NUST
NATIONAL UNIVERSITY
OF SCIENCES & TECHNOLOGY

INSTRUMENTATION AND MEASUREMENTS

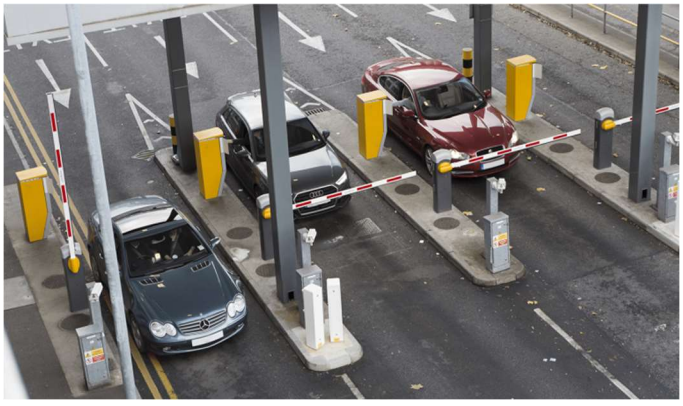
FINAL PROJECT REPORT

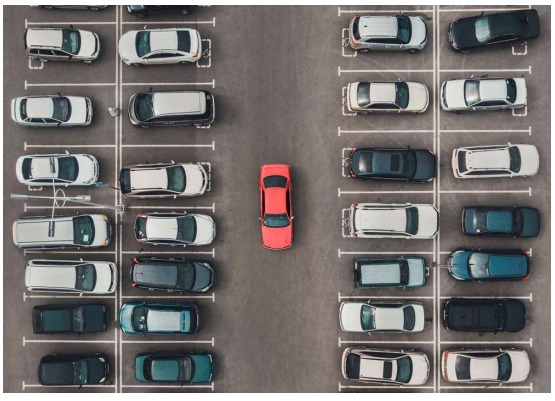
SUBMITTED TO: DR. MUSTAFA TAHSEEN

DATED: 30TH DECEMBER 2023

CLASS: BEE-13A

NAME	CMS
Muhammad Zohaib Irfan	372692
Rayan Malik	366756
Muhammad Samiullah	369138





SMART PARKING SYSTEM

ABSTRACT

This project addresses persistent challenges faced by our institution, notably the encroachment of students into faculty parking areas and unauthorized access to reserved spaces. Our Smart Parking System employs cutting-edge technologies, such as computer vision, machine learning, and object detection sensors, to offer a comprehensive solution. By seamlessly integrating these technologies, the system not only rectifies current parking issues but also contributes to the overarching objective of fostering a technologically advanced and user-friendly campus environment. This innovative approach not only ensures efficient parking management but also reflects our commitment to leveraging technology for enhancing the overall campus experience.

INTRODUCTION:

Parking management is an integral aspect of any institution, and our proposal for a Smart Parking System seeks to revolutionize this essential service. With the escalating number of vehicles on the road, efficient parking solutions have become imperative. Studies reveal that an average driver spends approximately **17 hours per year** (McCoy, 2017) searching for parking spaces, highlighting the pressing need for a streamlined approach to parking management.

At our institution, we face specific challenges, such as students encroaching into faculty parking areas and unauthorized access to reserved spaces. Our Smart Parking System leverages state of the art technologies, including computer vision, machine learning, and object detection sensors, to provide a comprehensive solution. This project not only addresses our current parking woes but also aligns with the broader goal of creating a technologically advanced and user-friendly environment within our campus.

LITERATURE REVIEW

The literature review explores the challenges faced by educational institutions in managing parking facilities and the growing importance of Smart Parking Systems as innovative solutions. Traditional strategies often fall short in addressing issues such as unauthorized use of reserved spaces and increasing congestion. The review delves into the technological interventions that have proven effective, emphasizing the role of computer vision, machine learning, and object detection sensors in real-time monitoring and enforcement. The integration of user-friendly interfaces, including mobile applications, is highlighted as crucial for user acceptance and overall system effectiveness. Case studies demonstrate successful implementations in various academic settings, showcasing reduced violations, improved space utilization, and enhanced campus mobility. Importantly, the review emphasizes that these systems not only address immediate parking challenges but also align with broader campus goals of fostering a technologically advanced and user-friendly environment, reflecting the

institution's commitment to innovation. Future research is encouraged to explore scalability, cost-effectiveness, and sustainability in the evolving landscape of smart parking technologies within educational contexts.

AIMS AND OBJECTIVES:

Eliminate Unauthorized Parking: Distinguish between student and faculty vehicles at entry points, ensuring strict adherence to designated parking areas to eliminate unauthorized parking instances.

Reserved Faculty Parking Management: Implement a robust system for identifying reserved faculty members based on number plate recognition.

Real-time Parking Slot Availability: Monitoring occupancy status in parking lot and display real-time information on a centralized screen at the entrance, guiding incoming vehicles to available parking slots.

Optimization of Parking Resources: Utilizing data analytics to assess parking usage patterns and optimize space allocation, thus Improving overall efficiency in parking resource utilization for long-term sustainability.

PARKING AND LAYOUT PLAN FOR UNDERSTANDING:

In the above picture we can see that the incoming car first reaches the detection stage, it's number plate will be scanned here and compared by the database, if it matches with any of the faculty then its allowed to enter the right-hand side, else it will have to park in the student's parking in the left-hand side. Also, if any faculty has a reserved parking, his/her number plate will be scanned at his parking, if the number plate confirms it to be him/her, he/she will be allowed to get in.

For example, in the above picture we can see a possible place where we can detect the car if it is of any student's or teachers'.

IMPLEMENTATION AND REQUIRED COMPONENTS:

Detection Mechanism:

Faculty vs. Student Differentiation:

- Utilization of cameras (Raspberry Pi Cam or ESP32 Cam) at entry points.
- Live stream sent to an edge device for real-time processing.
- Implementation of computer vision and machine learning algorithms to distinguish between faculty and student vehicles based on number plate recognition.

Reserved Faculty Parking Identification:

- Similar camera-based setup for parking spaces.
- Number plate recognition and database matching.
- Identification of reserved faculty, allowing access to specific parking slots

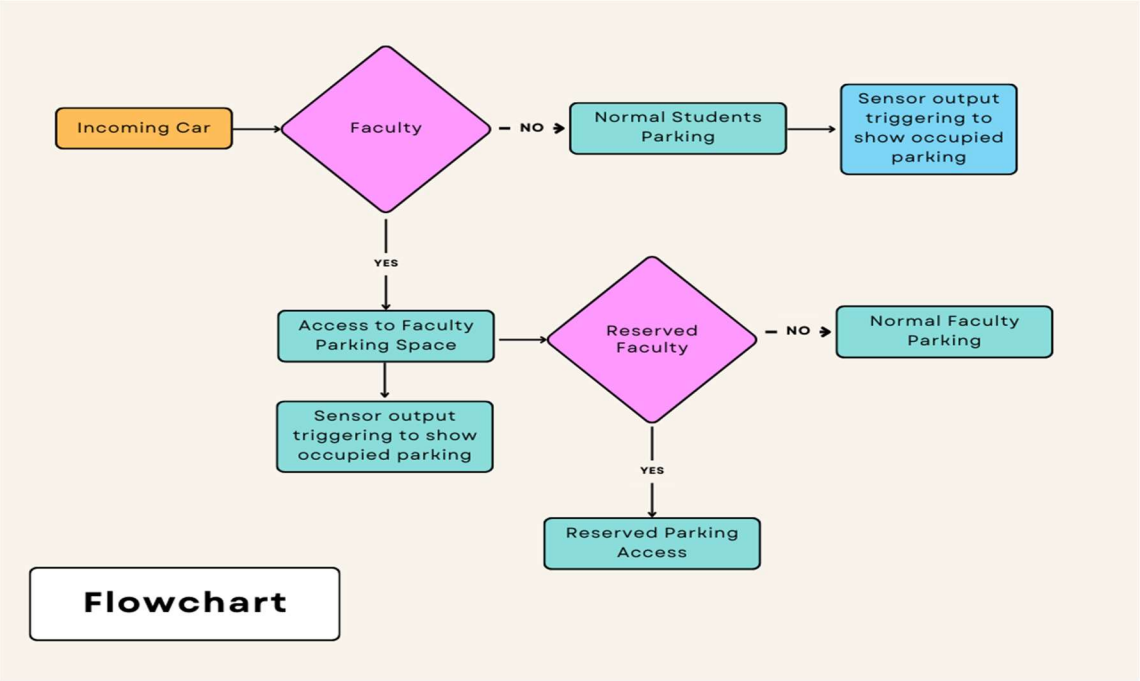
Parking Space Monitoring:

- Sensors like Ultrasonics, deployed in parking spaces to detect the presence of a vehicle.
- Real-time status updates sent to the central system.

User Interface:

- A large display screen at the parking lot entrance will showcase available parking slots, for students and faculty. Occupied spaces will be updated based on sensor inputs.

FLOW CHART:



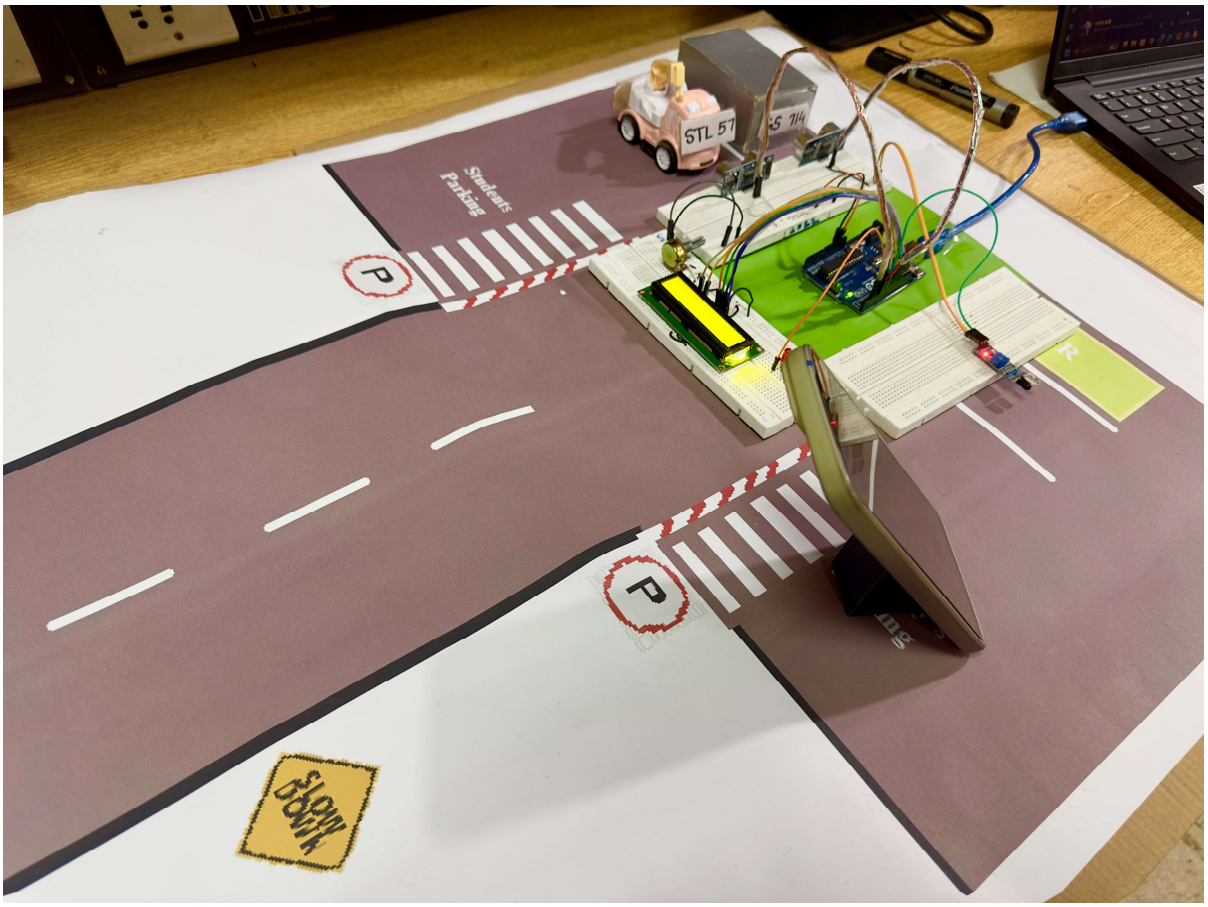
TOOLS USED:

- Arduino
- Google Cloud
- Google Colab
- OpenCV
- JavaScript

METHODOLOGY:

The methodology for our smart parking system involves integrating license plate detection and ultrasonic sensor-based occupancy monitoring. After collecting and preprocessing datasets for compatibility, a pretrained deep learning model is fine-tuned to seamlessly integrate ultrasonic sensor data. The model's detection results are transmitted to the cloud, where an Arduino is coupled to receive instructions, facilitating real-time categorization of parking spaces. This integration links detected license plates to user profiles, distinguishing between faculty and student parking. The resulting prototype optimizes parking space utilization, providing a seamless and efficient parking experience through cloud-aided communication between the ML model and the Arduino-controlled system.

HARDWARE IMPLEMENTATION:



ACHIEVED OUTCOMES:

- Streamlining traffic and reducing congestion in the campus parking facility.
- Strengthening security by restricting access to reserved faculty parking areas.
- Optimizing parking spaces through data-driven insights for efficient allocation
- Providing real-time parking availability information to save time and fuel for users
- Enhancing user satisfaction by alleviating parking-related frustrations

SUSTAINABLE DEVELOPMENT GOALS (SDGS):

SDG 9: Industry, Innovation, and Infrastructure:

The project leverages advanced technologies, including computer vision and machine learning, to enhance parking infrastructure and improve overall efficiency.

SDG 11: Sustainable Cities and Communities:

By optimizing parking space utilization and reducing traffic congestion, the project contributes to creating more sustainable and livable urban environments.

SDG 17: Partnerships for the Goals:

The Smart Parking System fosters collaboration between technology, infrastructure, and user communities, showcasing the significance of partnerships in achieving sustainable development objectives.

SDG 13: Climate Action:

The reduction in unnecessary vehicle movements and idling, facilitated by efficient parking management, supports initiatives to minimize carbon emissions and combat climate change.

APPLICATIONS:

While the illustrations and figures presented herein primarily focus on the Smart Parking System's application in the SEECS parking lot, it is crucial to recognize the versatility of this technology, making it applicable in various other settings. The system's adaptability extends to:

- Commercial Complexes
- Hospitals
- Corporate Campuses
- Public Events and Venues
- Residential Communities
- Educational Institutions:

WORK DISTRIBUTION:

The implementation of the Smart Parking System is a collaborative effort, and while the following tasks are broadly distributed among the three team members, it is essential to emphasize that collaboration and mutual assistance was made sure during whole project.

Hardware Integration and Installation:

- Procurement and setup of cameras, sensors, and edge devices at entry points and parking spaces.
- Deployment of hardware components at designated locations.

Software Development and User Interface:

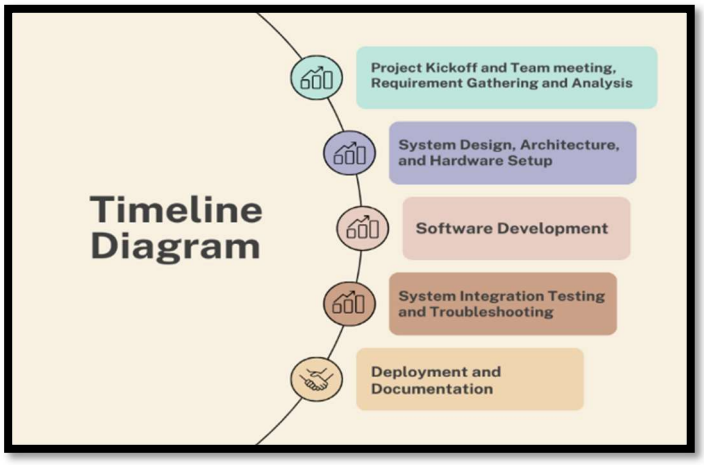
- Design and implementation of computer vision and machine learning algorithms for number plate recognition.
- Development of the central system for parking slot monitoring.
- Designing a user-friendly interface for the central screen displaying parking availability.

Testing, Troubleshooting, and Documentation:

- System-wide testing to ensure seamless operation, identifying and addressing any issues.
- Collaboration with hardware and software teams for integrated testing.
- Documentation of testing procedures, results, and troubleshooting guidelines.
- Regular updates and comprehensive documentation of progress, challenges, and solutions.

This expanded distribution ensures that each team member has a well-defined set of tasks, promoting a holistic and collaborative approach to the Smart Parking System project.

TIMELINE:



CHALLENGES FACED:

- Limited GPU resources hindered real-time processing, necessitating optimizations.
- Integrating cloud resources with ML model and Arduino for seamless transfer of data.

COST BREAKDOWN:

Component	Cost (Rs)
Arduino	2000
Ultrasonic Sensor (3x)	1200
LCD Screen	300
Printing & Poster	600
Breadboards	500
Total	**4600**

CONCLUSION:

- The Smart Parking System offers an innovative solution to address unauthorized parking and confusion in parking lots.
- Utilizing ultrasonic sensors, cameras, and machine learning, the system efficiently identifies faculty and students while pinpointing reserved parking spaces.
- The inclusion of a real-time display at the entrance streamlines the parking process, reducing congestion and improving the overall user experience.
- Positioned at the intersection of technology and practicality, the project promises a transformative impact on parking management in educational institutions.
- With its focus on order, efficiency, and user satisfaction, the Smart Parking System stands out as a solution that enhances the overall parking infrastructure in educational settings.

REFERENCES:

McCoy, K. (2017) *Drivers spend an average of 17 hours a year searching for parking spots*, *CNBC*. Available at: https://www.cnbc.com/2017/07/12/drivers-spend-an-average-of-17-hours-a-year-searching-for-parking-spots.html (Accessed: 30 November 2023).

(No date) *Free and customizable timeline templates - CANVA*. Available at: https://www.canva.com/templates/s/timeline/ (Accessed: 30 November 2023).

(No date a) *Google maps*. Available at: https://www.google.com/maps (Accessed: 30 November 2023).

ARDUINO CODE:

```
#include <LiquidCrystal.h>

// Define LCD pins
LiquidCrystal lcd(8, 9, 10, 11, 12, 13);

// Define ultrasonic sensor pins
const int trigPin = 3;
const int echoPin = 2;

// Updated pins for the second ultrasonic sensor
const int trigPin2 = 5;
const int echoPin2 = 6;

// IR sensor pin
const int irSensorPin = 7; // Change to pin 7

const int ledPin = 4; // Change LED pin

int x = 0;
int y = 0;
int z = 0;

void setup() {
  // Set up LCD
  lcd.begin(16, 2);

  // Set up ultrasonic sensor pins
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  // Set up pins for the second ultrasonic sensor
  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2, INPUT);

  // Set up IR sensor pin
  pinMode(irSensorPin, INPUT);

  // Set up LED pin
  pinMode(ledPin, OUTPUT);

  // Start serial communication
  Serial.begin(9600);
  Serial.setTimeout(1);
}
```

```
void loop() {
  // Check for user input from serial monitor
  if (Serial.available() > 0) {
    int command = Serial.parseInt();
    if (command == 1) {
      // Turn on the LED
      digitalWrite(ledPin, HIGH);
    } else if (command == 0) {
      // Turn off the LED
      digitalWrite(ledPin, LOW);
    }
  }

  // Trigger and measure distance for the first ultrasonic sensor
  x = triggerAndMeasureDistance(trigPin, echoPin);

  // Trigger and measure distance for the second ultrasonic sensor
  y = triggerAndMeasureDistance(trigPin2, echoPin2);

  // Check IR sensor
  z = !digitalRead(irSensorPin);

  // Print the LCD message based on the values of x, y, and z
  printLCDMessage(x, y, z);
```

```
int triggerAndMeasureDistance(int trigPin, int echoPin) {
  // Trigger ultrasonic sensor
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the duration of the pulse from the echo pin
  long duration = pulseIn(echoPin, HIGH);

  // Calculate distance in centimeters
  int distance = duration / 58;

  // No need to print here; just return 1 if triggered, 0 otherwise
  return (distance < 10) ? 1 : 0;
}

void printLCDMessage(int x, int y, int z) {
  lcd.clear();
  lcd.setCursor(0, 0);
```

```
  // Print the appropriate message based on the values of x, y, and z
  if (x == 0 && y == 0 && z == 0) {
    lcd.print("S1|S2|F|RF");
  } else if (x == 0 && y == 0 && z == 1) {
    lcd.print("S1|S2| |RF");
  } else if (x == 0 && y == 1 && z == 0) {
    lcd.print("S1|  |F|RF");
  } else if (x == 0 && y == 1 && z == 1) {
    lcd.print("S1|  | |RF");
  } else if (x == 1 && y == 0 && z == 0) {
    lcd.print("  |S2|F|RF");
  } else if (x == 1 && y == 0 && z == 1) {
    lcd.print("  |S2| |RF");
  } else if (x == 1 && y == 1 && z == 0) {
    lcd.print("  |  |F|RF");
  } else if (x == 1 && y == 1 && z == 1) {
    lcd.print("  |  | |RF");
  }
```

ML MODEL:

```python
# import dependencies
from IPython.display import display, Javascript, Image
from google.colab.output import eval_js
from base64 import b64decode, b64encode
import cv2
import numpy as np
import PIL
import io
import easyocr
```

```python
# function to convert the JavaScript object into an OpenCV image
def js_to_image(js_reply):
    # decode base64 image
    image_bytes = b64decode(js_reply.split(',')[1])
    # convert bytes to numpy array
    jpg_as_np = np.frombuffer(image_bytes, dtype=np.uint8)
    # decode numpy array into OpenCV BGR image
    img = cv2.imdecode(jpg_as_np, flags=1)

    return img

# function to convert EasyOCR result into bounding box image
def ocr_to_bbox_img(img, results):
    bbox_array = np.copy(img)
    for (bbox, text, prob) in results:
        (top_left, top_right, bottom_right, bottom_left) = bbox
        top_left = tuple(map(int, top_left))
        bottom_right = tuple(map(int, bottom_right))
        bbox_array = cv2.rectangle(bbox_array, top_left, bottom_right, (255, 0, 0), 2)
    return bbox_array
```

```python
from fuzzywuzzy import fuzz

# Threshold for fuzzy matching (adjust as needed)
fuzzy_threshold = 70

# Initialize an empty list to store detected text
detected_text_list = []

# Define the database of strings to match
database_strings = ["STL 57", "57 STL", "LEB 40", "40 LEB", "4o LEB", "LEB 4o", "ABC 123"]

# start streaming video from webcam
video_stream()
# label for video
label_html = 'Capturing...'
# initialize bounding box to empty
bbox = ''
count = 0
```

```python
        # Extract and store detected text
        detected_text = [result[1] for result in results]
        detected_text_list.extend(detected_text)

        # Check if any text is detected
        if detected_text:
        # Print the detected text for each frame
            print("Detected Text:", detected_text)

        # Check if the detected text matches or is similar to any string in the database
            x = 0  # match_status
            for text in detected_text:
            # Use fuzzy matching to handle slight variations
                if any(fuzz.ratio(text, db_string) > fuzzy_threshold for db_string in database_strings):
                # Set match_status to 'high' when there is a match or similarity
                    x = 1
                    break

        # Print the match status
            print(x)
            x = str(x)
            x = x.encode('utf-8')
            future = publisher.publish(topic_path, x)
```

PYTHON-ARDUINO BINDING:

```
import os
from google.cloud import pubsub_v1
from concurrent.futures import TimeoutError
import serial.tools.list_ports
serialInst = serial.Serial()


serialInst.baudrate = 9600
serialInst.port = "COM3"
serialInst.open()

x = False
credentials_path = 'arduino.privatekey.json'
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = credentials_path

subscriber = pubsub_v1.SubscriberClient()
subscription_path = 'projects/arduino-409416/subscriptions/arduinouno-sub'


def callback(message):
    x = message.data.decode('utf-8')
    serialInst.write(x.encode('utf-8'))
    print(f'{x}')
    message.ack()

streaming_pull_future = subscriber.subscribe(subscription_path, callback=callbac

print(f'Listening for messages on {subscription_path}')

with subscriber:
    try:
        streaming_pull_future.result()
    except TimeoutError:
        streaming_pull_future.cancel()
        streaming_pull_future.result()
```

PYTHON-CLOUD BINDING:

```
[5] import os
    from google.cloud import pubsub_v1
    credentials_path = '/content/arduino.privatekey.json'
    os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = credentials_path
    publisher = pubsub_v1.PublisherClient()
    topic_path = 'projects/arduino-409416/topics/arduinouno'
```