

# Covid Tracker

**Aldo Valentin Balsamo  
Reyes**

**410921331**

410921331@gms.ndhu.edu.tw

—

**Intro To Programing 2 - NDHU**

—

**Final Project**

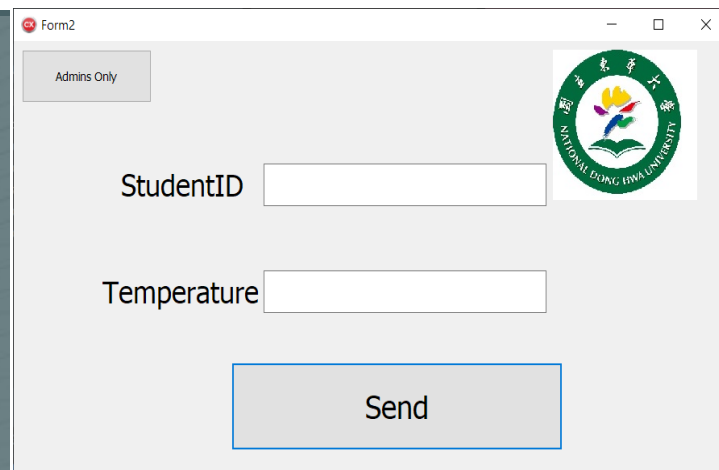
---

## Introduction

The following report is going to indulge and explain the ideas and functionality of the final project for Intro To Programing 2, a Covid Tracker.

Covid Tracker is program design to store and track the temperature of students at Dong Hwa by making an intuitive and easy to use system to store the student's temperatures.

*Please check last page to know how to start the program.*



The screenshot shows a Windows application window titled "Form2". In the top-left corner, there is a button labeled "Admins Only". In the top-right corner, there is a circular logo of National Dong Hwa University. The main area of the window contains two text input fields: "StudentID" and "Temperature". Below these fields is a large "Send" button.

## THE PROCESS

---

### IDEA

The project was done as a measure to fight against COVID-19 in Taiwan by automating a job which is being done by persons at inside of the dormitories on daily basis.

Students usually enter to the dormitory and are checked their temperature by a student/assistant and then given a sticker.

This is not only not efficient but really precarious since it can not keep track of the temperature of the students they are being recorded, which may lead to impossibility to track sudden changes in someone temperature or perhaps get to know when this person entered the dorm as a COVID positive person.

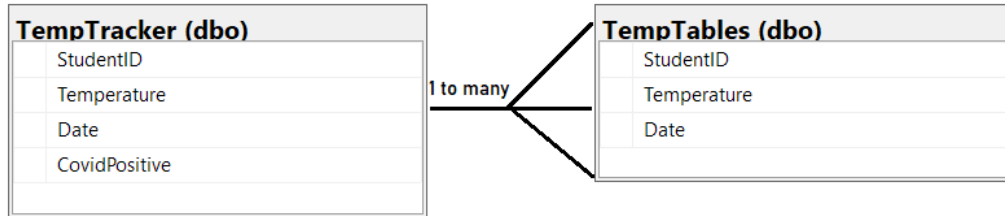
---

### Methodology

This program works by making the student input their ID and current temperature to the system, which will either record it, all of it goes into a database in Microsoft Azure for easy retrieval and modification.

It was made with the help of C++ Builder, a software the helps precompiling and generating Windows widgets and forms to work with graphical interfaces and connecting to databases swiftly and smoothly, this leads to logic code being the main focus instead.

# Database Design



We have two main tables, one with unique student's ID and another one with all the temperatures and datetimes of each student.

We also have a Admin table to store the administrators id and password so they are able to check the temperatures of the students.

The passwords available at the moment are the following:

password	userid
1234	admin
allpass	tkchang
pr109	gena

If needed more can be added since they are stored on the same Database as the other tables.

## How to operate

This is the welcome screen, user inputs his student ID and temperature. The system only accepts numbers. If a valid student ID is given a message will be shown, depending if it's a new student ID to be add to simply a student updating his temperature it will show different messages.

Form2

Admins Only

StudentID 0

Temperature 36

Cardpanel

Student ID is not valid

OK

If given an invalid student ID it will show an error message.

Then if we click “Admins Only” we are directed to the **admin login screen**, it contains a hint to what one of the passwords is, all the passwords are available on page 4 of this report. More can be added, please contact me if needed.

Form3

Back

Admin User admin

Password 1234

Log In

This works like an usual login screen, input the correct arguments and it will display the **tracking and surveillance window**, if wrong it will display an error message.

Back

Admin User 2

Password •

Cardpanel

Wrong User or Password, contact the manager please.

OK

*Note: Try logging with Admin User: **tkchang** and Password: **allpass***

Form3

Back

Admin User tkchang

Password •••••••

Log In

Also, the password is not visible by others.



# Tracking and Surveillance Window

Form4

— □ ×

The screenshot shows a software window titled "Tracking and Surveillance Window" with a "Form4" label. It contains three data tables and several interactive elements. Numbered callouts identify the following components:

- (1)** Points to the first table, which lists student data with columns: StudentID, Temperature, and Date. The first row is highlighted.
- (2)** Points to the second table, which shows detailed data for a single selected student (StudentID: 410921444).
- (3)** Points to the third table, which lists all students with columns: StudentID, Temperature, and Date.
- (4)** Points to a "Refresh" button.
- (5)** Points to a checkbox and an input box for "StudentID".
- (6)** Points to a checkbox and an input box for "Temperature >".
- (7)** Points to a "Search" button.

This is perhaps the most important but also most complex window in the program, please follow the numbers.

Number	Name	Description
1	Unique students	Shows all the students and their last temperature and last time they updated theirs. This is based on the “one” or “unique” table of the diagram shown above.
2	Track of specific student	Will show all the data of the selected student, this can be filtered with <b>(5 StudentID entry box)</b> .
3	Track of all students	Will show all the data of all student, this can be filtered with <b>(6 Temperature entry box)</b> .
4	Refresh button	Will refresh all tables to the most updated state, <b>strongly recommended</b> to do as soon as entering this window.
5	StudentID entry box	It's a entry box with a checkbox, if marked is going to be executed. It will query the <b>(2 Track of specific student)</b> to show all the students with that ID.
6	Temperature entry box	It's a entry box with a checkbox, if marked is going to be executed. It will query the <b>(3 Track all specific student)</b> to show all the students with that temperature or higher.
7	Search Button	When pressed it will query the data tables based on the checked arguments, being these <b>(2 Track of specific student)</b> and <b>(3 Track all specific student)</b>

# Code and Structure

C++Builder is a rapid application development (RAD) environment, originally developed by Borland and as of 2009 owned by Embarcadero Technologies (a subsidiary of Idera), for writing programs in the C++ programming language currently targeting Windows (both IA-32 and x64), iOS and for several releases, macOS and Android (still supported, but only Android 32-bit apps.) C++Builder combines the Visual Component Library and IDE written in Object Pascal with multiple C++ compilers.

The code was written with the help of this IDE, mainly since Microsoft stop supporting Visual C++ which led to the alternative of using this consistently updated software based on the deprecated Visual C++ and Form management that Visual Studio used to bring.

By utilizing this software, we are allowed to make beautiful looking GUIs fast and efficiently, since they do the “paperwork” of writing and compiling all the necessary code to make a basic interface. The code works via the Forms and Units system, a Form is a template ready to be worked to our needs, you can add Buttons, Text boxes and even real times queries with it, without the necessity of writing thousands of lines of jargon and deprecated code, it automatically does that for use, this allows us to focus on the actual code rather than library after library.

Each form possesses a similar structure to a Class

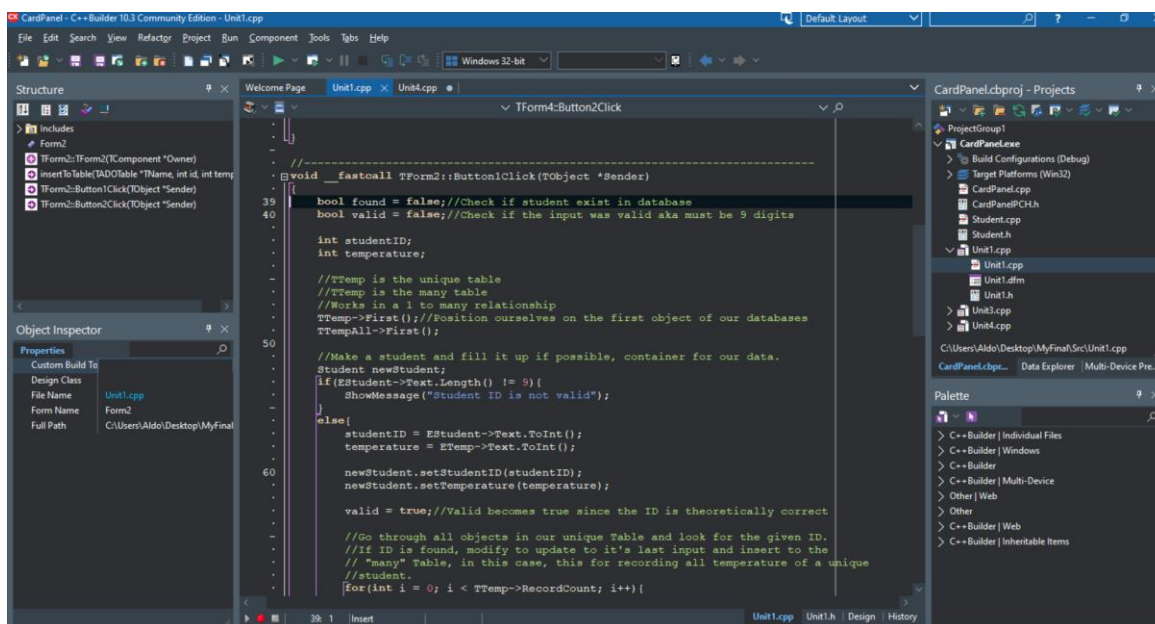
- Unit.h
- Unit.cpp
- Unit.dfm

Unit.h will know what widgets (Buttons, Text boxes) are in the current form.

Unit.cpp will let us actually code this Buttons or interactions with the form itself.

Unit.dfm just stores the positions and needed code for this button without us worrying for it,

Example of how the IDE looks like (actual code for this project is shown).



Because of the three windows of this program, we possess three sets of Units, and also a Student Class with a similar architecture just to store temporally the inputs.

Here's a example how code normally looks like.

```

·  __fastcall TForm3::TForm3(TComponent* Owner)
-   : TForm(Owner)
·  {
·      EAdmin->TextHint = "admin";
·      EPassword->TextHint = "1234";
·  }
20  //-----
·  void __fastcall TForm3::Button1Click(TObject *Sender)
·  {
·      Form3->Close();
·  }
·  //-----
·  void __fastcall TForm3::Button2Click(TObject *Sender)
·  {
29      |
30      bool found = false;
·      TAdmin->First();
·      for(int i = 0; i < TAdmin->RecordCount; i++){
·          if(TAdmin->FieldByName("userid")->AsString == EAdmin->Text && TAdmin->FieldByName("password")->AsString == EPassword->Text){
·              found = true;
·              break;
·          }
·          TAdmin->Next();
·      }
40
·      if(found){
·          Form4->ShowModal();
·      }
·      else{
·          ShowMessage("Wrong User or Password, contact the manager please.");
·      }
·  }

```

We define `__fastcall` for the different Objects of our Forms, these are event based calls that execute when an interaction with one Object occurs, for example look at **TForm3::Button2Click(TObject \*Sender)**, it will be executed when the "Button2" in "TForm3" is pressed, if so we are going to do a query to our database and try to locate the inputs in the EditBox at the form.

All the code that can be seen on the source file gives, please write me if explanation is needed.

```

·  #pragma once
·  #include <string>
·
·  class Student{
-  public:
·      Student();
·      Student(int studentID, int temp);
·      int getStudentID();
10
·      int getTemperature();
·
·      int setTemperature(int n);
·      int setStudentID(int n);
·
·  private:
·      int studentID;
·      int temperature;
20
·  };
22

```

We also have a Student Class, we show here its header since it's quite simple class but just what we needed on this project to store inputs given by the user temporarily on a variable while working on it aka check, for example if: **Student.getStudentID()** in Table1?



# How to read the source file?

Without C++ Builder is hard to compile since this IDE makes a Makefile and compiles all the necessary resources on runtime, but if at least want to see how the code looks like please take a look the carpet at **MyFinal>Src>Win32>Debug** and here you will see the Unit files for the program

# How to run the program?

**MyFinal>Src>Win32>Debug>CardPanel.exe**

# Thanks.

References:

<https://www.embarcadero.com/products/cbuilder>

<https://azure.microsoft.com/en-us/>