IOT Phase 4: DEVELOPMENT PART 2

Project 10: Traffic Management System

To implement a smart traffic management system into a mobile app, we need to integrate various technologies and APIs.

Code snippet using the Google Maps API and Firebase Realtime Database to get traffic information and display it in a mobile app:

1. Set up the project:

```
import UIKit
import GoogleMaps
import Firebase
class ViewController: UIViewController, GMSMapViewDelegate {
    var mapView: GMSMapView!
    var trafficRef: DatabaseReference!
    override func viewDidLoad() {
        super.viewDidLoad()
        // Initialize Firebase
        FirebaseApp.configure()
        trafficRef = Database.database().reference().child("traffic")
        // Initialize Google Maps
        let camera = GMSCameraPosition.camera(withLatitude: 37.7749, longitude: -122.4194, zoom: 12.0)
        mapView = GMSMapView.map(withFrame: CGRect.zero, camera: camera)
        mapView.delegate = self
        self.view = mapView
        observeTrafficData()
    }
    func observeTrafficData() {
        trafficRef.observe(.value) { (snapshot) in
            // Handle traffic data changes in Firebase Realtime Database
            guard let trafficData = snapshot.value as? NSDictionary else { return }
            // Update the traffic conditions on the map
            self.updateTrafficOverlay(trafficData: trafficData)
        }
    }
    func updateTrafficOverlay(trafficData: NSDictionary) {
        mapView.clear()
        for (key, value) in trafficData {
            guard let info = value as? NSDictionary,
                  let latitude = info["latitude"] as? Double,
                  let longitude = info["longitude"] as? Double,
                  let condition = info["condition"] as? String else {
                continue
            }
            let marker = GMSMarker()
            marker.position = CLLocationCoordinate2D(latitude: latitude, longitude: longitude)
            marker.title = "\(key)"
            marker.snippet = "Traffic condition: \(condition)"
            // Set marker color based on traffic condition
            switch condition {
            case "low":
                marker.icon = GMSMarker.markerImage(with: .green)
            case "medium":
                marker.icon = GMSMarker.markerImage(with: .orange)
            case "high":
                marker.icon = GMSMarker.markerImage(with: .red)
            default:
                marker.icon = GMSMarker.markerImage(with: .gray)
            }
            marker.map = mapView
        }
    }
}
```

2. Prepare the Firebase database structure:

Before running the app code, we need to configure and populate our Firebase Realtime Database with traffic information.

```
{
  "traffic" : {
    "location1" : {
      "latitude" : 37.786865,
      "longitude" : -122.406044,
      "condition" : "medium"
    },
    "location2" : {
      "latitude" : 37.765191,
      "longitude" : -122.419209,
      "condition" : "high"
    },
    "location3" : {
      "latitude" : 37.791681,
      "longitude" : -122.407777,
      "condition" : "low"
    }
  }
}
```

This code sets up a simple mobile app with a Google Map embedded, connecting to a Firebase Realtime Database to fetch and display traffic information.