Use Case Document: **Movement Classification for Palletized Goods Using IoT and Data Analytics**

**Domain:** Logistics

**Use Case Name:** Movement Classification for Palletized Goods Using IoT and Data Analytics

**Use Case Description:** This use case aims to simulate and classify mechanical stress patterns experienced by palletized goods during different phases of transportation. Using IoT sensors (e.g., accelerometers, gyroscopes) and data analytics, the system will identify and differentiate between four distinct movement classes.

**Use Case Goal:** To accurately classify the type of movement (Maritime, Terrestrial, Lift, Idle) experienced by palletized goods using IoT sensor and Data analytics, in order to improve logistics handling and reduce product damage.

**Actor:** Logistics managers, warehouse staff.

**Pre-Conditions:**

* IoT sensors (accelerometer, gyroscope) are properly installed and functioning on pallets.
* Internet connectivity is available for data transmission.
* Data analytics system is deployed and running.

**Post-Conditions:**

* Sensor data is transmitted, processed, and classified into one of the four movement types.
* Visual dashboards are updated with timelines and movement classifications.
* Logistics insights (e.g., high-stress points) are generated.

**Logistics managers review data for operational insights.**

**Main Flow (Basic Path):**

**IoT Layer:**

* Attach IoT sensor modules to pallets to record:
  + Acceleration (X, Y, Z axes)
  + Angular velocity (gyroscope)
  + Vibration patterns
  + Optional: GPS (to aid in location tagging)
* Data collection at high frequency during transport and handling phases

**Data Analytics Layer:**

* Preprocess sensor data (noise filtering, normalization)
* Feature extraction (e.g., RMS acceleration, frequency domain features)
* Classification using ML algorithms (e.g., Random Forest, SVM, CNN for time-series)
* Output: Classify into one of the four states:
  1. **Maritime**
  2. **Terrestrial (Truck/Train)**
  3. **Lift (Forklift Handling)**
  4. **Idle (Warehouse Storage)**

**Technology Stack:**

* **Hardware:** MPU6050 (accelerometer + gyroscope), ESP32, battery pack
* **Connectivity:** Wi-Fi
* **Data Pipeline:** MQTT/HTTP → Cloud Storage (AWS/Azure/GCP)
* **Analytics:** Python (Pandas, Scikit-learn, TensorFlow), Jupyter Notebooks
* **Visualization:** Grafana dashboard to visualize movement classification.

**Alternative Flows:**

* **Sensor Malfunction or Disconnection**: IoT sensor stops transmitting data due to hardware failure or low battery. System detects missing or irregular data stream. Alert is generated and sent to the warehouse/logistics team.
* **Noisy or Incomplete Data:** Sensor data is too noisy, inconsistent, or missing segments. Data preprocessing flags low-quality data. The system may attempt to interpolate or clean the data.
* **Loss of Connectivity (Wi-Fi/Internet):** Pallet moves through a zone with no Wi-Fi coverag e. sensor stores data locally in buffer memory. Upon reconnecting, buffered data is uploaded in batch. System timestamps and aligns delayed data accordingly.
* **Misclassification by ML Model**: Movement is incorrectly classified due to model limitations. Anomaly detection flags unusual patterns or inconsistent transitions. Data is logged for model retraining. Optionally, a manual review can override classification in critical cases.

**Exception Handling:**

* **Sensor Hardware Failure:** Sensor is physically damaged or non-functional. System detects no incoming data for a predefined time. Status is updated as "Sensor Offline." Notification sent to maintenance/logistics team for inspection. Log entry created with timestamp for traceability.
* **Power Loss / Battery Drain:** ESP32 or sensor module runs out of power. System flags the last known battery status (if monitored). Logs the exact time of power loss. Alerts team for immediate battery replacement or recharge.
* **Data Upload Failure:** Data transmission to cloud fails due to network outage or server error. ESP32 caches data locally (in flash memory). Retry mechanism triggers periodic reattempts to upload data. On successful reconnection, buffered data is uploaded in sequence.
* **Machine Learning Model Crash:** ML inference process fails due to corrupted input, timeout, or server error. Invalid input is logged and discarded. Fallback logic can retry with previous data segment or apply a default rule (e.g., mark as "Unknown"). System health monitor restarts ML service if crash is detected.
* **Classification Conflict**: Two movement types are classified at the same timestamp (e.g., Maritime and Lift).Conflict resolution rule applied (e.g., prioritize Lift over Maritime during high acceleration).Segment flagged for manual review if confidence scores are close.
* **Incomplete Feature Extraction**: Missing values in time series prevent full feature set from being generated. System skips affected window .Logs issue and continues with next valid data window. Dashboard visually marks skipped sections.

**Possible Test Cases:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Input Data** | **Expected Outcome** |
| TP01 | Sensor data transmission from pallet. | Accelerometer+gyroscope data via ESP32 over wifi | Data is successfully received and logged in cloud storage |
| TP02 | Movement classification- maritime | Simulated acceleration and vibration pattern of sea transport | Output=“maritime”  Classification with confidence> 90% |
| TP03 | Movement classification- terrestrial | Simulated truck/train motion data | Output=“terrestrial”  Classification with confidence> 90% |
| TP04 | Movement classification- lift | Short, sharp lift pattern from forklift simulation | Output=“lift”  Classification with confidence> 90% |
| TP05 | Movement classification- idle | No movement very low acceleration values | Output=“idle”  Classification |
| TP06 | Handle sensor disconnection | Sudden loss of sensor data string | System logs disconnection and sends alert |
| TP07 | Data preprocessing- noise filtering | Raw sensor data with noise | Filtered and smoked signal ready for feature extraction |
| TP08 | Data upload failure and retry | Simulate network failure during data upload | Data is stored locally and uploaded once connection resumes |
| TP09 | Classification with incomplete data | Partial feature vector due to dropped packet | System skips classification or returns “unknown” status with appropriate flag |
| TP10 | Dashboard update and visualization | Valid classification result from ML model | Grafana dashboard updates with correct moment type and timestamp |
| TP11 | Conflict in classification output | Two classes output simantaneously | Conflict resolved using rules ; classification decision logged |
| TP12 | Sensor calibration accuracy | Known movement pattern with calibrated sensor | Output matches expected classification within defined error margin |

**Expected Outcome:**

* Accurate classification of movement/stress type with >90% accuracy
* Visual timelines of movement patterns per pallet
* Identification of high-stress points for logistics optimization
* Actionable insights to improve packaging, handling, or route planning

**Expected Deliverables**

1. Extend the use case document to have
   1. Sequence diagram
   2. Workflow diagram
   3. Functional Architecture diagram
   4. Database architecture diagram
2. UI / UX – If applicable
3. Work in progress code – Code repository Git
4. Test cases – Plan Vs Actuals
5. Summary Report
   1. Technical summary
   2. Challenges faced
   3. Lesson learnt
6. LinkedIn – Update project work