IoT Project Deliverables

Location Tracker

Name	Roll Number
Muhammad Anas Asim	21L-5789
Muhammad Abdullah Sami	21L-5837
Syed Farhan Jafri	21L-6074

1. Project Aims

The primary aim of this project is to design and develop a portable IoT-based location tracking device capable of real-time geolocation. The project seeks to:

- **Enable Accurate Real-Time Tracking:** Utilize GPS technology to obtain precise location data of the target.
- **Establish Reliable Communication:** Transmit location data to a cloud server using wireless technologies.
- **Implement Geofencing Functionality:** Define virtual boundaries and trigger alerts when the tracked object enters or exits specified zones using Rule Based Restrictions.
- **Detect Anomalous Movement Patterns:** Integrate an AI model to analyze travel behaviour and identify irregularities such as unexpected stops, jumps or deviations from typical routes.
- **Design a Compact and Cost-Effective Device:** Assemble the hardware using affordable components while maintaining portability.
- **Develop a User-Friendly Interface:** Create a simple and intuitive front-end interface to visualize real-time tracking.

2. Project Flow

Step 1: ESP32 → HiveMQ Cloud

- ESP32 reads GPS data (latitude, longitude, speed).
- Publishes data to HiveMQ Cloud via MQTT on the `location_tracker/device/location` topic.
- **Libraries Used**:
  - `TinyGPS++` (GPS parsing).
  - `PubSubClient` (MQTT communication).

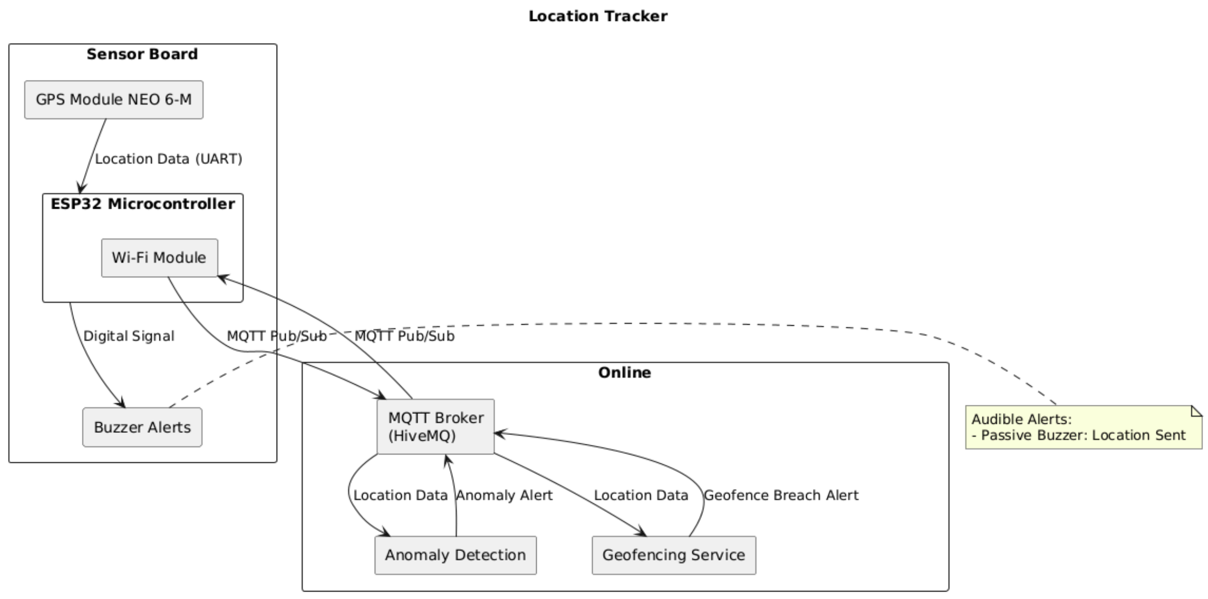
Step 2: HiveMQ Cloud → Azure Services

- **Azure Function (Free Tier)**:
  - Subscribes to HiveMQ's `location_tracker/+/location` topic.
  - Forwards data to **Azure Anomaly Detector** (free tier) for analysis.
- **Anomaly Detection**:
  - Azure Anomaly Detector flags anomalies (e.g., sudden speed spikes, route deviations).
  - Alerts are sent back to HiveMQ Cloud via the `location_tracker/device/alerts` topic.

Step 3: Frontend Dashboard

- **React Web Page**:
  - Subscribes to HiveMQ's MQTT topics.
  - Visualizes real-time GPS data and alerts on a map.

3. Sensor Board Block Diagram



4. MQTT Topics

Topic	Use
location_tracker/device/location	Publishes complete location data
location_tracker/device/location/latitude	Latitude coordinates only updates
location_tracker/device/location/longitude	Longitude coordinates only updates
location_tracker/device/status	For device health monitoring(battery, signal strength, errors). Sent periodically.
location_tracker/device/alerts	To use for centralized alert systems.
location_tracker/device/alerts/geofence	Dedicated geofence breaches (exit of defined zones). High-priority
location_tracker/device/alerts/anomaly	AI-detected anomalies (unexpected stops).

5. Details of Cloud AI Model

Model Type:

- **Azure Anomaly Detector (Univariate Time-Series Model)**
  - Pre-trained AI service optimized for detecting anomalies in sequential data (e.g., GPS coordinates, speed).

- Algorithm: **Spectral Residual (SR)** for fast, unsupervised detection without training data.

Key Parameters:

1. `sensitivity`:
   - Adjusts anomaly detection strictness (range: `0–99`).
   - Example: `sensitivity=90` flags minor deviations (high sensitivity).
2. `granularity`:
   - Time interval between data points (set to `per second` or `per minute` based on GPS sampling rate).
3. `maxAnomalyRatio`:
   - Maximum percentage of data points flagged as anomalies (default: `0.25`).

Input Data Format:

```
{
  "timestamp": "2023-10-05T14:30:00Z",
  "value": 45.5  // e.g., speed (km/h), distance from home (meters), or route deviation score
}
```

Output:

- `isAnomaly`: Boolean (`true`/`false`).
- `severity`: Anomaly score (`0`–`1`, higher = more severe).
- `expectedValue`: Predicted normal value.

Accuracy:

- **Claimed Precision**: >95% for common time-series anomalies (per Azure documentation).
- **Limitations**:
  - Accuracy depends on data granularity (works best with stable sampling intervals).

- Struggles with sparse data (e.g., GPS outages >10 minutes).

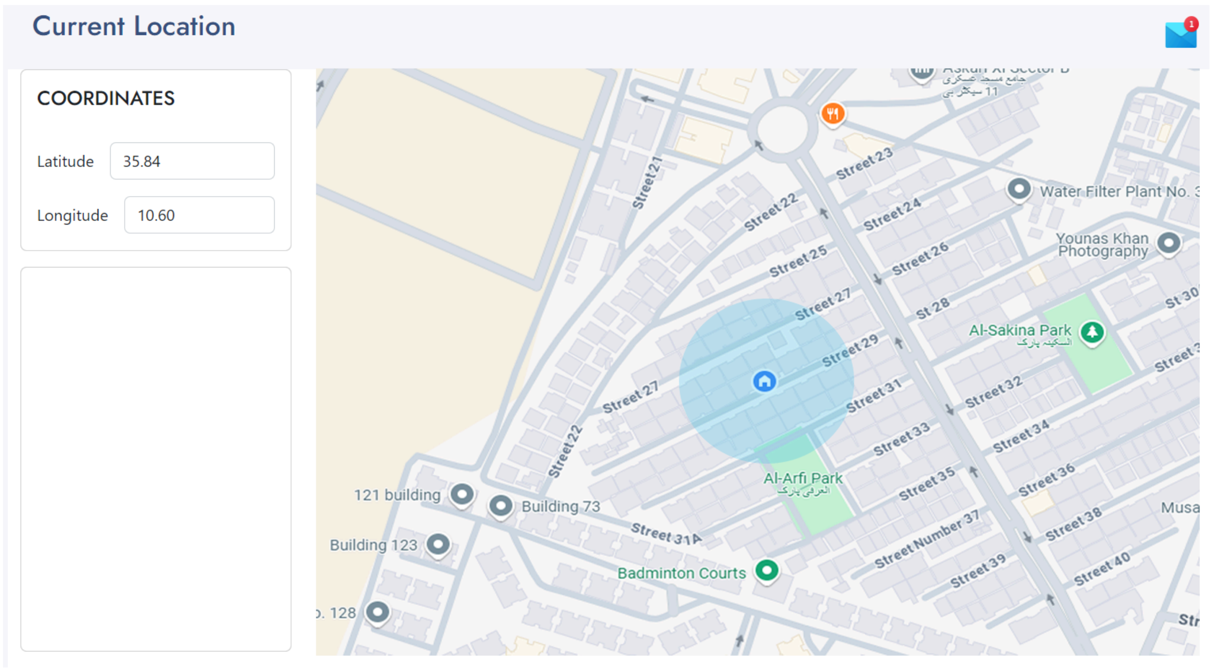
Integration with Azure IoT Hub:

- GPS data sent via **MQTT** to IoT Hub → Streamed to Anomaly Detector via **Azure Functions**.
- Alerts triggered using `severity` thresholds (e.g., `severity > 0.75`).

6. User Guide for Front-End

The project will have a sign in and registration of your GPS device to ensure the proper data transfer from the GPS module to the web application. The website will be able to get the exact location of the device via a single button in the web app.

Further, the app will support a notification based system which will alert users through a message icon (as shown in the screenshot below). This will alert the user when it detects the GPS module/the object has left the area marked by the geofence and also generate an alert when an anomaly is exhibited by the module indicating potential deviations from expected travel patterns such as impossible scenarios, location jumps etc.



Current Location

COORDINATES

Latitude: 35.84

Longitude: 10.60