

## Background and Goals

- Arduino is an AVR microcontroller that provides an interface for interacting with sensors.
- The Arduino board is programmed using a dialect of C++ language.
- Arduino provides a low-cost way for novices and professionals to create devices and experiments.
- Since coding could be daunting for novices, a GUI interface is needed for quick experiment creation and extraction of data.

## Back end

1. Create a backend API to support basic menu options such as:
  - compilation and upload of Arduino code.
  - creation of new experiments and sensors.
2. management of results using pandas.
3. Create a system to analyze code using a custom tag system and support:
  - Merging code of 2 or more sensors into an experiment.
  - Changing experiment attributes such as the sampling frequency and pin numbers.

## Front end

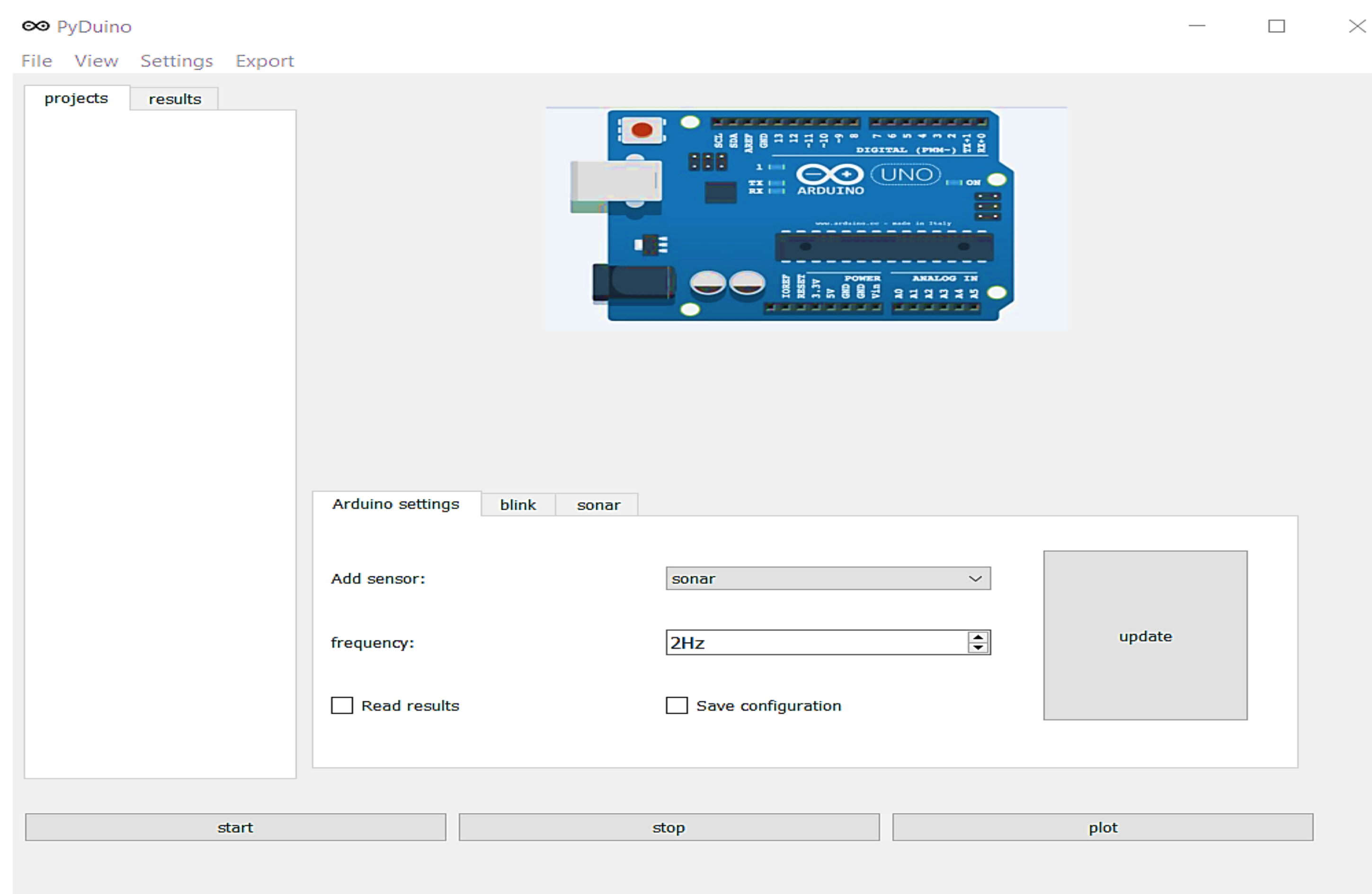
1. Create a graphic interface using PyQt5 to support:
  - ❖ Loading and display of current and past projects.
  - ❖ Changing project and sensor attributes
2. Add a Text editor to enable quick access to sensor/project code for non-trivial changes.
3. Add an interface for plotting results and manipulating data using matplotlib and PyQt5
4. A dedicated option for adding new sensor code/project from the PC

## Arduino IDE

```
// ** libraries
#include <Arduino.h>
// ** pins : digital
const int led = 13;
const int trigPin = 7;
const int echoPin = 8;
// ** pins : analog
// ** frequency
const int sleep_time = 1000;
// ** global
int duration;
float distance;
// ** setup
void setup() {
  Serial.begin(9600);
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(led, OUTPUT);
  Serial.begin(9600); // Starts the serial communication
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  // sleep for 1 sec
  delay(1000);
}

// ** loop start
void loop() {
  digitalWrite(led, HIGH);
  delay(sleep_time);
  digitalWrite(led, LOW);
  delay(sleep_time);
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance = duration * 0.034 / 2;
}
```

## Experiment builder interface



## Result plotter

