



## **IOT-BASED ACCIDENT DETECTION SYSTEM – PROJECT PROGRESS**

VIDEO 3 (CTP1)

# Hardware Integration



## Sensor and Output Device Setup

### ✓ Sensors used:

- ☐ **MPU6050** detects a crash based on sudden acceleration or tilt beyond a threshold.
- ☐ **vibration Sensor** – Detects strong vibrations (collision).
- ☐ **GPS** NEO-6M (**Module** – gets location (latitude & longitude)
- ☐ **MQ2 Sensor** – detects smoke/gas (fire risk)
- ☐ **Alcohol sensor MQ3** → detects drunk driving
- ☐ **DHT11** – monitors temperature and humidity
- ☐ **Seatbelt Detection** – simple switch or sensor to check if seatbelt is fastened

### ✓ Central Microcontroller Unit

**Arduino MKR WiFi 1010** processes sensor data and manages communication between sensors and output devices

### ✓ Output Devices for Alerts

- ☐ OLED display shows system status
- ☐ buzzer provides audible alerts,
- ☐ cancel button allows alert cancellation.



# ACCIDENT DETECTION LOGIC

## Set thresholds for:

- MPU6050 (impact)
- MQ2 (gas levels)
- MQ3 fire checking
- DHT11 (temperature spike)
- Vibration Sensor -- SW-420

## Threshold-Based Detection

- Arduino continuously reads sensor values and compared against predefined threshes.
- If any thresholds are exceeded (impact, smoke, alcohol, temperature, seatbelt off), it triggers an **accident detection**

## OLED displays messages like:

- Accident Detected”
- “Sending Alert in 10s...”
- “Alert Cancelled”

Helps the driver understand what’s happening in real time.

**Buzzer activates** to warn the driver.

## A 10-second countdown starts:

- If the **cancel button** is pressed, the alert is aborted.
- If not, the system proceeds to send an alert.



# Precautionary Features

## Temperature Thresholds:

- **Above 35°C:**
  - **OLED Message:** "Careful! Engine is heating!"
  - This warns the driver early before it becomes dangerous.
- **Above 60°C**
  - **OLED Message:** "Overheat detected! Sending alert..."
  - This triggers the **accident alert system** and sends data to Firebase.

## Buzzer Activation

- activate the **buzzer** when temperature exceeds 60°C to give an audible warning.
- Example: short beep for 35°C warning, long beep for 60°C alert.

## GPS Zone Awareness

- School/Hospital zone detection using Google Places API
- OLED prompts driver to slow down
- Purpose: Encourage safe driving in critical zones



# Data Transmission ( Cloud Communication)

Arduino sends JSON data via HTTPS POST to Firebase Cloud Function.



## Data Includes:

Sensor readings	GPS coordinates	Timestamp	Severity level	Flags ( <code>accidentdetected</code> , <code>overheatdetected</code> ,)
-----------------	-----------------	-----------	----------------	--

# Firebase Cloud Function Processing

## ❑ Data Verification & storage

- **Authentication** (for login)
- **Authentication** (Manages user login and access control.)
- **Cloud Functions** ( runs backend logic)
- **Fire store** ( Stores your sensor and alert data.)
- **Hosting** (Hosts your web dashboard.)

## ❑ The Cloud Function task :

**Verify Token** using Firebase Admin SDK → Ensures the data is coming from your authorized device.

**Extract *device id*** → Identifies which vehicle/device sent the data.

**Validate and Store Data** → Checks the data format and saves it to Firestore.

**Use Google Reverse Geocoding API** → Converts GPS coordinates into a readable address (e.g., "123 Main St, Perth").

## ❑ How Alerts Are Sent

**Step 1:** Arduino detects an accident and sends data to **Firebase Cloud Function** via HTTPS POST.

**Step 2:** Cloud Function processes the data:

- ✓ Verifies the device token using Firebase Admin SDK
- ✓ Stores accident data in **Firestore**
- ✓ Uses **Google Places API** to find:
  - Nearest hospital
  - Nearest police station

**Step 3:** Cloud Function sends an **SMS alert via Twilio** to the predefined family contact.



# Fire store Data Storage (Backend)

## What is Fire store?

Firestore is a cloud database inside Firebase that stores all the data sent by the Arduino.

### ❑ What Fire store Stores:

#### 1. Raw Sensor Data:

- Accelerometer reading (accel\_g)
- Temperature (temp\_c)
- Seatbelt status (true/false)
- GPS coordinates (latitude, longitude)
- Vehicle ID (CAR123)
- Severity level (based on impact force)
- Device token (for authentication)
- Timestamp (e.g., 2025-10-15T22:42:00Z)

#### 2. Alert Metadata:

- Twilio alert status (sent or not)
- Predefined user medical info (e.g., blood group, allergies)
- GPS address (converted using Google Reverse Geocoding API)

#### 3. Event Flags:

- *accident\_detected*: true/false
- *overheat\_detected*: true/false
- *seatbelt\_off*: true/false

### ❑ Purpose of Stored Data:

- ✓ Accident history tracking
- ✓ Insurance reporting
- ✓ Real-time dashboard display



# Web Dashboard (frontend)

dashboard is a **website** that shows the data stored in Firestore.

dashboard is for **real-time monitoring and visual**

## ***What the Dashboard Displays:***

- Live sensor readings ( Current temperature,overheat , acceleration, seatbelt status)
- Accident status (detected or not)
- Twilio alert status
- Historical graphs (acceleration, temperature)
- Accident logs with time and location

It uses:

**HTML/CSS/JavaScript** for layout

**Firebase SDK** to read data from Firestore

**Google Maps API** to show location

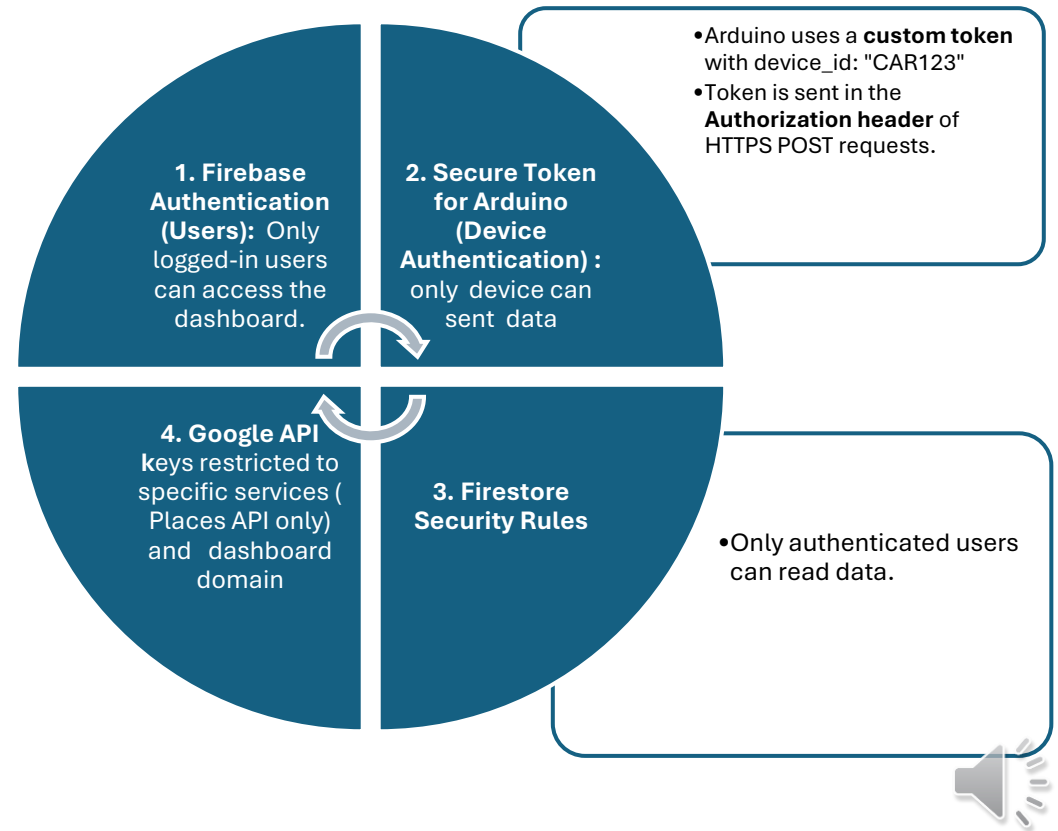
**Chart.js** to show graphs

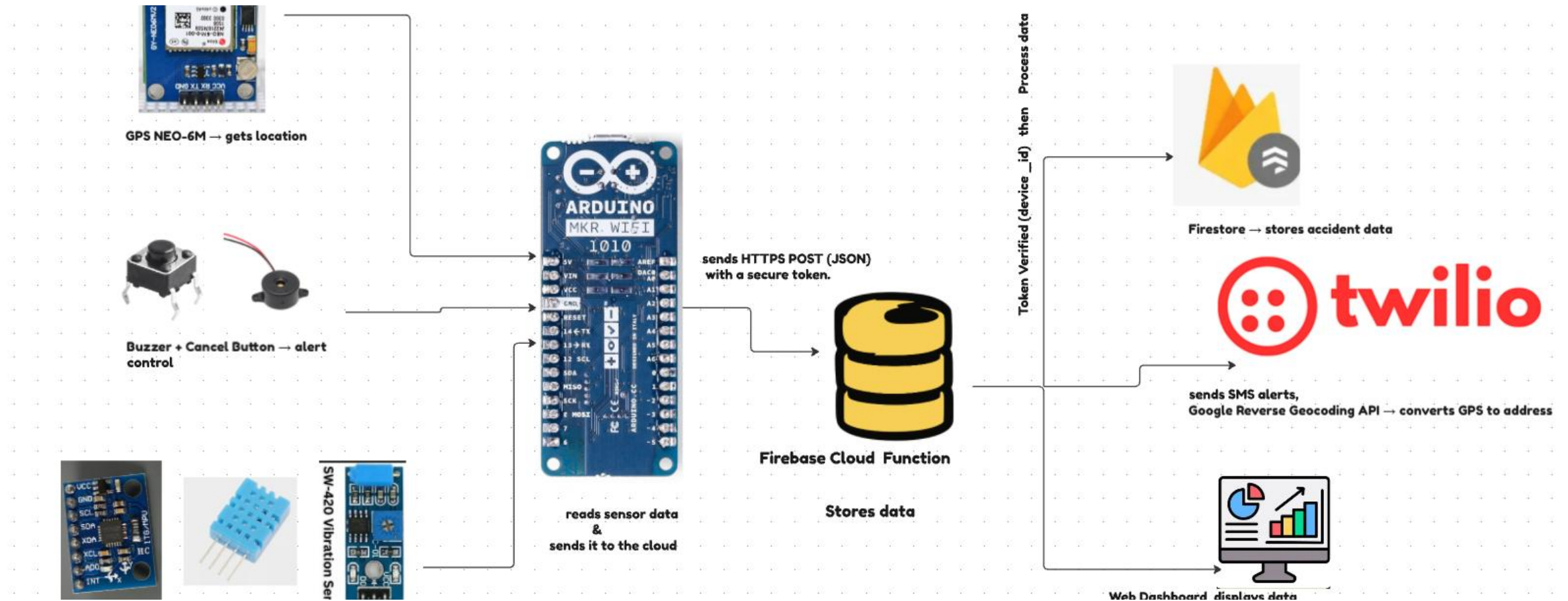






# Security Steps Taken





# System Processing Flow



# Conclusion & future goals



## Conclusion:

Project is in the **design and planning phase**

All modules and logic have been **mapped out**

**A lot modification has been done compared to the project form .**

System is designed to be **real-time, cloud-connected, and driver-friendly**



## Future Goals:

Begin hardware integration and sensor calibration

Implement accident detection logic

Build and test the web dashboard

Integrate Firebase Cloud Function , Fire store , Twilio alert system



## Future Improvements:

Add **offline mode** using LoRa communication

Improve dashboard UI/UX

**Add** fatigue detection (Eye Blink Monitor , Time-Based Alerts and voice alert )



# Project Timeline for CTP 2

Video 1 :final Plan &  
project Outline Progress

Video 2: Report on the  
progress

Video 3:Reflection and  
final status

