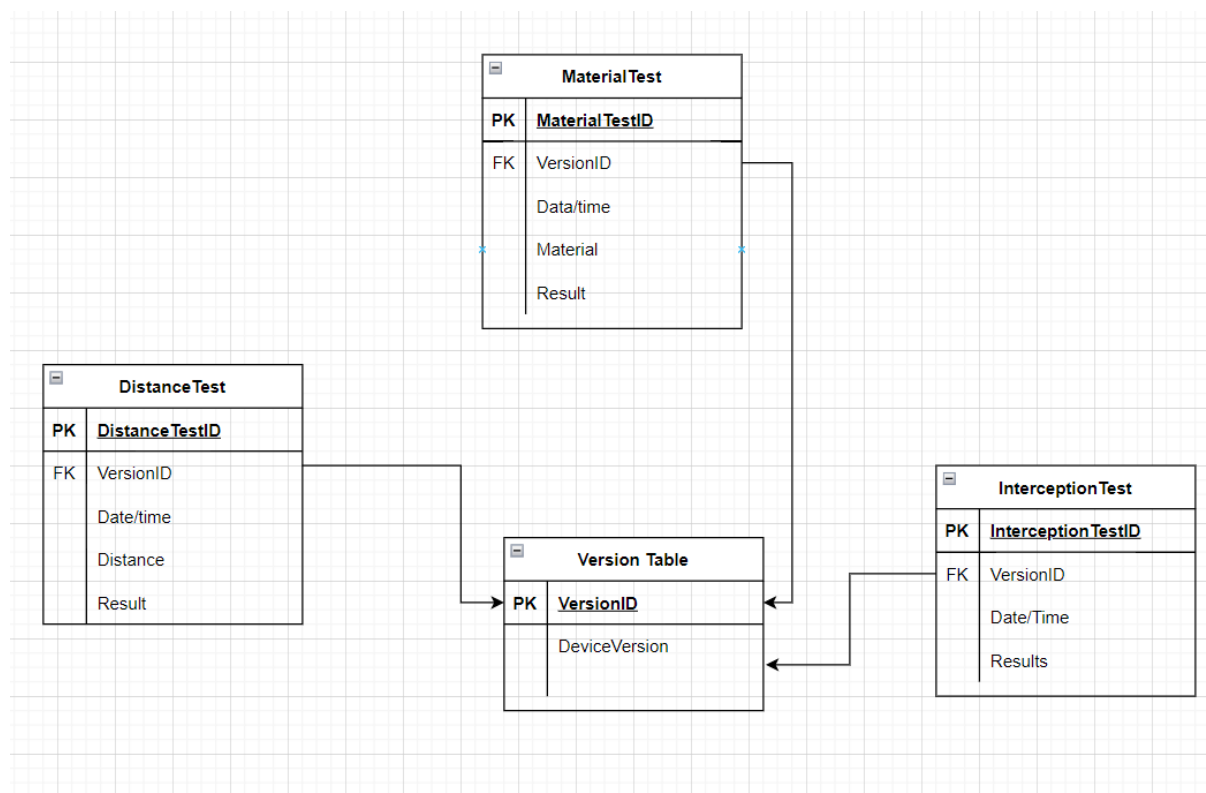


Figure 3: ER Diagram

## 2.5 Detailed Specification

Type(Menu/Button)

Location (page location where function is located)

Action (describe what happens when item is selected)



## 2.5.1 NFC Connection

### 2.5.1.1 Connect

Type: Button

Location: NFC Connection window

Action: connects to the NFC device that will be used for tests, retrieves data from that device.

### 2.5.1.2 Disconnect

Type: Button

Location: NFC Connection window

Action: disconnects the current NFC device from the program.

### 2.5.1.3 List of Devices

Type: List Box

Location: NFC Connection Window

Action: displays the list of smart card readers that can be found by the computer.

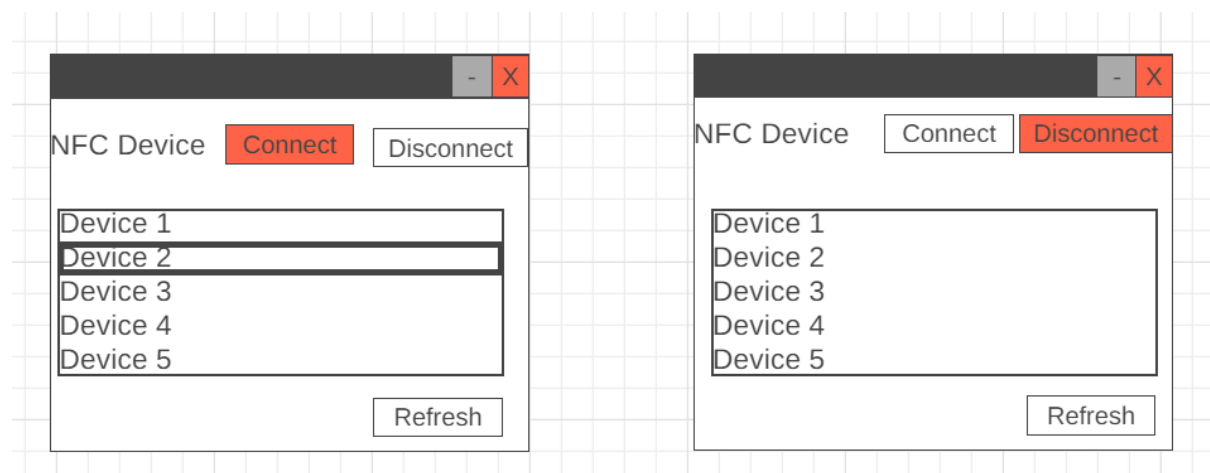


Figure 4: NFC Connection Wireframe

## 2.5.2 Test Function

### 2.5.2.1 NFC Distance Test Application

- This checks to see if the NFC device/tag connects to the reader off the distance given by the user which is using a ruler.

#### 2.5.2.1.1 Device Version

Type: Text Box

Action: Displays the driver version of the device used

#### 2.5.2.1.2 Date/Time

Type: Text Box

Action: Displays the date and time

#### 2.5.2.1.3 Distance

Type: Text Box

Action: User enters the distance that corresponds to the ruler used i.e. 2cm, 5cm.

#### 2.5.2.1.4 Connected

Type: True/False

Action: This sets automatically when testing, defaulted too false

#### 2.5.2.1.5 Result

Type: Success/Failed

Action: This sets automatically when testing, defaulted to Failed.

If Connection is true = result is Success, Connection is false = result is Failed.

#### 2.5.2.1.6 Save Button

Type: Button

Action: Saves results onto the Database

#### 2.5.2.1.7 Wireframe

The wireframe shows a user interface for a distance test application. At the top, there is a navigation bar with four tabs: 'Distance' (highlighted in orange), 'Material', 'Interception', and 'Results'. Below the navigation bar, the interface is divided into two main sections. The left section contains three input fields: 'Device Version' with a text box containing '-----', 'Date & Time' with a text box containing 'DD/MM/YY HH:MM:SS', and 'Distance' with a text box containing 'cm' and a dropdown arrow. The right section contains two input fields: 'Connected' with a text box containing 'True/False', and 'Result' with a text box containing 'Success/Failed'. At the bottom right of the right section, there is a 'Save Button'.

Figure 5: Distance Wireframe

### 2.5.2.2 NFC Material Test Application

- Checks if the NFC device/tag can connect with the material picked by the user that will be in between the reader and device/tag.

#### 2.5.2.2.1 Device Version

Type: Text Box

Action: Displays the driver version of the device used

#### 2.5.2.2.2 Date/Time

Type: Text Box

Action: Displays the date and time

#### 2.5.2.2.3 Material

Type: Text Box

Action: User enters the type of material used in the testing stage.

#### 2.5.2.2.4 Connected

Type: True/False

Action: This sets automatically when testing, defaulted to false

#### 2.5.2.2.5 Result

Type: Success/Failed

Action: This sets automatically when testing, defaulted to Failed.

If Connection is true = result is Failed, Connection is false = result is Success.

#### 2.5.2.2.6 Save Button

Type: Button

Action: Saves results onto the Database

#### 2.5.2.2.7 Wireframe

The wireframe shows a form with four tabs: Distance, Material (active), Interception, and Results. The Material tab contains the following fields:

Device Version:	<input type="text" value="-----"/>	Connected:	<input type="text" value="True/False"/>
Date & Time:	<input type="text" value="DD/MM/YY HH:MM:SS"/>	Result	<input type="text" value="Success/Failed"/>
Material:	<input type="text" value="-----"/>		

Save Button

Figure 6: Material Wireframe

#### 2.5.2.3 NFC Interception Test Application

- Checking if at least one NFC device connects when there's 2 devices over the Reader.

##### 2.5.2.3.1 Device Version

Type: Text Box

Action: Displays the driver version of the device used

##### 2.5.2.3.2 Date/Time

Type: Text Box

Action: Displays the date and time

##### 2.5.2.3.3 Connected

Type: True/False

Action: This sets automatically when testing, defaulted too false

#### 2.5.2.3.4 Result

Type: Success/Failed

Action: This sets automatically when testing, defaulted to Failed.

If Connection is true = result is Success, Connection is false = result is Failed.

#### 2.5.2.3.5 Save Button

Type: Button

Action: Saves results onto the Database

#### 2.5.2.3.6 Wireframe

Distance	Material	Interception	Results
Device Version	<input type="text" value="-----"/>	Connected:	<input type="text" value="True/False"/>
Date & Time:	<input type="text" value="DD/MM/YY HH:MM:SS"/>	Result	<input type="text" value="Success/Failed"/>
<div>Save Button</div>			

Figure 7: Interception Wireframe

#### 2.5.2.4 NFC Result Application

- This shows the results that have been made already

##### 2.5.2.4.1 Distance Navigation bar

This shows the results of ID, Device Version, Date/Time, Distance, Connected & Result

Distance	Material	Interception			
ID	Device Version	Date/Time	Distance	Connected	Results

Figure 8: Result Distance Wireframe

2.5.2.4.2 Material Navigation bar

This shows the results of ID, Device Version, Date/Time, Material, Connected & Result

Distance		Material		Interception		Results
Distance		Material		Interception		
ID	Device Version	Date/Time	Material	Connected	Results	

Figure 9: Results Material Wireframe

2.5.2.4.3 Interception Navigation Bar

This shows the results of ID, Device Version, Serial Number 2, Date/Time, Connection & Result

Distance

Material

Interception

Results

Distance		Material	Interception
ID	Device Version	Results	

Figure 10: Results Interception Wireframe

### **3 Implementation Chapter**

#### **3.1 Introduction**

Implementation of NFC testing ideas on a C# application. The Technical Information section is all about the technology that is used in the making of the application. This explains how and why it was used. The Project Management Section will be discussing the work on how it was done, going through deadlines and expenditure. The Design Quality Section explains the steps into making a creative application. The next section is Design Decisions, this discuss about the changes made during the making of the application, the difficulties found and what outcomes were discovered. System Structure identifies all structures in the application. The last section is called Completed System, this explains the finished work on the application.

#### **3.2 Technical Information**

The integrated development environment (IDE) used is Visual Studio, this allows for written and test code, and also allows for graphic user interface. The languages used in Visual Studio are C++, C# (.NET Framework) and much more. C# is the language that is being used in the application. This was picked because it was handy to use for connecting to the NFC Smart Reader and other information's that can be collected from the inside of the computer, i.e., using Windows Management Instrumentation (WMI) that can be used by admins to monitor information that can be retrieved through PowerShell. WMI is used to collect the information of the Smart Card Readers, the information is used to identify what device is being used for the application.

Windows Presentation Foundation (WPF) is the User Interfaced (UI) used in the application, this is used with .NET Framework. WPF works for making a desktop application, it's very similar to the Windows Forms that is also used with .NET Framework. WPF can be used for creativity in the windows UI, there are many controls that can be put onto the windows, i.e., grids, these are used to display the other controllers in aligned areas, making everything in line. Styles can be used also for the creativity, these help with the height and weight, colour, and font size or font family etc.

The Smart Card Reader is a device that can collect the data from an NFC Tag or items that have NFC Chips in them. All the information that is retrieved will be saved into an SQL Server on the local machine. To test and study these out the SQL Server Management System (SSMS) is used, this application shows all the information retrieved from the database, which is used to store data.

#### **3.3 Project Management**

Managing the workflow of the project and the material needed for the project.

### 3.3.1 Deadlines

These are the deadlines that are made for this project.

	Due Date	Task Completed
Get Primary Material	Before Christmas	Yes
Get Secondary Material	Before February	Yes
NFC Connection Window	4 <sup>th</sup> of February	Yes
- Distance Window	18 <sup>th</sup> of February	Yes
- Material Window	25 <sup>th</sup> of February	Yes
- Interception Window	4 <sup>th</sup> of March	No
- Result	11 <sup>th</sup> of March	Yes
Testing Chapter	1 <sup>st</sup> of April	Yes

### 3.3.2 Material

There are 2 types of material used in this project. Primary which are the main items and secondary which is the items that would help the shape the project.

#### 3.3.2.1 Primary material

The smart card reader is ACR122U which reads and writes the NFC tags that are detected. This is a USB device that is connected to the hardware. There is library on the visual studio NuGet that could be installed to help use of the smart card reader on projects.



Figure 11: NFC Card Reader

The NFC cards that are used in this project are student cards and leap cards shown in Figure 12 and 13.

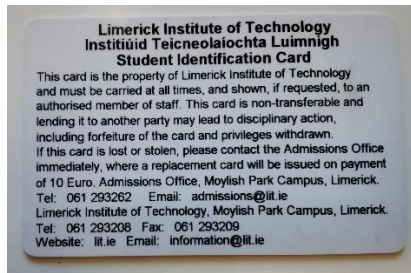


Figure 12: Student Card



Figure 13: Leap Card

The testing equipment has slots where the NFC tag can be placed when the application has asked for it.



Figure 14: Distance Testing Equipment

### 3.3.2.2 Secondary

The ruler is used to outline and measure the distance for the distance material

- Cardboard



Figure 15: Cardboard



- Tin Foil



Figure 16: Tin Foil

- Plastic



Figure 17: Plastic

### 3.4 Design Decisions

There were changes during the implementation stages. The biggest change was the use case diagram as shown in Figure 18. It now has one main window that follows on to three other windows and a text file button for Results.

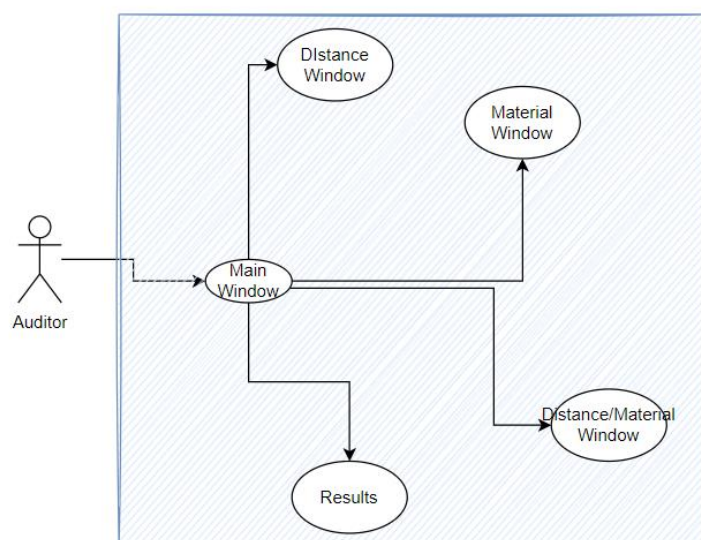


Figure 18: User Case Changes

### 3.4.1 ER Diagram

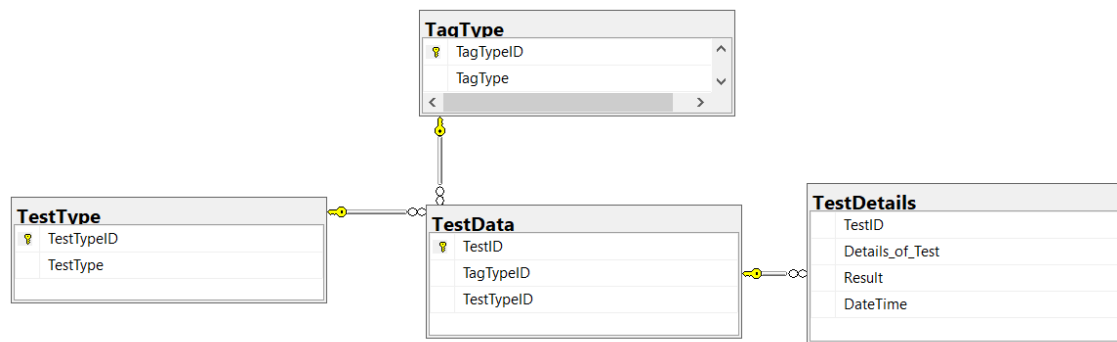


Figure 19: Changes ER Diagram

### 3.4.2 Main window

The main window now gives the actor the options to pick which Reader they are using, the tag type of the tag they are using. The user will have to gather the Chip ID from putting the tag onto the reader that is being used. The user gets an option to pick which test they will be doing in the tag – Distance and Material, they can pick one or both. There is a button named result for the user to get the last result made on this application, it will open a text file with the information.

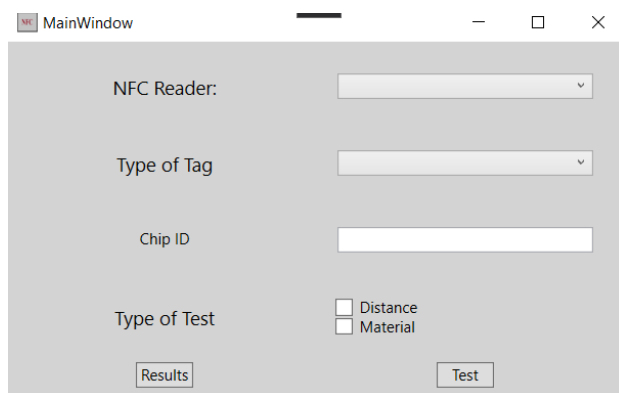


Figure 20: Main Window

## 3.5 System Structure

The solution consists of 4 projects, the DatabaseConnection which connects the database with the main application, the NFCLibrary which helps connect the smart card reader with the main application, NFCTestApplication is the main application that has the WPF interface and the SmartCardReaderRetrieves class that gets the smart card reader description, and the last project is UnitTesting which tests the application with unit tests, discussed in the testing chapter.

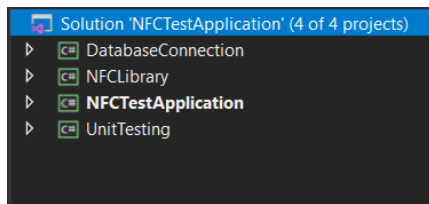


Figure 21: System Structure

### 3.5.1 NFC Library

The NFC Library used is an extracted from h4khba/nfc-reader GitHub. This was used to help connect to the NFC device/tag. This also is licenced by MIT, which allows for Commercial use, Modification, Distribution and Private use.

Copyright notice:

MIT License

Copyright (c) 2019 Hüseyin Akbaş

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Figure 22: MT License

This library is then referenced in the NFCTestApplication project to be used in the distance, material, distance & material windows by implementing the code NFCReader reader = new NFCReader. When the user clicks the check button in this application the reader.Connect() is then used to check if the smart reader is picking up the NFC tag on the device used in the test.

```
1 reference
private void btDCheck_Click(object sender, RoutedEventArgs e)
{
    if (reader.Connect())
    {
        tbDResults.Text = "Success";
        tbDconnection.Text = "True";
    }
    else
    {
        tbDResults.Text = "Failed";
        tbDconnection.Text = "False";
    }
}
```

Figure 23: NFC Connecting Code

### 3.5.2 NuGet NFC-ACR122U Library

In the main window, NFC-ACR122U library is used, this library is to read the chips unique identification number (UID). This is an NuGet Extension that can be installed into Visual Studio projects. When it is in the project, the line `using Sydesoft.NfcDevice;` is used, this connects the class to the project.



Figure 24: NuGet Library

### 3.5.3 Management Object

The class `SmartCardReaderRetrieves` retrieves a list of smart card readers that are connected to the hardware from the `Win32_PNP` Entity. They are all placed in a `ManagementObjectCollection` which is then iterated through to saving the description of the smart card into a list. The list is then passed back to the Main Window.

```
2 references
class SmartCardReaderRetrieves
{
    2 references
    public static List<string> RetrieveSmartReader() {
        List<string> smartReaderName = new List<string>();

        //setting up the ManagementObjectCollection
        ManagementObjectCollection collection;

        //query in the Management ObjectSearcher
        using (ManagementObjectSearcher searcher = new ManagementObjectSearcher(@"Select * From Win32_PNPEntity where Description Like ""%Smart Card Reader%"""))
            collection = searcher.Get();

        foreach (ManagementObject smartReader in collection) {
            smartReaderName.Add((string)smartReader["Description"]);
        }
        collection.Dispose();

        return smartReaderName;
    }
}
```

Figure 25: Win32\_PNP Code

### 3.5.4 Database Structure

Using `System.Data.SqlClient` the program can write and read data from the database which is in the LAPTOP-NSP31V6F server, or the equivalent user's server. `OpenConnection()` is called to open the connection to the `myNFCdb`. Doesn't take in any variables and does not return one either.

```

public void openConnection()
{
    connect = new SqlConnection();
    connect.ConnectionString = @"Data Source=localhost;Initial Catalog=myNFCdb;Integrated Security=True";
    try
    {
        connect.Open();
    }
    catch (System.Exception ex)
    {
        Console.WriteLine("Failer to connect", ex.ToString());
        connect.Close();
        Environment.Exit(0); // Force application to close after failer
    }
}

```

Figure 26: Database Code

### 3.6 Completed System (how it works)

In the first window that executes the user picks the NFC reader that will show up in the first combo box. In the second combo box the user picks what type of tag they are using i.e. Phone or credit card, and if the tag that they haven't used yet is not on the list they can add a new one to the list, which will save to the database.

The user must put the tag on the NFC reader for it to read the UID which is put into the Chip ID text box that is read-only. The user then picks the type of test they are going to do with the tag they are using. They can pick Distance, Material, or both.

Figure 27: Completed Main Window

In the table below there are the types of tags, manufacture of the tags and some type of things the tags are in. This will help the user pick the tag type in the Main Window. This is found in the NFC Application OneNote, under the instructions heading. To find the manufacture of the NFC chip an application on the phone was used “NFC Tools”.

Tag Type 1	Innovision Topaz	
Tag Type 2	NXP MIFARE UL, MIFARE UL-C, NTAG 203,210, 2012, 2013, ETC.	Clothing & Footwear
Tag Type 3	Sony FeliCa(Lite)	Electronic money cards
Tag Type 4	NXP MIFARE DEFire EV1 or EV2, Inside Secure VaultIC 151/161, HID Trusted Tag™, NTAG 413 /424 DNA	Leap Card, Bank card (Revolut)
Tag Type 5 (NFC-V/ISO 15693)	NXP ICODE SLIx family, EM4233, Fujitsu FRAM MB89R118C, MB89R112, HID Vigo™	Gaming Sports
MIFARE Classic/EV1	NXP MIFARE Classic EV1 1K, 4k	Student card Bank card (AIB)

Figure 28: Table of Tag Types

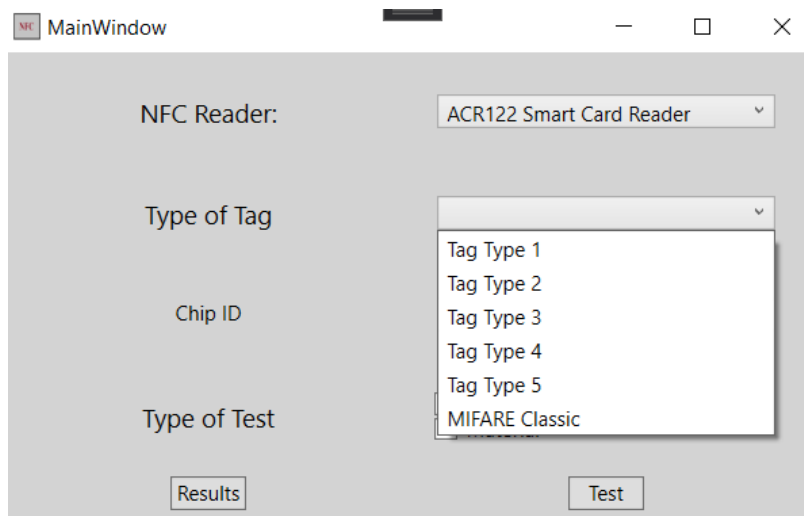


Figure 29: Tag Type Selection

All the data that is put into the main window will be saved into the database.

The next window depends on which test the user picked for the test cases.

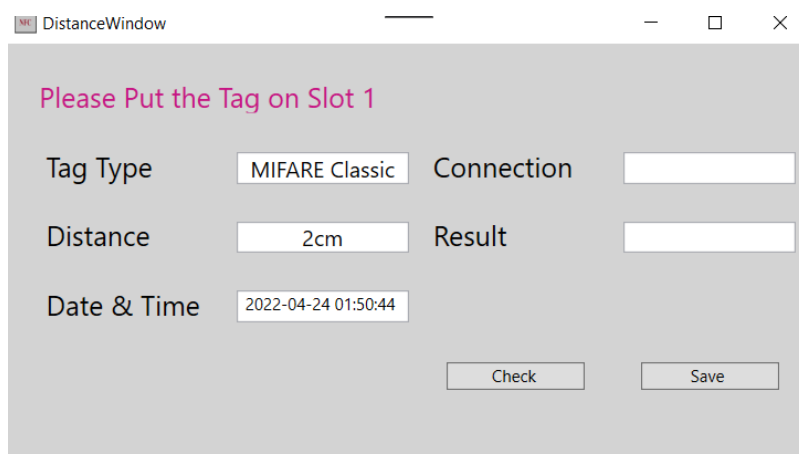


Figure 30: Distance Window

The distance window and the material window have instructions on where to put the tag item on the distance scale and if it needs material with it.

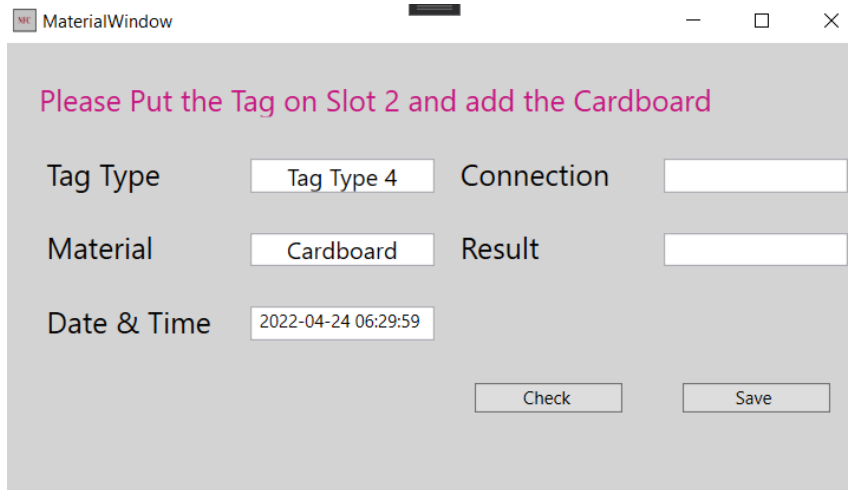
The image shows a software window titled "MaterialWindow". At the top, there is a pink instruction: "Please Put the Tag on Slot 2 and add the Cardboard". Below this, there are three rows of input fields. The first row has "Tag Type" with a dropdown menu showing "Tag Type 4", and "Connection" with an empty text box. The second row has "Material" with a dropdown menu showing "Cardboard", and "Result" with an empty text box. The third row has "Date & Time" with a text box showing "2022-04-24 06:29:59". At the bottom right, there are two buttons: "Check" and "Save".

Figure 31: Material Window

### 3.7 Discussion

The main thing this application needs is to find out the tag type on its own, hence what the “NFC Tool” does where it finds the manufacture of the tag. It would improve the application since you won’t have to use an outside app.

Would add more material to test, it’s quite boring only having 3 materials where only 1 blocks the signal. This would improve testing and the capability of more material to be able to block signal.

Another way to improve the application is having the result button be able to display information better, hence by making another window that displays it like that was in the Analyse and Design chapter.

Making another test window that would help detect interception of two tags, this was the plan when making the wireframes and use cases, but it was quite complicated to make.

These features could be made in future versions, there would be more time to make them and less pressure.

## 4 Testing Chapter

### 4.1 Experimental Testing

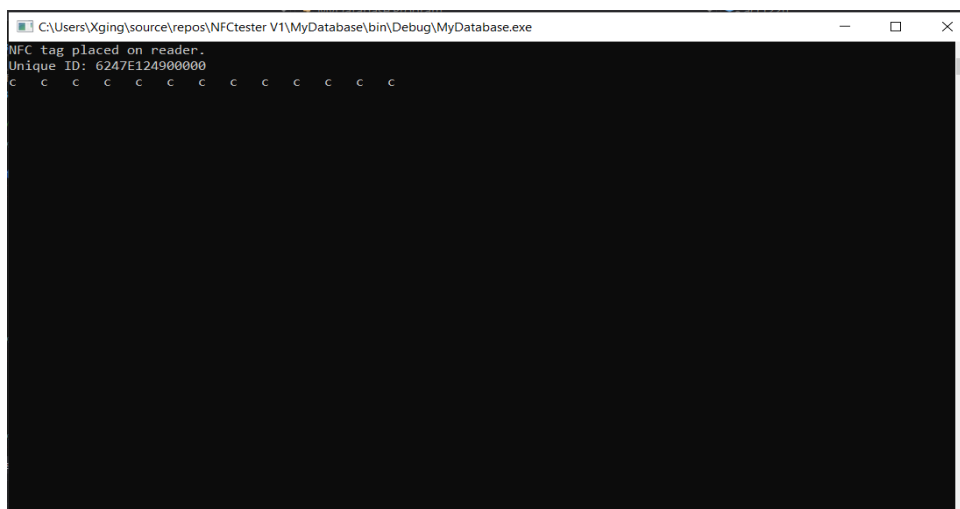
#### 4.1.1 NuGet NFC-ACR122U Library

Experimenting with the NFC extension on visual studio. Using the code in the following figure, it notifies the method when a card is placed on the card reader. The method takes the UID of the NFC tag and displays it, it also reads and displays the data on the card.

```
//private static NdefLibrary NdefLibrary = NdefLibrary0;  
private static Acr122u acr122u = new Acr122u0;  
0 references  
static void Main(string[] args)  
{  
    acr122u.Init(false, 50, 4, 4, 200);  
    acr122u.CardInserted += Acr122u_CardInserted;  
    acr122u.CardRemoved += Acr122u_CardRemoved;  
    Console.ReadLine();  
}  
  
1 reference  
private static void Acr122u_CardRemoved()  
{  
    //do nothing  
}  
  
1 reference  
private static void Acr122u_CardInserted(ICardReader reader)  
{  
    Console.WriteLine("NFC tag placed on reader.");  
    Console.WriteLine("Unique ID: " + BitConverter.ToString(acr122u.GetUID(reader)).Replace("-", ""));  
    var data = acr122u.ReadID;  
    Console.WriteLine( Encoding.UTF8.GetString(acr122u.ReadData(reader)));  
}
```

Figure 32: Experimental Testing NuGet Library

The result of a Limerick Institution of Technology (LIT) student card is displayed like in the figure below. The data didn't give off any information on what's written on the NFC chip and the UID didn't give any information about the chip.



```
C:\Users\Xging\source\repos\NFCtester V1\MyDatabase\bin\Debug\MyDatabase.exe  
NFC tag placed on reader.  
Unique ID: 6247E124900000  
c c c c c c c c c c c c c c c c
```

Figure 33: Experimental Testing NuGet Library



### 4.1.2 NFC Connection Window Prototype

Connecting to the Smart card prototype, was made in less than a few days. This prototype didn't make the final cut, it wasn't getting much valuable information. The information that was received of the tag was then displayed on the window.

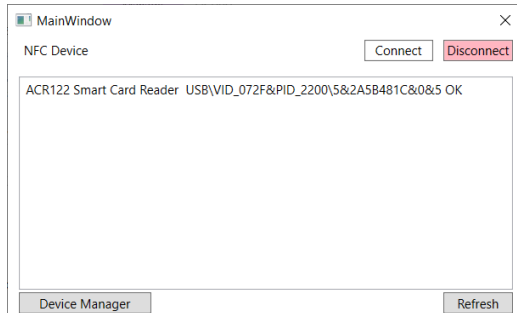


Figure 34: Experimental Testing NFC Connection

The connection button only changed colour when it was pressed, nothing was done in the application. It was the same with the disconnected button. The Device Manager button did execute the Device Manager.

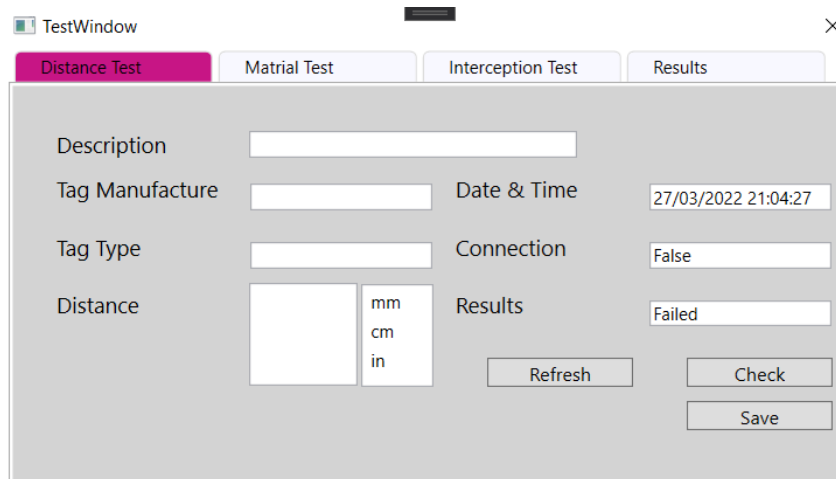


Figure 35: Experimental Testing Test Window

The test window was not giving off the feel of a testing application, this was not a great process. The data had to be put in manually when it should've been automatic by the system.

### 4.1.3 Gathering Information

Using the information that was gathered in the database, by displaying it in a table (Figure 36) and a Graph. This information shows that every type of tags has different distances. Also found out that the phones UID changes every time it connects to the Reader/writer.

UID	Tag Type	2cm	5cm	7cm	10cm	15cm	20cm
04-33-20-5A-9F-5B-80	Tag Type 4	Success	Failed	Failed	Failed	Failed	Failed
04-6C-49-AA-0D-5D-80	Tag Type 4	Success	Failed	Failed	Failed	Failed	Failed
85-6A-F6-BA-90-00-00	MIFARE Classic	Success	Success	Failed	Failed	Failed	Failed
08-04-D1-06-90-00-00	Tag Type 2	Success	Success	Success	Failed	Failed	Failed

Figure 36: Table of Success Rate for Distance

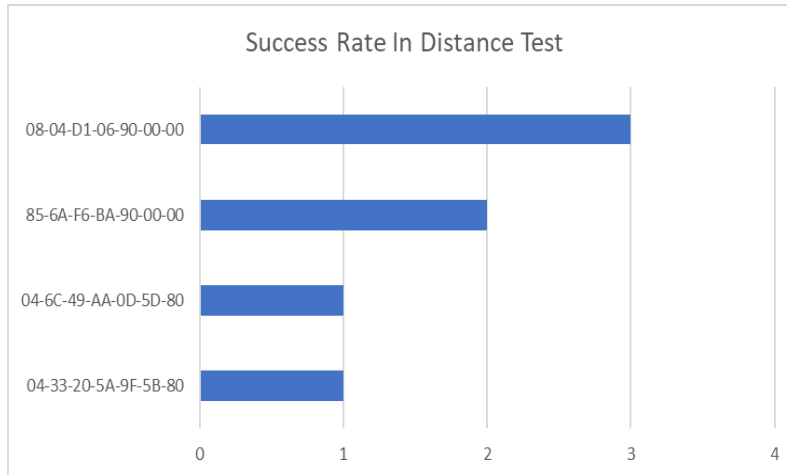


Figure 37: Graph for Success Rate of Distances

More information gathered was in the Material test. It doesn't matter what tag type is used, its all about the material. The material that stops the frequency waves is tin foil. In Figure 38 the table shows the success of the blockage.

UID	Tag Type	Cardboard	Tin Foil	Plastic
04-6C-49-AA-0D-5D-80	Tag Type 4	Failed	Success	Failed
62-47-E1-24-90-00-00	MIFARE Classic	Failed	Success	Failed

Figure 38: Table of Success Rate for Material Blockage

## 4.2 Verification/Validation

Using the test data that was received in the tests, a Traceability Matrix could be made (Figure 39). This shows the different type of test made for the test data, and the different requirements that were met.

Type of Test	Test	UID	2cm	5cm	7cm	10cm	15cm	12cm	Cardboard	Tin foil	Plastic
Distance	Test 1	04-33-20-5A-9F-5B-80	x								
	Test 2	04-6C-49-AA-0D-5D-80	x								
	Test 3	85-6A-F6-BA-90-00-00	x	x							
	Test 4	08-04-D1-06-90-00-00	x	x	x						
Material	Test 5	04-6C-49-AA-0D-5D-80								x	
Distance/Material	Test 6	62-47-E1-24-90-00-00	x	x						x	

Figure 39: Traceability Matrix

## 4.3 Software Testing

### 4.3.1 Unit Testing

Testing involves installed NUnit, which can be found in NuGet in the Visual studio IDE, into the UnitTesting project in the Application. This is used to unit test applications to find bugs in the application. In an ideal world testing should be done during the implementation but for this project it was done after.

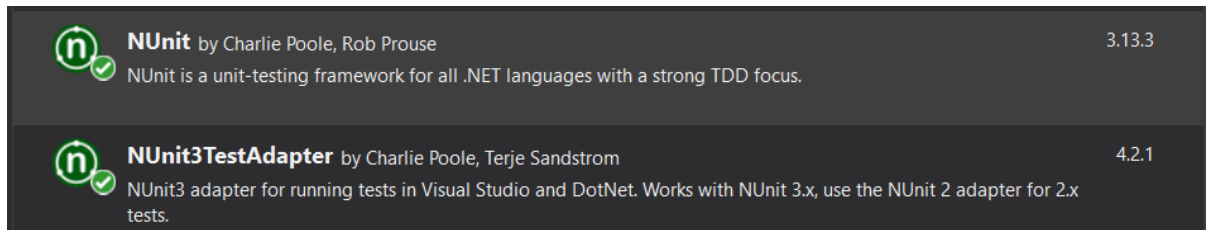


Figure 40: NUnit Installation

#### 4.3.1.1 Database Test

The first unit test made was for connecting to the Database, which passed. This indicates when connecting to the database there's no errors.

```
[Test]
✓ | 0 references
public void Database()
{
    Database db = new Database();
    Assert.IsNotNull(db.getConnection());
}
```

Figure 41: Database Connection

Checking to see if too instances of the Database are the same, which is true, shown in Figure 42.

```
[Test]
✓ | 0 references
public void TestingDatabaseComponents() {
    var db1 = Database.GetInstance();
    var db2 = Database.GetInstance();
    Assert.IsTrue(db1 == db2);
}
```

Figure 42: Testing Database Components

## 5 Conclusion

The technical area I picked for my project was NFCs. I chose this topic because I use it very often with google pay due to my own bank card not having an NFC tag in it, so I use my phone instead. I use my phone daily to pay for things and I was curious to know what security I could do to protect myself from people stealing my information. I wanted to find out what NFCs were and what they could do. I investigated and noted what I could find on these NFC tags i.e. how they worked and what they were used for.

My software program was designed with the hopes of finding out the distances different types of tags had, what type of material blocked the connection, and if two tags would interfere with the connection. The distance would be great information to have in order to determine how close or far a person needs to be to you to scan your card for data. The material is a useful bit of information as you could know what material you could keep in your pocket, wallet or phone case that will stop someone from scanning your card without you knowing. The interfering tags would help if there were two or more tags in your pocket i.e. bank cards, would a person be able to get the two cards in questions information or just one card.

I used C# to make this program and WPF for the user interface display. I got a USB reader/writer to use in this program to be able to read the tags. I built my own equipment box for testing the different distances and used it for the material blockage test. I had a sliding flat surface that would go into the slots of the box. In the window you can pick the reader/writer you'd like to use, the tag type that the user is using, grab the unique identification number from putting the tag up to the reader/writer, and pick which type of test they'd like to do i.e. distance, material, or both.

This program changed over the course of a couple of months. I figured out that I wouldn't have been able to make the interfering feature of the program because I didn't have enough time to find out a way to code it. I also couldn't make the results window show the different types of data that was recorded from doing my own testing of the tags, instead I made a button that would show the last test I made using the program, if it was distance, material, or both.

In my own experimental testing, I found out that some tag types have different distances at which they could connect to the reader/writer. The farthest distance I noted was 7cm and that was from my phone. From the material test I discovered that tin foil is effective at blocking the connection of the tag and reader/writer.

In making this program I realised that I would need to get the types and/or manufactures of the tags. The library that I used in the program cannot get this information from the tags, so I had to use an app called "NFC Tools" on android to retrieve them. I made my own table to help me

decide which tag type would be used from the manufacturer of the tag. If I had more time I would search for the library or libraries that would support C# application to get the manufacturer names from the tags.

If I were to do this project again, I would give more time to the actual programming aspect rather than thinking about programming it as I feel like I would achieve way more that way (e.g. getting the manufacturer and having a result window). I found myself thinking about it over and over again which caused me a lot of stress and slowed me down which in turn made me unable to incorporate some of these features.

The project was a successful a learning experience. I learned a lot about myself, I'd stress myself out thinking about doing it more than making the program. However, overall, I enjoyed making the project and it taught me how to connect to tags through reader/writers and has given me a greater knowledge of the security aspects of NFC.

## References

Blue Bite, 2020. *NFC Misconceptions*. [Online]

Available at: <https://www.bluebite.com/nfc/nfc-misconceptions>

[Accessed October 2021].

Christiansen, K. R., Kis, Z. & Beaufort, F., 2021. *WEB NFC*, s.l.: Web NFC Community Group.

Seritag, 2020. *NFC Tags Explained*. [Online]

Available at: <https://seritag.com/learn/using-nfc/nfc-tags-explained>

[Accessed 07 11 2021].

Thrasher, J., 2013. *RFID versus NFC: What's the difference between NFC and RFID?*.

[Online]

Available at: <https://www.atlasrfidstore.com/rfid-insider/rfid-vs-nfc/>

[Accessed October 2021].

WORLDPAY EDITORIAL TEAM, 2019. *How secure are NFC terminals?*. [Online]

Available at: <https://www.fisglobal.com/en/insights/merchant-solutions-worldpay/article/how-secure-are-nfc-terminals>

[Accessed 30 11 2021].