2211CS020517

IOT CASE STUDY

Case Study 1: Attendance Monitoring System Using a Camera Module

This case study focuses on developing an automated attendance monitoring system for classrooms. The system uses a camera module to capture images or videos of students as they enter, verifying their identities and marking their attendance. This reduces the manual effort required, increases efficiency, and ensures accurate attendance records.

Key Components:

1. **Camera Module**: Captures images of students.
2. **Microcontroller**: (e.g., Arduino) Manages the system and interfaces with the camera.
3. **Face Recognition Library**: Matches captured images with stored data.
4. **Database**: Stores student details and face templates.

Operation:

1. The camera captures images when students enter.
2. Face recognition algorithms verify identities.
3. Attendance is logged upon successful matching.
4. A success/failure message is displayed.

Embedded C Code:

```
#include <stdio.h>

#include <camera.h>

#include <face_recognition.h>

#include <attendance_db.h>   #define

CAMERA_PIN 7    void
```

```
initializeCamera() {     if
(camera_init(CAMERA_PIN) == 0) {
        printf("Camera Initialized\n");
    } else {
        printf("Camera Initialization Failed\n");
return;
    }
} int captureImage() {     if
(camera_capture() == 0) {
printf("Image Captured\n");
        return   1;                    }   else   {
printf("Image Capture Failed\n");
        return 0;
    }
} int recognizeFace() {     if
(face_recognize() == 0) {
printf("Face Recognized\n");
        return 1;       } else {
printf("Face Recognition Failed\n");
        return 0;
    }
} void markAttendance()
{        if
```

```
(attendance_mark() ==
0) {
printf("Attendance
Marked\n");
   } else {        printf("Failed to Mark
Attendance\n");
   }
} int main() {
initializeCamera();
while(1) {        if
(captureImage()) {
if (recognizeFace()) {
markAttendance();
          }
       }    }
return 0;
}
```

Case Study 2: IoT in Logistics and Fleet Management

This study highlights how IoT improves logistics and fleet operations. From real-time shipment tracking to predictive maintenance, the system leverages connected devices to optimize performance and reduce costs.

Key Components:

1. **GPS Modules**: Real-time tracking of vehicles and shipments.
2. **IoT Sensors**: Monitor vehicle health and environmental conditions.
3. **Central Control System**: Processes sensor data and provides analytics.
4. **Communication Protocols**: GSM, LTE, or LoRa for data transmission.

Benefits:

- **Real-Time Tracking**: Continuous monitoring of location and status.
- **Predictive Maintenance**: Prevents breakdowns using data analytics.
- **Route Optimization**: Reduces delays and fuel costs.
- **Condition Monitoring**: Ensures safe transport of sensitive goods.

Embedded C Code:

```
#include <stdio.h>

#include <gps.h>

#include <temperature.h>

#include <gsm_module.h>

#define TEMP_SENSOR_PIN 3

#define GPS_PIN 4

void initializeSensors() {

gps_init(GPS_PIN);

temperature_init(TEMP_SENSOR_PIN); }

void reportData(float latitude, float

longitude, float temperature) {     char

data[100];     sprintf(data, "Location: %.2f,
```

```
%.2f; Temp: %.2f C", latitude, longitude,

temperature);     gsm_send(data);

} int main() {     initializeSensors();     float latitude,

longitude, temperature;     while (1) {

gps_get_location(&latitude, &longitude);         temperature =

temperature_read(TEMP_SENSOR_PIN);

reportData(latitude, longitude, temperature);

delay(10000);

   }

return 0;

}
```

Case Study 3: IoT in Healthcare for Remote Patient Monitoring

IoT technologies enable continuous patient monitoring through remote systems. Data collected by sensors is transmitted to healthcare providers, allowing proactive interventions and reducing hospital visits.

Key Components:

1. **Wearable Devices**: Track vital signs like heart rate and glucose levels.
2. **IoT Sensors**: Monitor parameters like temperature and oxygen saturation.
3. **Communication Modules**: Use Wi-Fi or GSM for data transfer.
4. **Cloud Database**: Provides real-time access and analytics.

Embedded C Code:

```
#include <stdio.h>
#include <heartbeat_sensor.h>
#include <temperature_sensor.h>
#include <gsm_module.h>
#define HEARTBEAT_SENSOR_PIN 5
#define TEMP_SENSOR_PIN 3
void initializeSensors() {
heartbeat_sensor_init(HEARTBEAT_SENSOR_PIN);
temperature_sensor_init(TEMP_SENSOR_PIN);
} void sendAlert(float heartRate, float temperature) {     char alert[100];     sprintf(alert,
"ALERT! HeartRate: %.1f bpm, Temp: %.1f C", heartRate, temperature);
gsm_send(alert);
}
int main() {
initializeSensors();     float
heartRate, temperature;
while (1) {
      heartRate = heartbeat_read(HEARTBEAT_SENSOR_PIN);
temperature = temperature_read(TEMP_SENSOR_PIN);        if
(heartRate > 100 || temperature > 37.5) {
sendAlert(heartRate, temperature);
      }
      delay(10000);
```

```
    }
return 0;
}
```

Case Study 4: IoT and Augmented Reality for Enhanced Experiences

Integrating IoT with AR creates interactive, immersive experiences. Real-time IoT data is visualized in AR environments, enabling applications like guided tours and virtual training.

Key Components:

1. **IoT Devices**: Sensors and actuators for monitoring.
2. **AR Devices**: Smartphones or AR glasses for visualization.
3. **Communication Protocols**: Data transmission using Wi-Fi or Bluetooth.
4. **Visualization Software**: Renders interactive 3D overlays.

Embedded C Code:

```
#include <stdio.h>
#include <temperature_sensor.h>
#include <wifi_module.h>
#define TEMP_SENSOR_PIN 3
void initializeSystem() {
temperature_sensor_init(TEMP_SENSOR_PIN);
wifi_connect("YourSSID", "YourPassword");
```

```
} void sendIoTData(float temperature) {     char
jsonData[50];     sprintf(jsonData, "{ \"temperature\": %.2f
}", temperature);     wifi_send_data("http://ar-
server.local/api/data", jsonData);
} int main() {
initializeSystem();
float temperature;
while (1) {
temperature =
temperature_read(T
EMP_SENSOR_PI
N);
sendIoTData(temper
ature);
delay(5000);
   }
return 0;
}
```

Case Study 5: Wearable IoT Devices for Health and Fitness

Wearable devices such as fitness trackers revolutionize personal health monitoring. These devices collect data on physical activities and vital signs, offering actionable insights.

Benefits:

- **Health Monitoring**: Tracks metrics like heart rate and sleep quality.
- **Exercise Guidance**: Provides workout suggestions.
- **Preventive Healthcare**: Alerts users to potential health issues.

Embedded C Code:

```
#include <stdio.h>

#include <heartbeat_sensor.h>

#include <step_counter.h>

#include <bluetooth_module.h>

#define HEARTBEAT_SENSOR_PIN 2

#define STEP_COUNTER_PIN 4

void initializeWearable() {

heartbeat_sensor_init(HEARTBEAT_SENSOR_PIN);

step_counter_init(STEP_COUNTER_PIN);

bluetooth_init("WearableTracker");

} void sendHealthData(float heartRate, int steps) {     char healthData[50];

sprintf(healthData, "Heart Rate: %.1f bpm, Steps: %d", heartRate, steps);

bluetooth_send(healthData);


} int main() {

initializeWearable();

float heartRate;     int

steps;     while (1) {
```

```
        heartRate = heartbeat_read(HEARTBEAT_SENSOR_PIN);
steps = step_count(STEP_COUNTER_PIN);
sendHealthData(heartRate, steps);          delay(10000);
    }
return 0;
}
```