

# COMMUNICATIONS

---

Wang Minxi John (2101925)

Ng Zi Hao (2102893)

Low Li Pin (2101542)

# Table of Contents

---

**01**

Overview

---

**03**

UART (ESP8266)

---

**02**

UART (M5StickC Plus)

---

**04**

Bluetooth (HC05)

---

The background features abstract geometric patterns in the corners, consisting of thin blue lines, dots, and circles. In the top-left, there are several parallel lines and a small cluster of dots. In the top-right, a circle with a dot inside is connected to a line. In the bottom-left, there are more parallel lines and a small cluster of dots. In the bottom-right, there are several parallel lines and a small cluster of dots.

# 01

---

## Overview

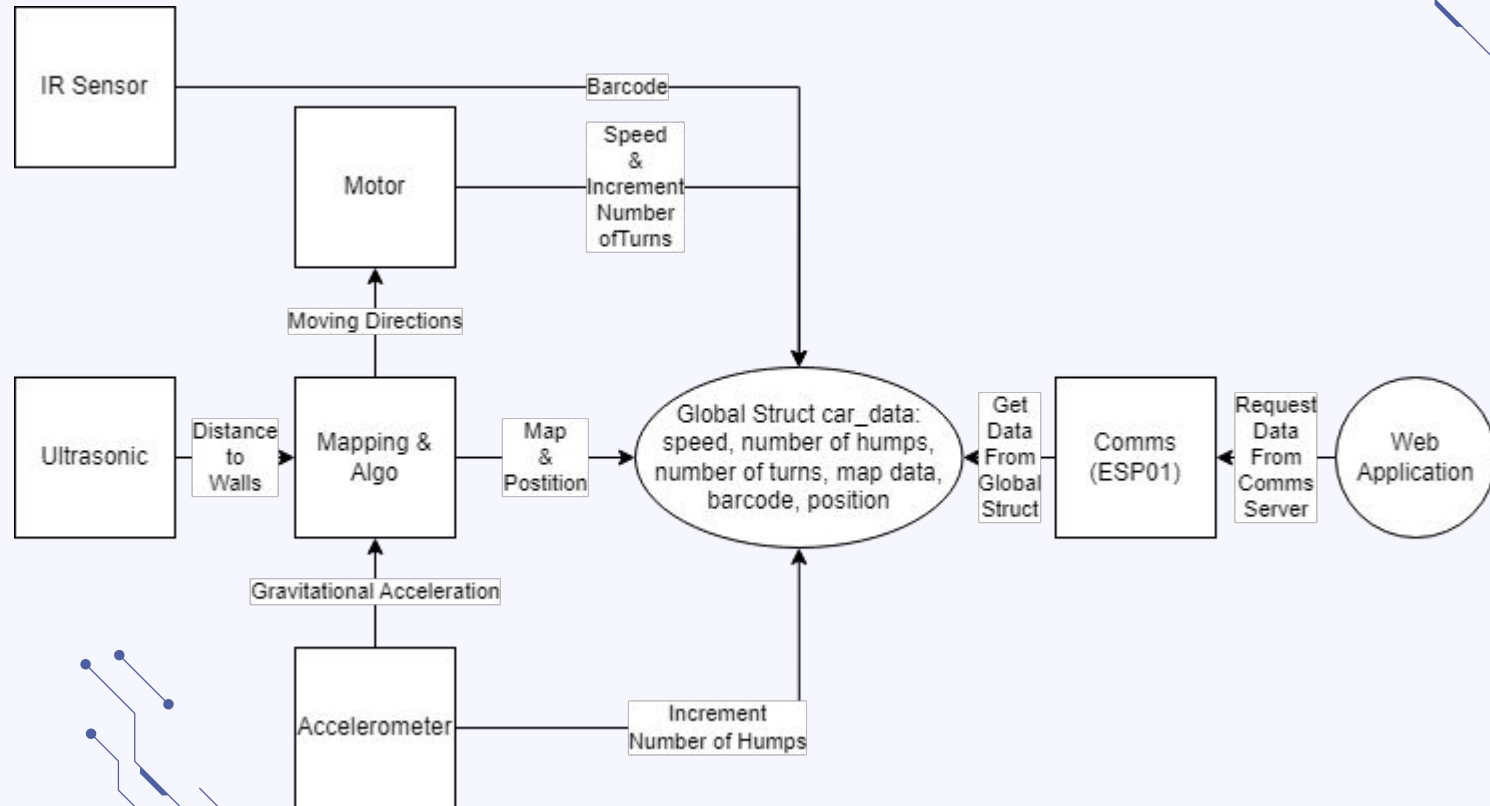
# Overview

- Pico Board
- Communication protocol used:
  - PICO -> ESP01 (UART)
  - PICO -> HC05 (UART)
  - PICO -> M5STICKC PLUS (UART)
  - MQTT (M5Stick)

# Overview

- Flask Web Application displaying data collected from the car
  - Functions:
    - Send starting node
    - Send direction of the car (left, front, right, back)
    - Display event (hump detected, turning, etc)
    - Display distance travelled
    - Display car speed
    - Display barcode information
    - Display number of turns
    - Display map nodes

# Program Flowchart

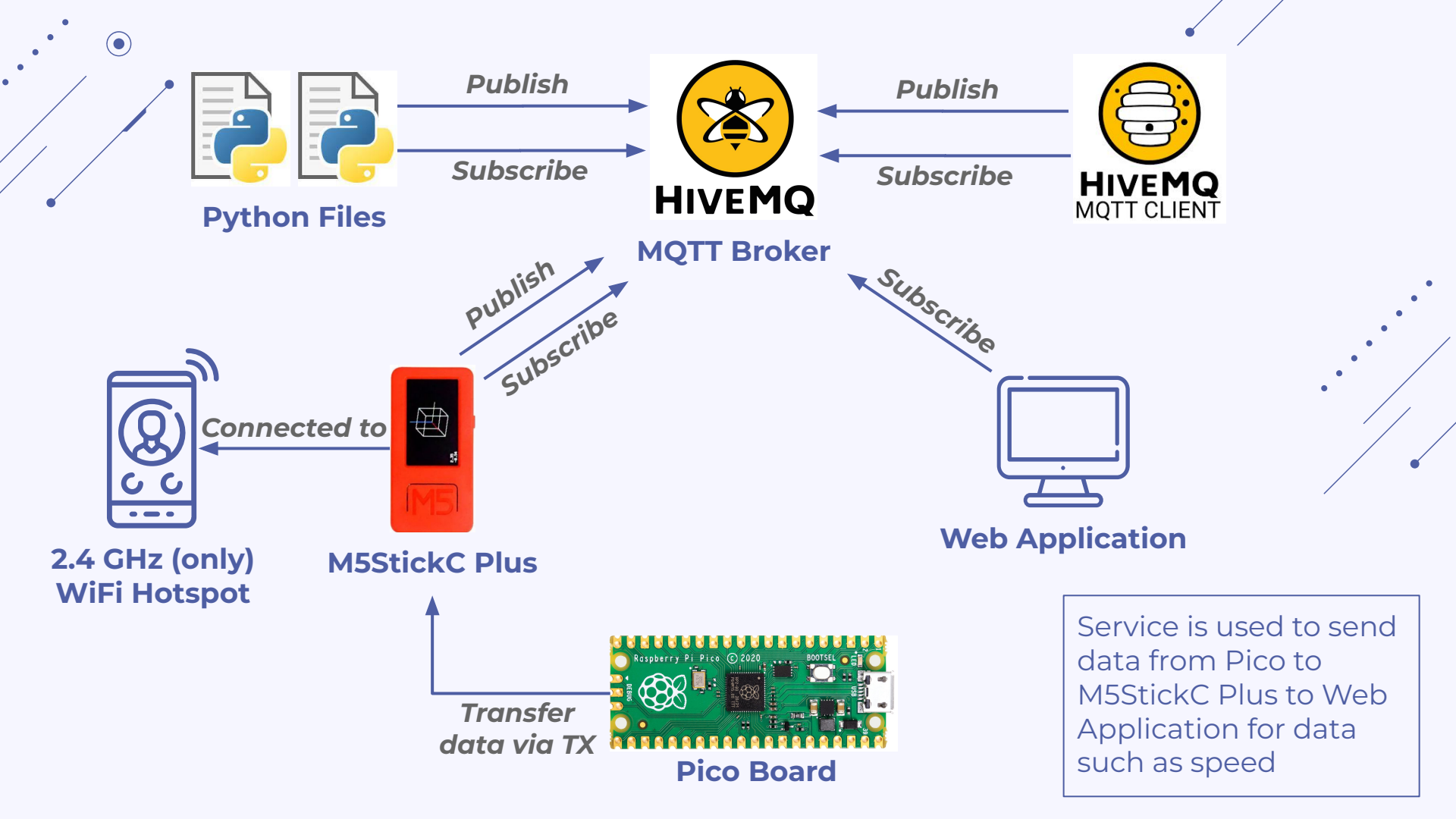


The background features decorative geometric patterns in the corners, consisting of thin blue lines, dots, and circles. In the top-left, there are several parallel lines and a small cluster of dots. The top-right has a circle with a dot inside and a line with a dot. The bottom-left shows a circle with a dot and some lines. The bottom-right contains a series of parallel lines, a circle with a dot, and a small cluster of dots.

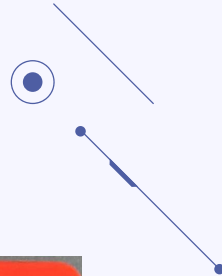
# 02

---

## UART (M5StickC Plus)





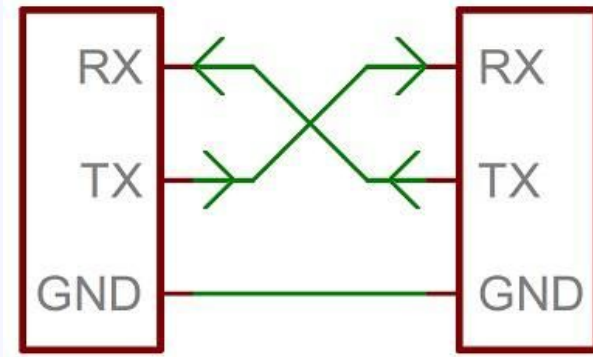


# Setup

- M5StickC Plus
  - TX -> G32
  - RX -> G33
  - Databits -> 8
  - StopBits -> 1
  - ParityBits -> None
- Pico Board
  - TX -> G1
  - RX -> G0
  - Databits -> 8
  - StopBits -> 1
  - ParityBits -> None
- UART0

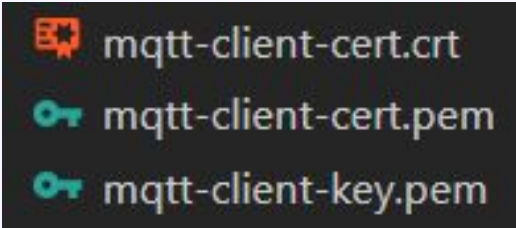
M5StickC Plus

Pico Board



# Setup

- MQTT
  - HiveMQ Broker
  - Broker URL
    - 3eb3b23922da49208766dd4507ecd30c.s1.eu.hivemq.cloud
  - Created 10 user accounts
    - user-1 -> user-10
    - SSL enabled (for secure MQTT)
    - SSL protocol (TLSv1.2)
- Web Application
  - CA\_CERTS
  - CERTFILE
  - KEYFILE



```
mqtt-client-cert.crt  
mqtt-client-cert.pem  
mqtt-client-key.pem
```

A terminal window with a dark background showing three lines of text. Each line has a small icon to its left: an orange document icon for the first line, a green key icon for the second, and a teal key icon for the third.

# Functions

	Main Functions
M5StickC Plus	<code>initiate_UART()</code>
	<code>if uart1.any():</code>
Pico Board	<code>uart_puts();</code>
Web Application	<code>@mqtt_client.on_message()</code>

# Video Demo

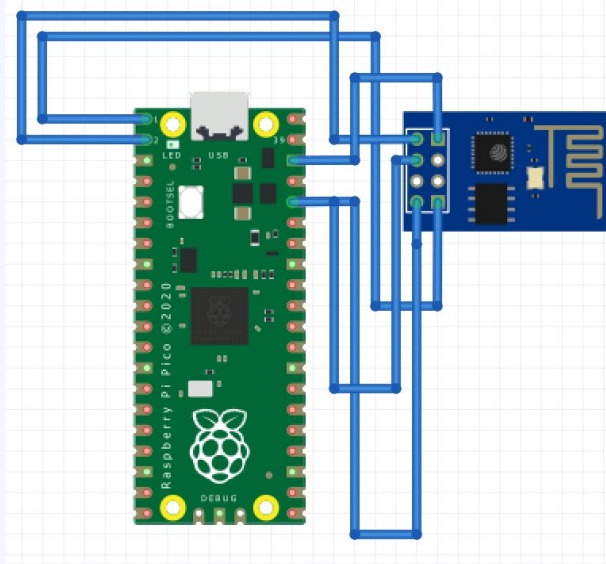


The background features decorative geometric patterns in the corners, consisting of thin blue lines, dots, and circles. In the top-left, there are several parallel lines and a cluster of dots. In the top-right, a circle with a dot inside is connected to a line. In the bottom-left, there are more parallel lines and a small circle with a dot. In the bottom-right, there are more complex line patterns and a circle with a dot.

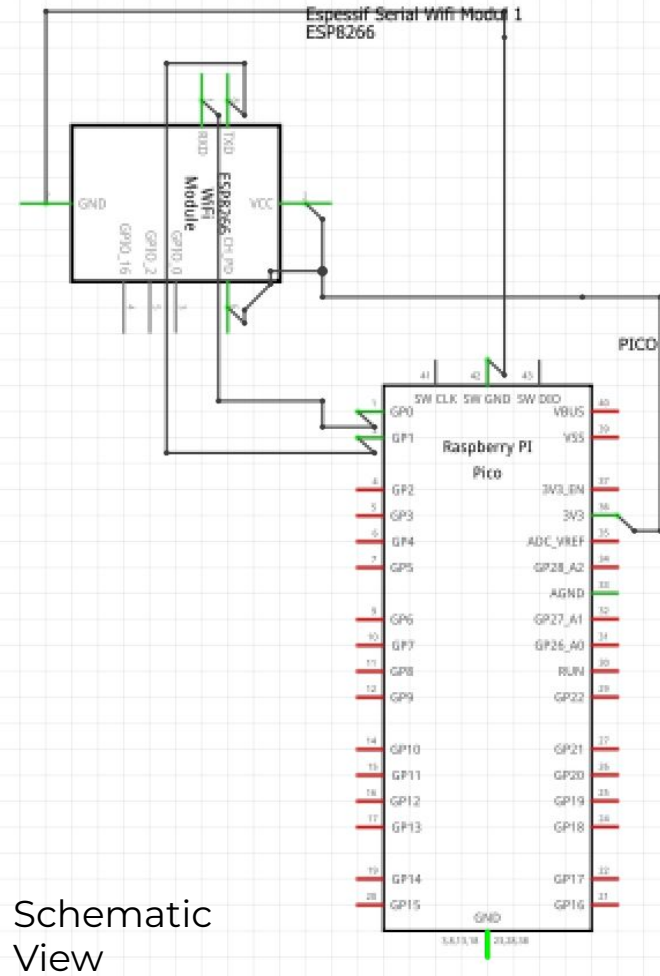
# 03

---

## UART (ESP8266)



## Breadboard View



## Schematic View

# Config

## Pico Config-

UART0

TX\_PIN 0, RX\_PIN 1

Baud Rate 115200

Databits 8 | Stopbits 1 | Parity None

## Pin Connection-

Pico GP0 (TX) -> ESP01 (RX)

Pico GP1 (RX) -> ESP01 (TX)

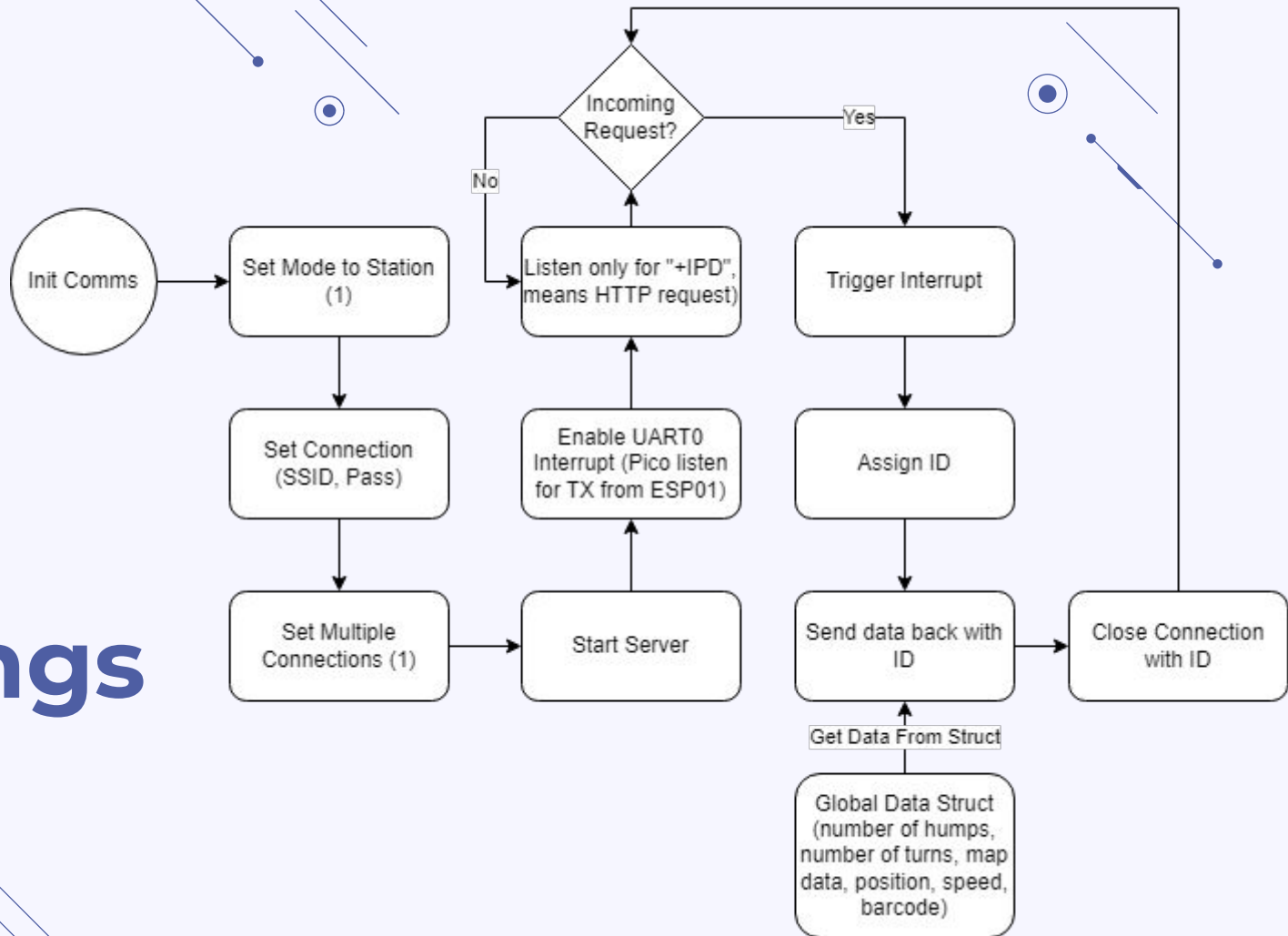
Pico 3V3-> ESP01 (VCC)

Pico GND -> ESP01 (GND)

Pico 3V3 -> ESP01 (EN)



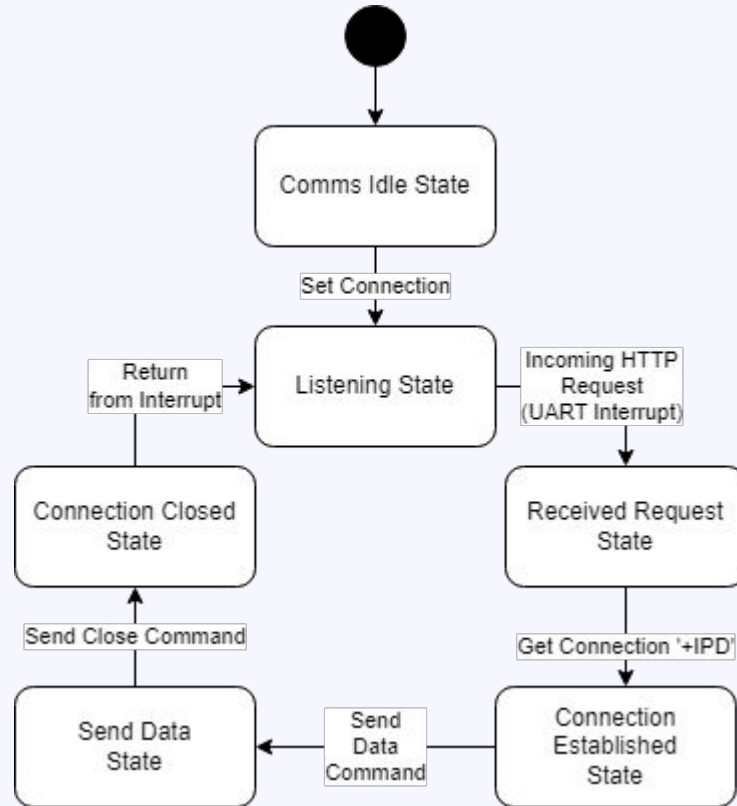
# Inner Workings



# Commands

- 1) Set up UART0 on Pico
- 2) Set mode with `"AT+CWMODE=1"`
- 3) Set connection with `"AT+CWJAP="ssid","password"`
- 4) Get IP with `"AT+CIFSR"`
- 5) Get ID by listening to first occurrence of "+IPD" by using strstr to check UART0 RX interrupt string
- 6) Set multiple connections with `"AT+CIPMUX=1"`
- 7) Start server with `"AT+CIPSERVER=1,80"`
- 8) Send data with `"AT+CIPSEND=<ID>"`, wait for ">", then send data
- 9) Close IPD connection with `"AT+CIPCLOSE=<ID>"`

# Finite State Machine



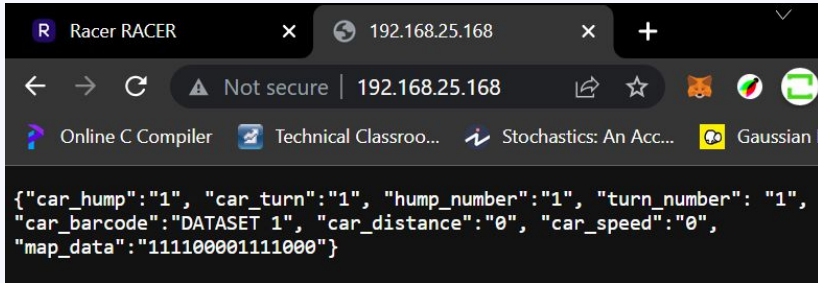
# Testing

Conducted black box testing

1) Testing Criteria

- a) Debug through UART0 should be displayed through Putty Serial
- b) “Wifi Connected”, “IP” should display through Putty Serial
- c) Connection details should display through Putty Serial when requesting resources through browser
- d) JSON resources should be available through browser

# Testing Results



A screenshot of a web browser window. The address bar shows the URL 192.168.25.168. The page content displays a JSON object with the following fields: "car\_hump": "1", "car\_turn": "1", "hump\_number": "1", "turn\_number": "1", "car\_barcode": "DATASET 1", "car\_distance": "0", "car\_speed": "0", and "map\_data": "111100001111000".

```
{"car_hump": "1", "car_turn": "1", "hump_number": "1", "turn_number": "1",  
"car_barcode": "DATASET 1", "car_distance": "0", "car_speed": "0",  
"map_data": "111100001111000"}
```

JSON results from requesting  
resource at IP address of PICO  
(ESP01)



A screenshot of a PuTTY terminal window titled "COM4 - PuTTY". It shows the output of several AT commands and an HTTP request. The AT commands and their responses are: AT+CWMODE=1 (OK), AT+CWJAP="pico\_test1","testtest" (WIFI DISCONNECT, WIFI CONNECTED, WIFI GOT IP), AT+CIFSR (OK, AT+CIFSR:STAIP,"192.168.25.168", AT+CIFSR:STAMAC,"ac:0b:fb:c8:4a:e4", OK), AT+CIPMUX=1 (OK), and AT+CIPSERVER=1,80 (OK). The terminal also shows an HTTP GET request from 127.0.0.1:8080 to 192.168.25.168.

```
COM4 - PuTTY  
AT+CWMODE=1  
  
OK  
AT+CWJAP="pico_test1","testtest"  
WIFI DISCONNECT  
WIFI CONNECTED  
WIFI GOT IP  
  
OK  
AT+CIFSR  
+CIFSR:STAIP,"192.168.25.168"  
+CIFSR:STAMAC,"ac:0b:fb:c8:4a:e4"  
  
OK  
AT+CIPMUX=1  
  
OK  
AT+CIPSERVER=1,80  
  
OK  
WIFI DISCONNECT  
WIFI CONNECTED  
WIFI GOT IP  
0,CONNECT  
  
+IPD,0,331:GET / HTTP/1.1  
Host: 192.168.25.168  
Connection: keep-alive  
User-Agent: Mozilla/5.0 (Windows NT 10.  
Accept: /*/  
Origin: http://127.0.0.1:8080  
Referer: http://127.0.0.1:8080/  
Accept-Encoding: gzip, deflate  
Accept-Language: en-US,en;q=0.9
```

Putty showing resources being  
requested

# Performance

Parameters:

ESP01 connected through a router  
Target Device -> Router (Less than 1m)

Router -> ESP01 (Less than 1m)

Average Latency (Calculated by taking average of response time):  
 $(483 + 186 + 193 + 219) / 4 = 270.25 \text{ ms}$

Throughput: ~6Mbps  
Max Range: ~200m

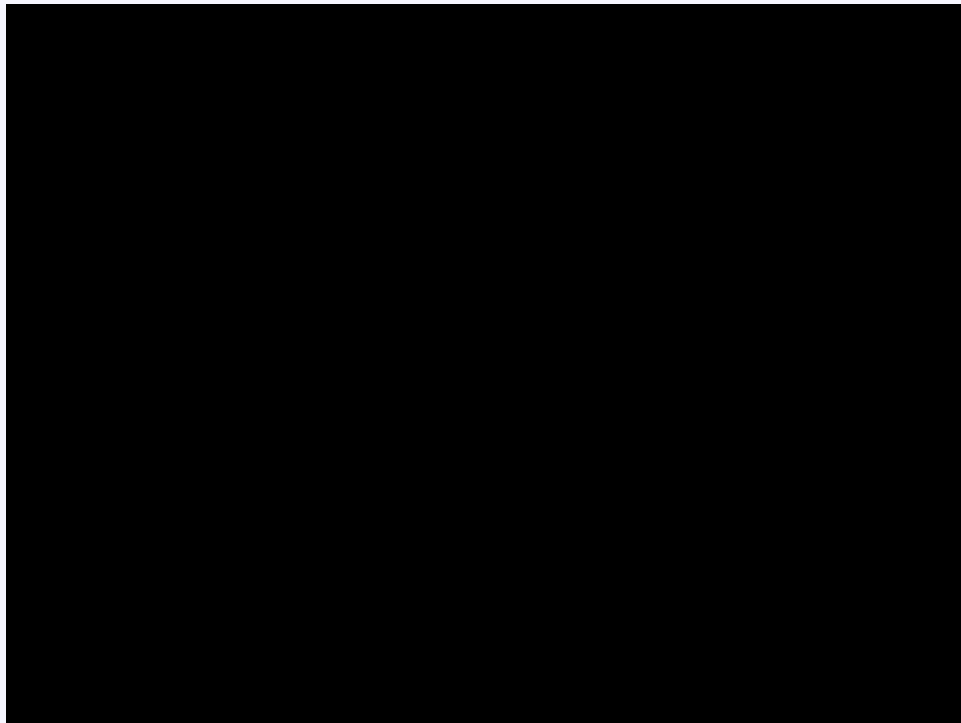
```
C:\Users\johnn>ping 192.168.25.168

Pinging 192.168.25.168 with 32 bytes of data:
Reply from 192.168.25.168: bytes=32 time=483ms TTL=255
Reply from 192.168.25.168: bytes=32 time=186ms TTL=255
Reply from 192.168.25.168: bytes=32 time=193ms TTL=255
Reply from 192.168.25.168: bytes=32 time=219ms TTL=255

Ping statistics for 192.168.25.168:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 186ms, Maximum = 483ms, Average = 270ms
```

Using CMD ping to measure latency

# Demo



The background features decorative geometric patterns in the corners, consisting of thin blue lines, dots, and circles. In the top-left, there are several parallel lines and a small cluster of dots. In the top-right, a circle with a dot inside is connected to a line. In the bottom-left, a circle with a dot inside is connected to a line, and there are several parallel lines. In the bottom-right, there are several parallel lines and a small cluster of dots.

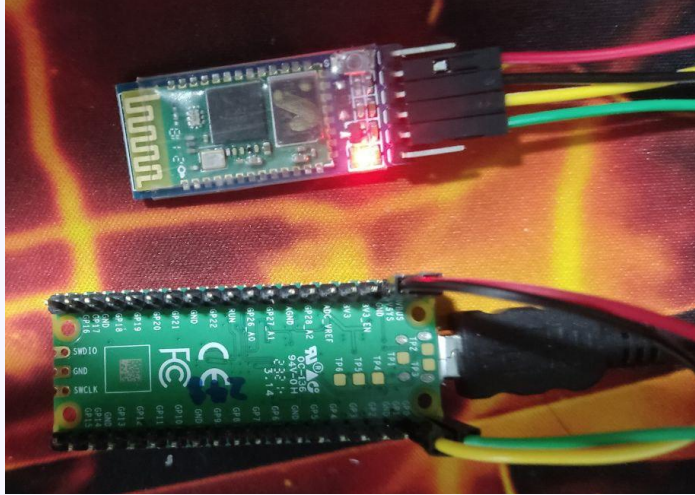
# 04

---

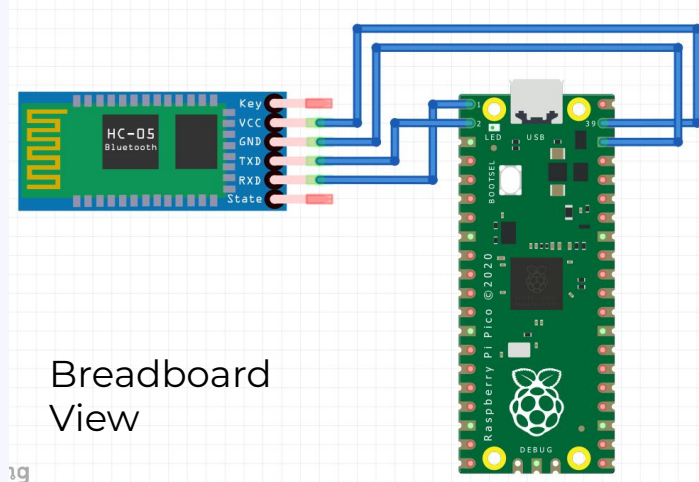
## Bluetooth



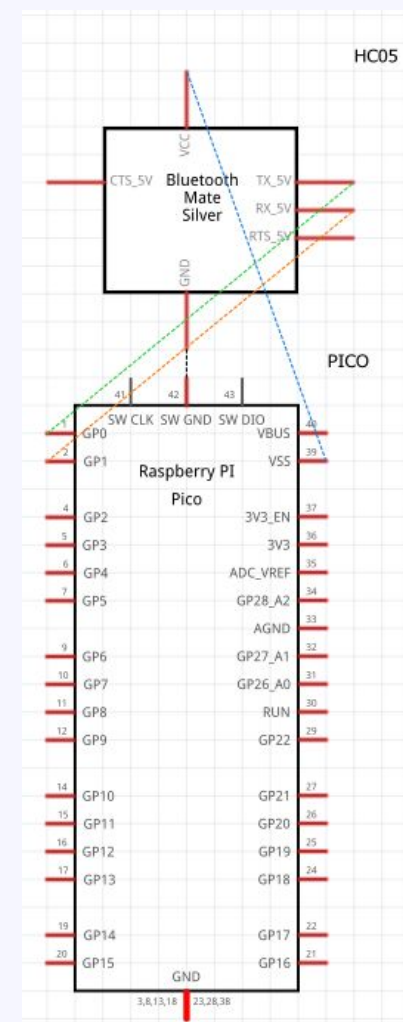
# Setup



Actual Setup



Breadboard View



Schematic View

# Config

## Pico Config-

UART0

TX\_PIN 0, RX\_PIN 1

Baud Rate 9600

Databits 8 | Stopbits 1 | Parity None

## Pin Connection-

Pico GP0 (TX) -> HC05 (RX)

Pico GP1 (RX) -> HC05 (TX)

Pico VSYS -> HC05 (VCC)

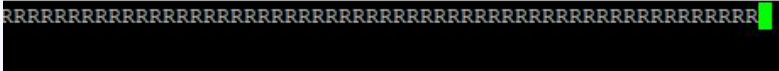
Pico GND -> HC05 (GND)

# Testing

Conducted black box testing

1) Testing Criteria

- a) 'HC05' should show up in bluetooth devices
- b) 'HC05' should be connectable through bluetooth
- c) Serial comms should be avail once bluetooth connected and able to connect
- d) Output should display 'R' (Sent by Pico) everytime something is inputted from Device to Pico



- Shows the 'R' being returned on every keystroke

# Serial Bluetooth Terminal (Android)

# Troubleshooting

- 1) Hold BOOTSEL (black button) on HC05 and connect to VSYS to enter BOOT MODE (Slow red flash every 2s)
- 2) Send the following command through PICO UART to HC05 to restore default settings
- 3) Reconnect HC05

## 4. Restore default

Command	Respond	Parameter
AT+ORGL	OK	-

Default state:

Slave mode, pin code :1234, device name: H-C-2010-06-01 ,Baud 38400bits/s.

# Performance

Parameters:

Target device is placed within 10cm  
of HC05

Average Latency (Calculated by  
taking average of response time):

$$[(191-136) + (997-887) + (818-778) + (843-667) + (562-444)] / 5 = 99.8\text{ms}$$

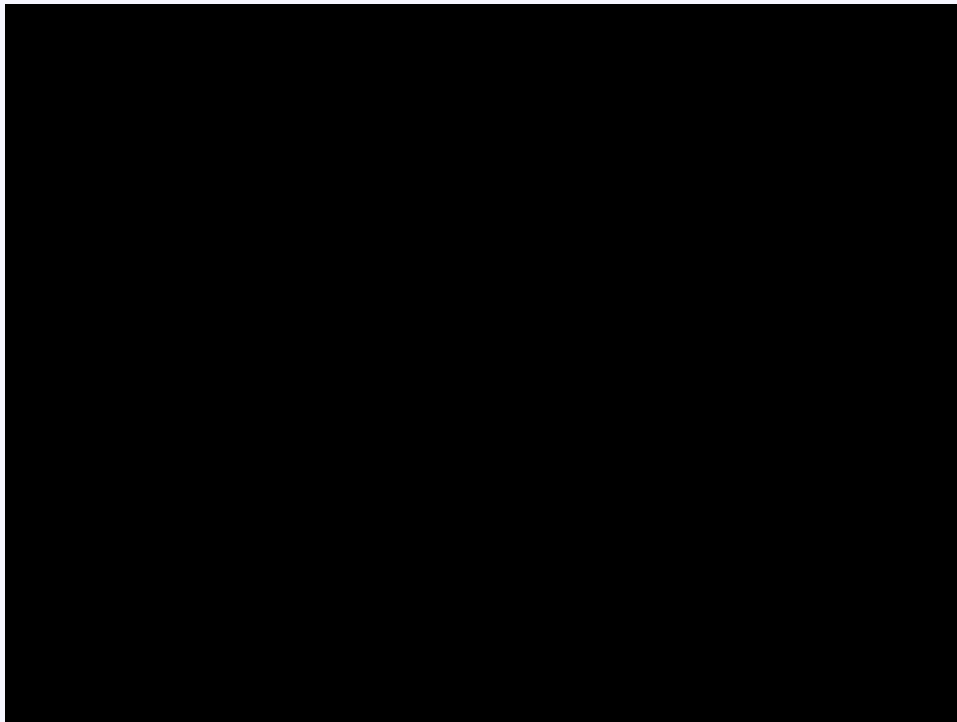
Throughput: 2.1Mbps(Max) / 160 kbps

Max Range: ~10m

```
04:47:11.716 Connecting to HC-05 ...
04:47:16.143 Connected
04:47:18.136 hi
04:47:18.191 RRRR04:47:18.887 hi
04:47:18.997 RRRR04:47:19.778 hi
04:47:19.818 RRRR04:47:20.667 hi
04:47:20.843 RRRR04:47:21.444 hi
04:47:21.562 RRRR04:47:22.176 hi
04:47:22.270 RRRR04:47:22.828 hi
04:47:22.903 RRRR04:47:23.571 hi
04:47:23.696 RRRR04:47:24.200 hi
04:47:24.335 RRRR04:47:24.908 hi
04:47:25.034 RRRR04:47:25.486 hi
04:47:25.560 RRRR04:47:26.080 hi
04:47:26.106 RRRR04:47:26.584 hi
04:47:26.616 RRRR04:47:27.114 hi
04:47:27.155 RRRR04:47:27.571 hi
04:47:27.596 RRRR04:47:28.024 hi
04:47:28.120 RRRR04:47:28.409 hi
04:47:28.523 RRRR04:47:28.778 hi
04:47:28.842 RRRR04:47:29.147 hi
04:47:29.249 RRRR04:47:29.513 hi
04:47:29.635 RRRR04:47:29.879 hi
04:47:29.965 RRRR04:47:30.241 hi
04:47:30.275 RRRR04:47:30.623 hi
04:47:30.683 RRRR04:47:31.008 hi
04:47:31.096 RRRR04:47:31.487 hi
04:47:31.690 RRRR04:47:31.858 hi
04:47:31.917 RRRR04:47:32.244 hi
04:47:32.318 RRRR04:47:32.630 hi
04:47:32.813 RRRR04:47:32.911 hi
04:47:33.032 RRRR04:47:33.230 hi
04:47:33.331 RRRR04:47:33.513 hi
04:47:33.549 RRRR04:47:33.842 hi
04:47:34.064 RRRR04:47:34.199 hi
04:47:34.264 RRRR04:47:34.500 hi
04:47:34.569 RRRR04:47:34.832 hi
04:47:34.881 RRRR
```

Using Serial Bluetooth Terminal  
for testing

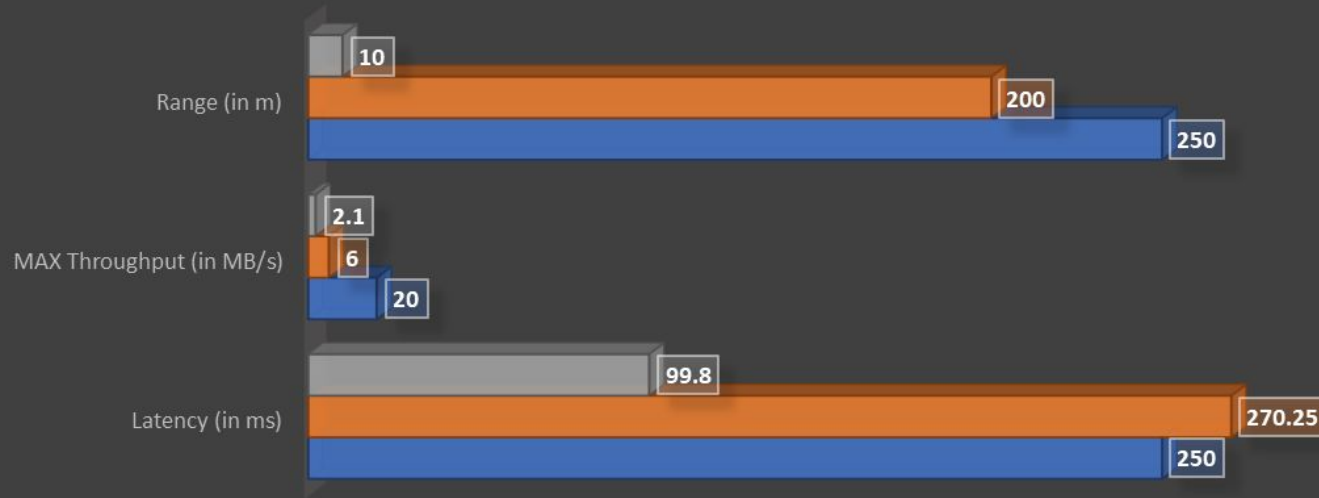
# Demo



# Comparison

## PERFORMANCE COMPARISON

■ HC05 via UART   ■ ESP01 via UART   ■ M5Stick via UART



### Chose ESP01:

- Lightest weight (Car already very heavy)
- High Range
- Enough Throughput (Only sending JSON)
- Acceptable Latency

	Latency (in ms)	MAX Throughput (in MB/s)	Range (in m)
■ HC05 via UART	99.8	2.1	10
■ ESP01 via UART	270.25	6	200
■ M5Stick via UART	250	20	250



The top corners of the slide feature abstract geometric designs. These include thin blue lines, small solid blue dots, and concentric circles, all arranged in a way that suggests a technical or architectural theme. The patterns are more dense in the top right corner and more sparse in the top left.

# Thank You!

---

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon** and infographics & images by **Freepik**

The bottom corners of the slide continue the abstract geometric theme seen at the top. They consist of thin blue lines, small solid blue dots, and concentric circles, arranged in a way that suggests a technical or architectural theme. The patterns are more dense in the bottom right corner and more sparse in the bottom left.