Car Accident Detection

Giovanni Gambigliani Zoccoli

IoT and 3D Intelligent Systems

Trasporto ideale

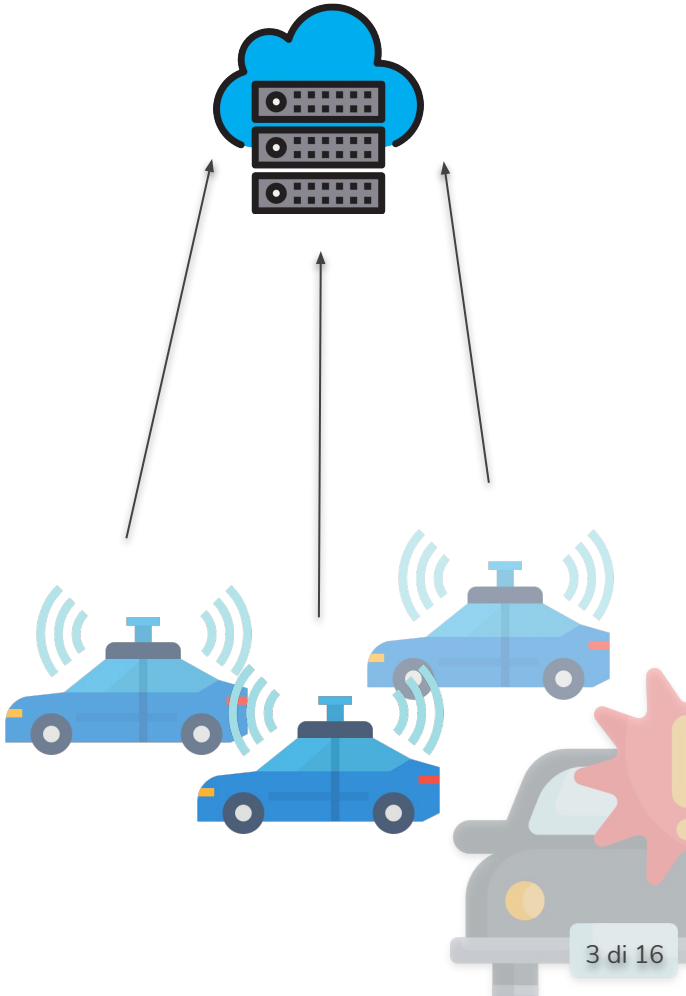
- Non subire incidenti
- Evitare lunghe code
- Puntualità

➔ Dispositivo in grado di rilevare e segnalare incidenti

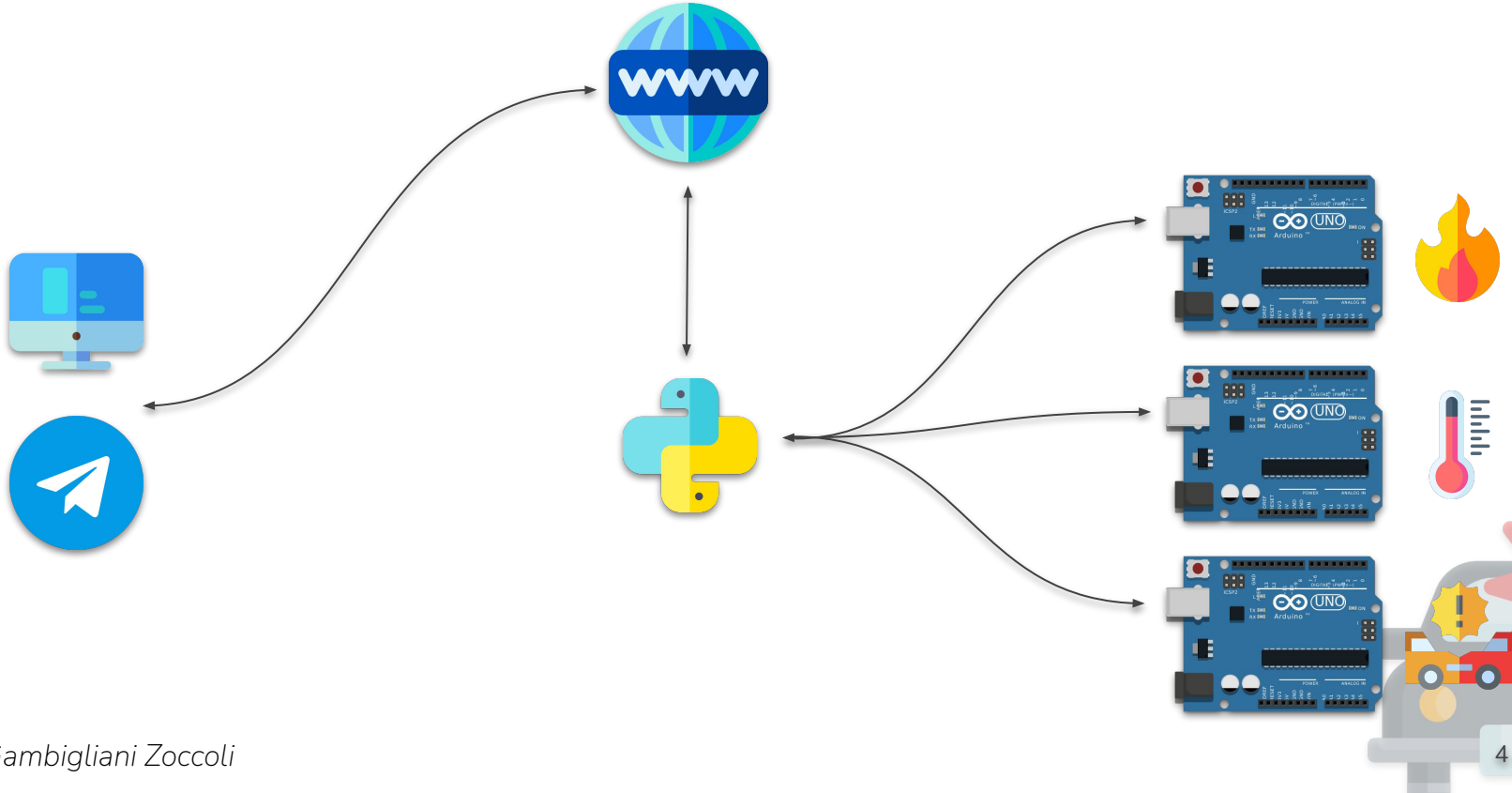
Idea

Realizzare un dispositivo in grado di:

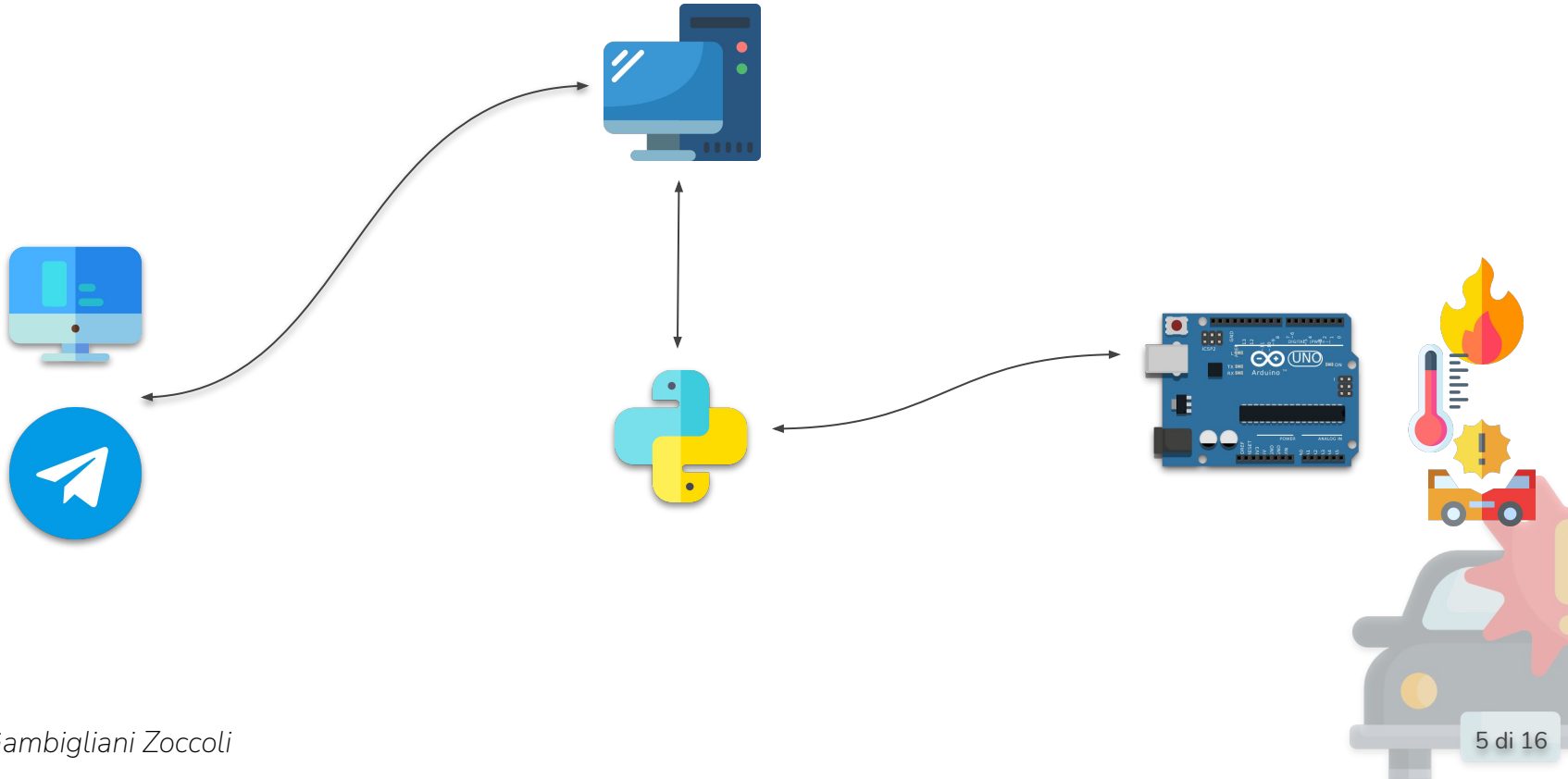
- Rilevare incidenti
- Segnalare posizione e dati sull'incidente
- Tempo reale



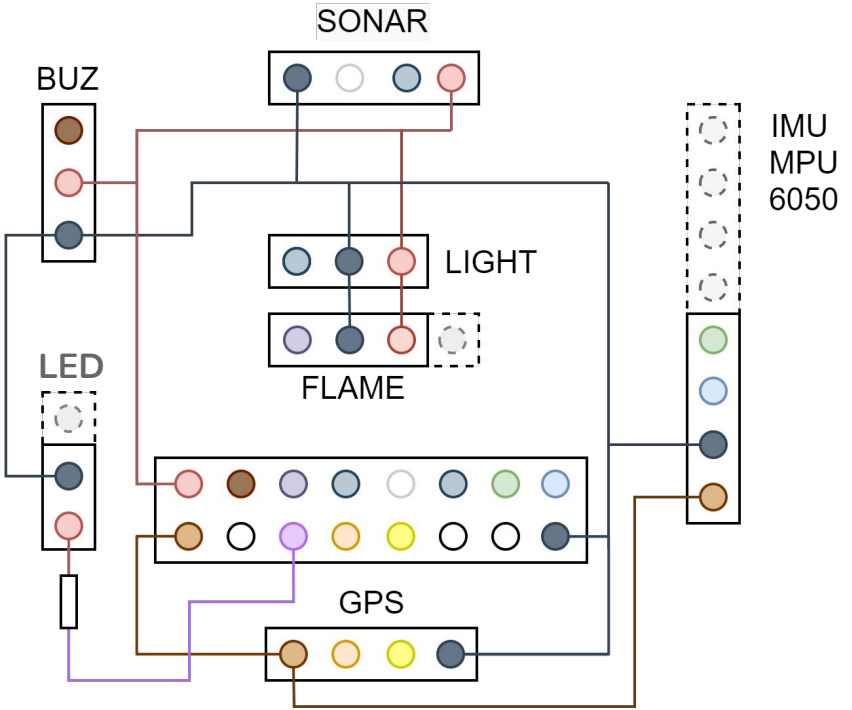
Architettura

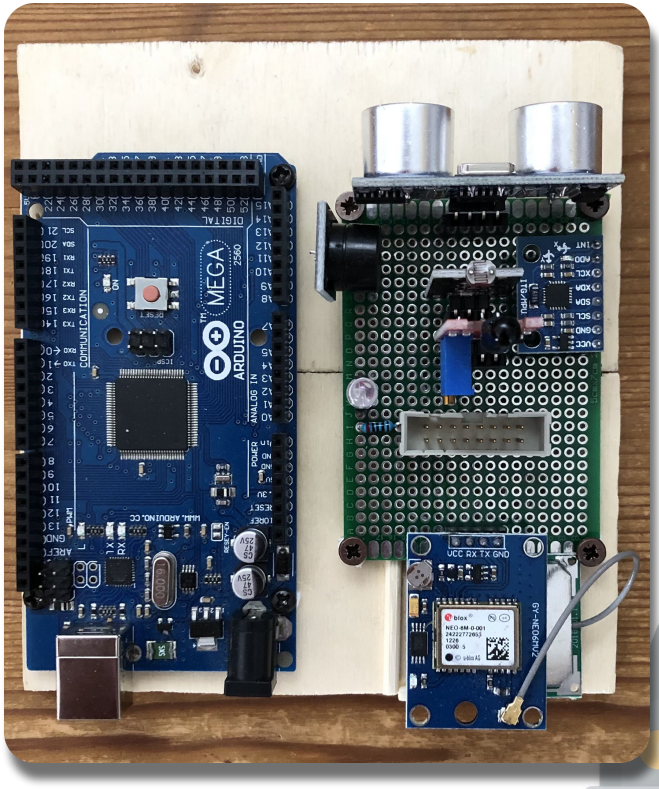


Architettura - Prototipo

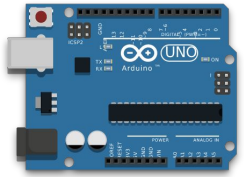


Prototipo



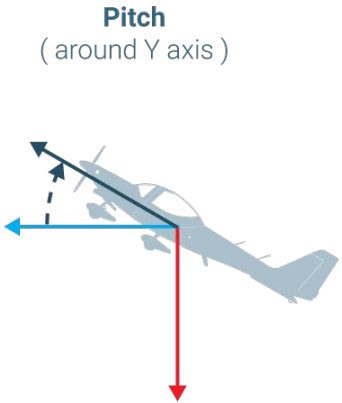


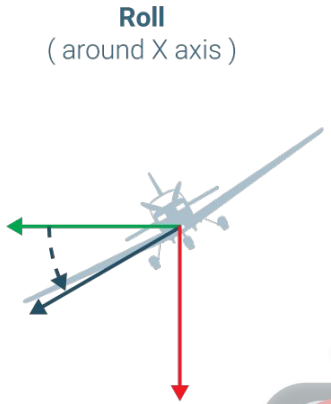
Arduino - Sensori e Attuatori



Sensori e attuatori:

- Flame → Light
- Ultrasonic (***HC-SR04***)
- Buzzer
- GPS (***NEO-6M***)
- IMU (Inertial Measurement Unit) (***GY-521***)
- Led



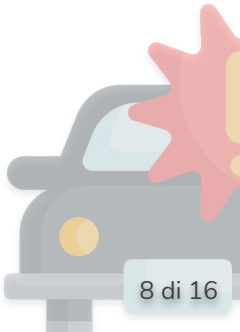


Codici Arduino

```
bool check_parameters(){
  //Incidente frontale o tamponamento
  if(distance <= threshold.distance)  frontal = true;

  //Caduta libera
  if(AcZ + AcX + AcY < threshold.fall) fall = true;

  //Rilevazione di ribaltamento
  if(Roll >= threshold.tilt || AcZ < -threshold.fall*30) tilt = true;

  //Incidente laterale
  if(AcX - prec_acx > 1.2) detection = true;

  //Tamponamento subito
  if(AcY - prec_acy > 1.2) detection = true;

  //Controllo se è stato rilevato un fuoco
  fire = fire_read();

  prec_acx = AcX;
  prec_acy = AcY;
  prec_acz = AcZ;

  return (frontal | tilt | fall | fire | detection);
}
```

```
void send_flame_light_temp_data(){
  uint8_t flame = map(flame_val, 0, 1024, 0, 253);
  uint8_t light = map(light_val, 0, 1024, 0, 253);
  uint8_t temp = abs(int(Tmp));
  uint8_t data[] = {0xFE, 0x04, flame, light, temp, 0x00, 0xFF};
  if(Tmp < 0){
    data[5] = 0x01;
  }
  for (uint8_t i = 0; i < sizeof(data); i++) {
    Serial.write(data[i]);
  }
}
```

```
void accident_detected(){
  tone(buz_pin, 500);
  while(!update_gps_data()){}
  send_data();
  //Aspetto 500ms per dare il
  //tempo al bridge di rispondere
  delay(500);
  //Controllo che il bridge abbia
  //ricevuto i dati
  bridge_ack();
  noTone(buz_pin);
  //INCIDENTE SEGNALATO E RICEVUTO
  delay(60000*10); //aspetto 10 minuti
  reset_fun();
}
```

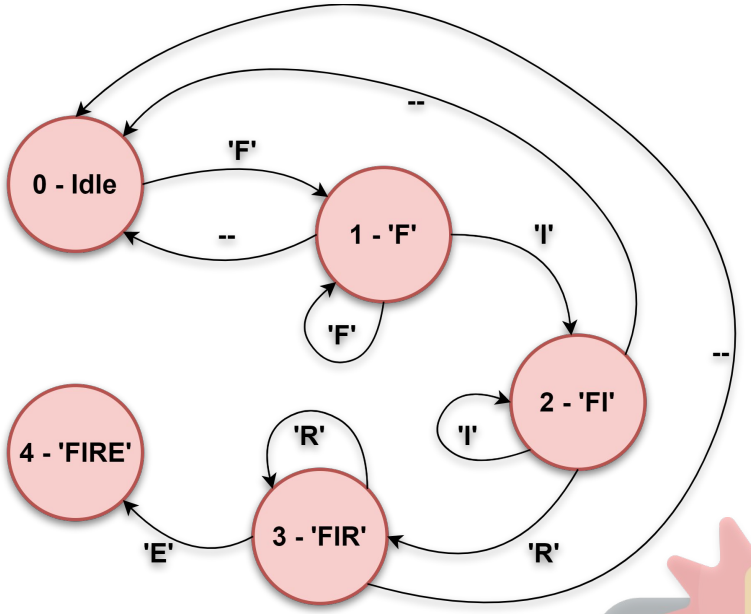


Python - Bridge

Struttura generale del pacchetto:

Payload:

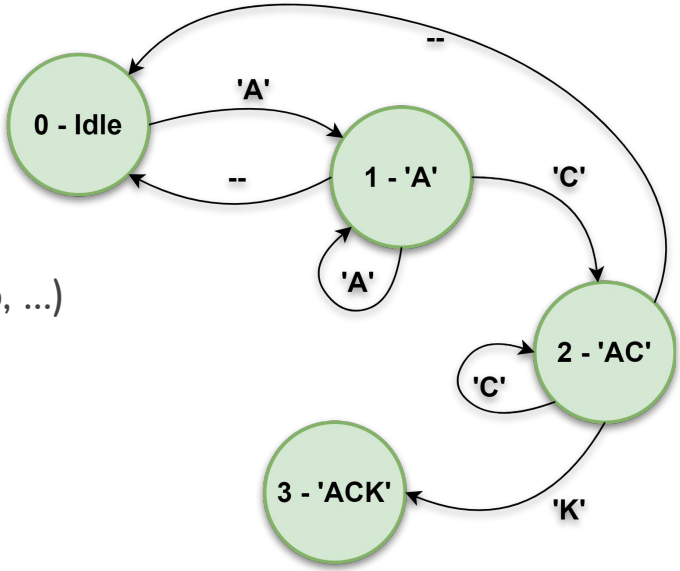
- Flame
- Light
- Temperatura



Python - Bridge

Payload:

- Posizione (Latitudine e Longitudine)
- Ora
- Data
- Informazioni sull'incidente (Frontale, Fuoco, ...)
- Temperatura
- Targa della vettura

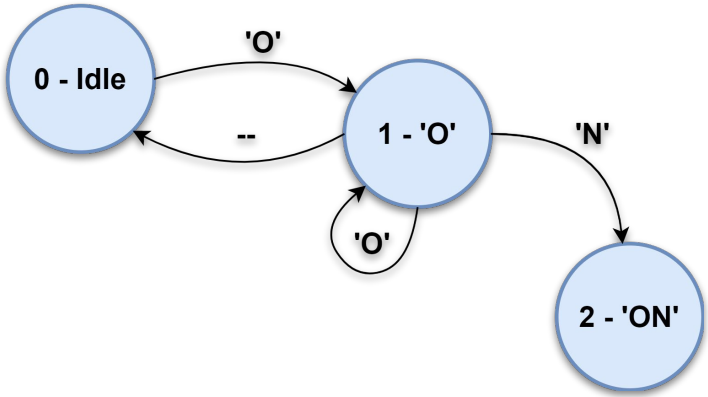


Api Restful → **POST** */api/v1/accidents*

Python - Bridge

Payload:

- Posizione (Latitudine e Longitudine)
- Velocità



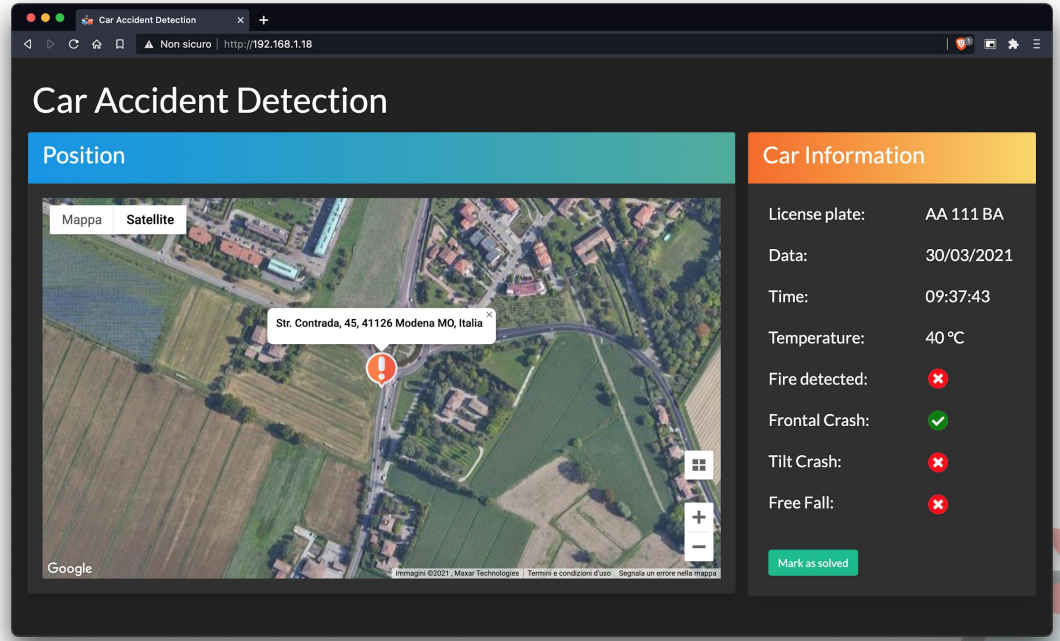
Raggio: 3 - 5 - 20 Km

Tempo: 10 - 20 - 30 minuti

Api Restful → **GET** */api/v1/accidents*

Flask - Dashboard


```
{
  "accidents": [
    {
      "date": "12/03/2021",
      "fall": false,
      "fire": false,
      "frontal": true,
      "id": 1,
      "lat": 44.540967,
      "license_plate": "AA 111 AA",
      "lng": 10.931247,
      "reported": false,
      "temperature": 40,
      "tilt": false,
      "time": "19:08:08"
    }
  ]
}
```

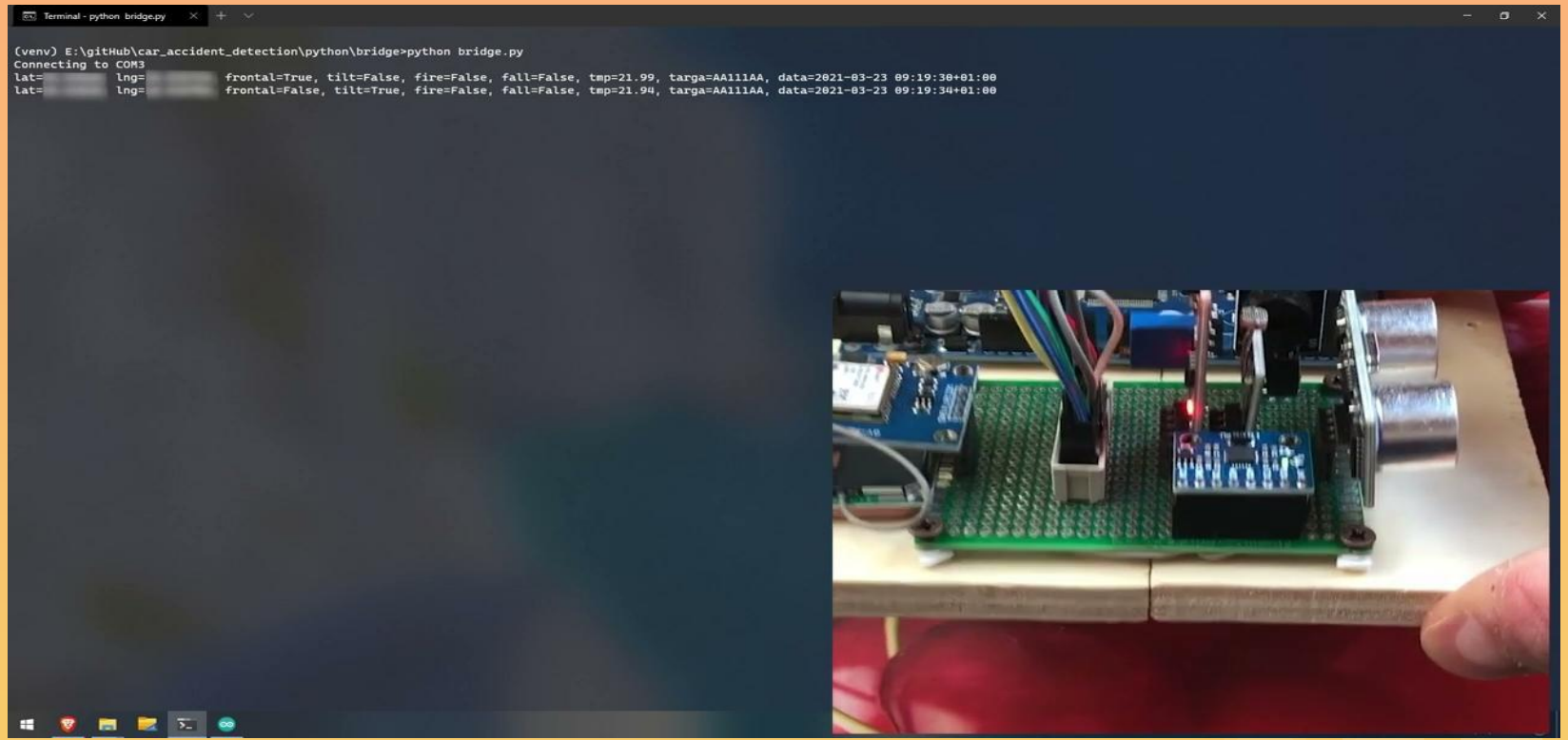
Telegram

Azioni disponibili:

- Aggiungere una auto
- Eliminare un auto
- Visualizzare la lista della auto registrate

➔ Segnalazione real-time dell’incidente

Demo...


Terminal - python bridge.py
(venv) E:\gitHub\car_accident_detection\python\bridge>python bridge.py
Connecting to COM3
lat=[redacted] lng=[redacted] frontal=True, tilt=False, fire=False, fall=False, tmp=21.99, targa=AA111AA, data=2021-03-23 09:19:30+01:00
lat=[redacted] lng=[redacted] frontal=False, tilt=True, fire=False, fall=False, tmp=21.94, targa=AA111AA, data=2021-03-23 09:19:34+01:00

Conclusioni e possibili sviluppi futuri

Grazie per l’attenzione

Giovanni Gambigliani Zoccoli