GaitMate Test Session Workflow

This document outlines the complete workflow and lifecycle of a gait analysis test session from the patient interface to backend coordination, MQTT communication, and WebSocket updates.

▯ Overview

- A test session is initiated by the patient via the Patient Dashboard.
- It proceeds through three structured steps: **Calibration**, **Wearing Device**, and **Starting Test**.
- Real-time updates flow via WebSocket; commands are issued through REST and routed via MQTT to IoT devices.
- Session states are managed in the database and backed by asynchronous processing.

▯ Step-by-Step Workflow

1. Entry from Patient Dashboard

- The user clicks "Start Test" → navigates to `PatientTestSession` page.
- Backend sends WebSocket `DEVICE_ALIVE` message as soon as device status is detected.

2. Step 1: Calibration Check

- WebSocket: Device status (`DEVICE_ALIVE`) received and shown.
- Frontend sends: `POST /api/commands` with action `CHECK_CALIBRATION`
- Result sent back from backend via WebSocket `CALIBRATION_STATUS`
- If not calibrated:
  - Show “Start Calibration” button and progress bar.
  - On click: `POST /api/commands` with action `START_CALIBRATION`
  - Calibration progress tracked via WebSocket updates (`CALIBRATION_STATUS`)

3. Step 2: Wear Device

- User wears the device and clicks “I’m Ready”
- Sends: `POST /api/commands` with action `CAPTURE_ORIENTATION`
- Backend listens to MQTT → emits WebSocket message `ORIENTATION_CAPTURED`

4. Step 3: Start Test Session

- User clicks “Let’s Go” button
- Frontend sends: `POST /api/test-sessions`

  ```
  {
    "timestamp": <epoch_millis>,
    "action": "START"
  }
  ```

- Backend creates a new test session:
  - State = `ACTIVE`
  - Returns: `testSessionId`

5. During Test

- Device streams live sensor data via MQTT
- Backend publishes to WebSocket: `/user/topic/data/sensor`
- UI displays real-time analysis or visualization

6. Stopping the Test

- User clicks "Stop"
- Frontend sends: `POST /api/test-sessions/{id}/stop`
- Backend:
  - Sets state = `PROCESSING`

    - Sends data to a processing microservice for analysis

7. Session Completion

- Backend or external microservice sets final state to:
    - `COMPLETED` if success
    - `FAILED` if error occurs

□ Session State Transitions

Trigger	State Transition
Click “Let’s Go”	`ACTIVE`
Click “Stop”	`PROCESSING`
Analysis Completed	`COMPLETED`
Processing Error	`FAILED`

□ Summary

- WebSocket used for real-time device status & data.
- MQTT bridges IoT devices with backend.
- Commands issued via REST and routed to devices.
- Test session entity created **only** at step 3 to avoid waste.
- Microservices handle post-test processing for scalability.

This document serves as a reference for frontend, backend, and device coordination for all test session interactions.