Industrial Wearable AI — Final Technical Stack (Full Specifications)

Document: Hardware and software specifications for the complete platform.
Version: 1.0
Last updated: 2025

Table of Contents

1. System Overview

```
┌───────────────────┐     BLE / WiFi      ┌────────────────────┐     HTTP/WS
┌───────────────────┐
│   Wearable Node   │  ──────────────────▶ │  Edge Gateway      │
──────────────────▶ │   Backend + DB    │
│   (ESP32+IMU+Temp)│                       │   (RPi / Laptop)    │
│   (API + ML)      │
└───────────────────┘                      └────────────────────┘
└─────────┬─────────┘

│

▼

┌───────────────────┐

│  Web Dashboard    │
│   (React)          │
└────────────────────┘
```

2. Hardware Specifications

2.1 Wearable Node (Per Worker)

Component	Specification	Details
Microcontroller	**ESP32-WROOM-32** (or ESP32-DevKitC)	Dual-core 240 MHz Xtensa LX6; 520 KB SRAM; 4 MB flash; 34 GPIO; BLE 4.2 + BR/EDR; WiFi 802.11 b/g/n; 3.3 V logic.
IMU (Motion)	**MPU6050** (6-axis)	Accelerometer ±2/±4/±8/±16 g; Gyroscope ±250/±500/±1000/±2000 °/s; I2C (0x68); 3.3–5 V; built-in 16-bit ADC. Typical: ±2 g accel, ±250 °/s gyro for wrist.
Temperature	**DHT11** or **DS18B20**	**DHT11:** 0–50 °C, ±2 °C, humidity 20–80% RH (optional). **DS18B20:** −55 to +125 °C, ±0.5 °C; 1-Wire. Choose one.
Power	**Li-ion 3.7 V** (e.g. 500–1000 mAh) + **TP4056**	TP4056: charge module with protection; 1S Li-ion; USB charge. Runtime: ~6–12 h at 25 Hz IMU + BLE (estimate).
Connectivity	**BLE 4.2** (primary), WiFi (optional)	BLE: GATT server; advertise or connect to gateway. WiFi: fallback for long range if gateway supports.
Enclosure	**Wrist band / strap**	Non-metallic strap; secure mount for PCB/module; dominant-wrist placement. Ventilation for temp sensor.
Optional (V1.1)	**Heart rate** (e.g. MAX30102 / PPG)	For fatigue index; add when fatigue model is prioritized.

Wearable Bill of Materials (BOM) — Indicative

Part	Model / Type	Qty	Est. unit cost (INR)
ESP32 dev board	ESP32-WROOM-32 DevKit / NodeMCU-32S	1	350–500
MPU6050	GY-521 breakout (I2C)	1	80–150
Temperature	DHT11 or DS18B20 + 4.7k (DS18B20)	1	50–100
Li-ion cell	3.7 V 500–1000 mAh	1	80–150
TP4056 module	With protection	1	30–50

Part	Model / Type	Qty	Est. unit cost (INR)
Wrist band / enclosure	Strap + case	1	100–200
Wires, connectors	Dupont / soldering	—	50–100
Total per wearable			**₹750–₹1,250**

2.2 Edge Gateway (Per Room / Line)

Component	Specification	Details
Option A — Raspberry Pi	**Raspberry Pi 4 Model B**	Quad-core 1.5 GHz ARM Cortex-A72; 2 GB or 4 GB RAM; 32 GB microSD (min); BLE onboard (or USB BLE dongle); 5 V 3 A PSU.
Option B — Laptop (dev/pilot)	x86_64 laptop	4+ GB RAM; USB BLE 4.0 dongle (e.g. CSR/Intel); used for development and small pilot.
Option C — Alternative SBC	e.g. Orange Pi, Rock Pi	2+ GB RAM; Linux; BLE via USB dongle if no onboard BLE.
Storage	32 GB+ (RPi: microSD)	OS + Python env + logs + optional local CSV buffer.
Connectivity	Ethernet or WiFi	To backend server (same network).
Power	5 V 3 A (RPi); UPS recommended	For 8+ hour shift reliability.

Edge BOM — Indicative

Part	Model / Type	Qty	Est. cost (INR)
Raspberry Pi 4	2 GB or 4 GB	1	3,500–4,500
microSD card	32 GB Class 10	1	300–500
USB BLE dongle (if needed)	BLE 4.0	1	200–400
Power supply	5 V 3 A USB-C	1	200–300
Case (optional)	RPi case	1	150–300
Total edge (RPi)			**₹4,350–₹6,000**

2.3 Backend Server (On-Prem or Cloud)

Component	Specification	Details
MVP / Pilot	Same machine as edge (RPi or laptop)	FastAPI + PostgreSQL on same host; minimal load.
Production (single factory)	Dedicated server or VM	2+ vCPU; 4+ GB RAM; 20+ GB SSD; Linux (Ubuntu 22.04 LTS or similar).
Database	PostgreSQL on same host or separate	2 GB RAM for Postgres; 10+ GB disk for 6–12 months of event data.

2.4 Network & Connectivity Summary

Link	Technology	Range / Notes
Wearable ↔ Edge	BLE 4.2 (GATT)	~10 m indoor; one wearable per connection or BLE central (multi-peripheral) on edge.
Edge ↔ Backend	HTTP/1.1, WebSocket	LAN; same subnet or routed.
Dashboard ↔ Backend	HTTPS, WebSocket	Browser; same network or via reverse proxy.

3. Software Specifications

3.1 Wearable Firmware (ESP32)

Item	Specification
IDE / Build	Arduino IDE 2.x **or** PlatformIO (VS Code); Arduino core for ESP32 **or** ESP-IDF v4.4+
Language	C++ (Arduino style or ESP-IDF)
Key libraries	**Adafruit_MPU6050** (or MPU6050_light / SparkFun); **DHT sensor library** (DHT11) or **OneWire** + **DallasTemperature** (DS18B20); **BluetoothSerial** (ESP32 Arduino) or **NimBLE** (BLE stack); **ArduinoJson** (v6) for JSON payload
Sampling	IMU: 25–50 Hz (configurable); Temperature: 0.2–1 Hz
Output	JSON over BLE (GATT characteristic write/notify) or UART; see Section 4 for payload.
Tasks	1) Read IMU; 2) Read temp; 3) Pack timestamp + worker_id + ax,ay,az,gx,gy,gz,temp; 4) Send via BLE. No ML on device.
Versioning	Semantic version (e.g. 1.0.0); configurable worker_id (e.g. stored in NVS or compile-time).

Firmware dependency summary

Library	Purpose	Version (indicative)
Adafruit_MPU6050	IMU read, calibration	2.x
DHT sensor library	DHT11	1.4.x
ArduinoJson	JSON serialize	6.x
NimBLE or BluetoothSerial	BLE	—

3.2 Edge Gateway Software

Item	Specification
OS	Raspberry Pi OS (64-bit) **or** Ubuntu 22.04 LTS
Runtime	Python **3.10** or **3.11** (recommended)
BLE stack	**bleak** (async BLE, cross-platform) **or** pybluez (Linux); prefer bleak for GATT client.
Core dependencies	**numpy** ≥1.24; **pandas** ≥2.0; **scikit-learn** ≥1.2; **joblib** (model load); **requests** or **aiohttp** (HTTP); **websockets** (optional, to backend); **python-dotenv** (config)
Process pipeline	1) Connect to wearable(s), read BLE stream → 2) Buffer raw samples → 3) Noise filter (e.g. low-pass 5–10 Hz) → 4) Sliding window (e.g. 2–5 s, 50% overlap) → 5) Feature extraction (mean, std, min, max, zero-crossing, etc.) → 6) Load activity model (joblib), predict label → 7) Rule-based ergo/fatigue (optional) → 8) POST /events to backend (batch or stream) or WebSocket push.
Config	Backend URL, API key, window size, model path, BLE device ID(s).
Logging	File + console; log level configurable; rotate logs.

Edge Python stack (versions)

Package	Min version	Purpose
Python	3.10	Runtime
numpy	1.24	Arrays, filtering
pandas	2.0	DataFrames, CSV I/O
scikit-learn	1.2	Model load, inference
joblib	1.2	Serialized model load
bleak	0.21+	BLE client (async)
aiohttp	3.8+	Async HTTP client

Package	Min version	Purpose
python-dotenv	1.0+	Env config

3.3 Backend API & Services

Item	Specification
Framework	**FastAPI 0.100+**
Runtime	Python **3.10** or **3.11**
ASGI server	**Uvicorn** (0.22+); workers: 1–2 for MVP.
Database	**PostgreSQL 14** or **15**; driver: **asyncpg** or **SQLAlchemy 2.x** (async).
ORM / queries	SQLAlchemy 2.x (async) **or** raw asyncpg; migrations: **Alembic**.
Auth	**JWT** (access token) **or** API key for edge; simple login for dashboard (username + password → JWT). **python-jose** or **PyJWT**; **passlib** + **bcrypt** for password hash.
Realtime	**WebSocket** endpoint for dashboard (e.g. `/ws/live`); broadcast activity updates. **Optional:** Redis Pub/Sub for multi-instance.
Endpoints (minimal)	`POST /api/events` (batch activity events); `GET /api/workers`; `GET /api/sessions`; `GET /api/sessions/{id}/summary`; `GET /api/live` or WebSocket for live state; `POST /api/auth/login`.
CORS	Allow dashboard origin; credentials if cookie/session.

Backend dependency summary

Package	Min version	Purpose
fastapi	0.100+	API
uvicorn	0.22+	ASGI server
sqlalchemy	2.0+	ORM (async)
asyncpg	0.28+	PostgreSQL async driver
alembic	1.12+	Migrations
pydantic	2.0+	Validation (via FastAPI)
python-jose / PyJWT	—	JWT
passlib[bcrypt]	—	Password hashing

Package	Min version	Purpose
websockets	11+	WebSocket (if used)
python-dotenv	1.0+	Config

3.4 Database Schema (PostgreSQL)

Table	Columns (key)	Purpose
workers	id (PK), name, role, device_id, created_at	Worker master.
sessions	id (PK), worker_id (FK), started_at, ended_at, shift_label	Shift/session.
activity_events	id (PK), session_id (FK), ts, label (enum: sewing, idle, adjusting, error, break), risk_ergo (bool), risk_fatigue (bool)	Time-series of labels.
session_aggregates	session_id (PK, FK), active_pct, idle_pct, adjusting_pct, error_pct, alert_count, updated_at	Per-session summary.
devices	id (PK), hardware_id, worker_id (nullable), last_seen_at	Wearable devices.

Indexes: `activity_events(session_id, ts)` ; `sessions(worker_id, started_at)` .

3.5 ML Pipeline (Training & Inference)

Item	Specification
Training env	Python 3.10+; **pandas**, **numpy**, **scikit-learn** 1.2+, **xgboost** 1.7+ (optional).
Input	Labeled CSV: timestamp, ax, ay, az, gx, gy, gz, temp, label.
Preprocessing	Sliding windows (e.g. 2–5 s, 50% overlap); feature set: per-axis mean, std, min, max, zero-crossing rate; optional: magnitude, correlation between axes.
Model	**RandomForestClassifier** or **GradientBoostingClassifier** (sklearn) **or XGBClassifier**; class labels: Sewing, Idle, Adjusting, Error, Break.
Export	**joblib** dump (`.joblib` or `.pkl`); loaded on edge at startup.
Evaluation	Hold-out 20%; metrics: accuracy, per-class precision/recall/F1; target accuracy ≥85%.
Inference (edge)	Same feature extraction + `model.predict(feature_vector)` ; latency target <100 ms per window.
Ergo / Fatigue	Rule-based in edge or backend: e.g. idle > 2 min → idle_alert; sustained high gyro → ergo_risk; high temp + high idle % → fatigue_risk.

ML dependency summary

Package	Purpose
scikit-learn	RF/GBM, preprocessing, metrics
xgboost	Optional XGBoost classifier
joblib	Model serialize/load
pandas, numpy	Data and features

3.6 Supervisor Dashboard (Frontend)

Item	Specification
Framework	**React 18.x**
Build	**Vite** 4+ **or** Create React App
Language	TypeScript **5.x** (recommended) or JavaScript
Charts	**Recharts** 2.x **or Chart.js** 3.x + react-chartjs-2
HTTP client	**fetch** or **axios**
Realtime	**WebSocket** (native or library) to backend `/ws/live` ; fallback: polling every 2–5 s.
State	**React Context** or **Zustand** (lightweight) for user, workers, live state, alerts.
UI components	**Material-UI (MUI)** 5.x **or Chakra UI or** plain CSS/Tailwind; table/cards for workers; badges for state/risk.
Auth	Login form → POST /api/auth/login → store JWT (memory or httpOnly cookie); Authorization header on API calls.
Routing	**React Router** 6.x: `/` (live), `/sessions` , `/workers` , `/login` .
Alerts / toasts	**react-hot-toast** or **notistack** or MUI Snackbar for real-time alerts.

Dashboard dependency summary

Package	Purpose
react	18.x
react-dom	18.x
react-router-dom	6.x
recharts or chart.js	Charts

Package	Purpose
@mui/material or chakra-ui	UI (optional)
axios or fetch	API
zustand or react context	State
react-hot-toast / notistack	Toasts
typescript	5.x (if TS)

Views (minimum)

- **Live:** List/cards of workers with current state (Active / Idle / Adjusting / Error / Break) and risk badges (Ergo / Fatigue).
- **Alerts:** List of recent alerts with time and worker.
- **Shift summary:** Table or bars of active % per worker for selected session/date.
- **Trends (optional):** Line chart of active % over days.

3.7 DevOps & Environment

Item	Specification
Version control	**Git**; repo layout: `firmware/`, `edge/`, `backend/`, `dashboard/`, `ml/`, `docs/`.
Python env	**venv** or **poetry** or **pipenv**; `requirements.txt` per service (edge, backend, ml).
Containers (optional)	**Docker**; Dockerfile for backend + dashboard (e.g. nginx for SPA); docker-compose for backend + PostgreSQL + Redis (optional).
CI (optional)	GitHub Actions / GitLab CI: lint (ruff/flake8, ESLint), tests (pytest, Jest), build frontend.
Secrets	Env vars (`.env`); no secrets in repo; API keys and DB URL in env.

---

4. Data Formats & Protocols

4.1 Wearable → Edge (BLE / JSON)

Payload (per sample or batched, e.g. every 100 ms):

```
{
  "worker_id": "W01",
  "ts": 1704067200123,
  "ax": -0.3, "ay": 1.2, "az": 9.6,
  "gx": 21, "gy": -4, "gz": 3,
  "temp": 31.5
}
```

- **worker_id:** string (assigned to device).
- **ts:** Unix milliseconds (optional if edge stamps on receive).
- **ax, ay, az:** acceleration (g); **gx, gy, gz:** gyro (°/s); **temp:** °C.
- **BLE:** GATT characteristic write or notify; JSON string in UTF-8.

4.2 Edge → Backend (HTTP POST /events)

Request body (batch):

```
{
  "device_id": "ESP32_ABC123",
  "worker_id": "W01",
  "events": [
    { "ts": 1704067200123, "label": "sewing", "risk_ergo": false, "risk_f
    { "ts": 1704067202123, "label": "idle", "risk_ergo": false, "risk_fat
  ]
}
```

- **label:** one of `sewing`, `idle`, `adjusting`, `error`, `break`.
- **ts:** Unix ms; **risk_ergo**, **risk_fatigue:** boolean.

4.3 Backend → Dashboard (WebSocket / REST)

WebSocket message (live state):

```
{
  "worker_id": "W01",
  "name": "Operator 1",
  "current_state": "idle",
  "risk_ergo": false,
  "risk_fatigue": true,
  "updated_at": 1704067202123
}
```

REST GET /api/sessions/{id}/summary:

```
{
  "session_id": "uuid",
  "worker_id": "W01",
  "active_pct": 72.5,
  "idle_pct": 18.2,
  "adjusting_pct": 6.1,
  "error_pct": 2.0,
  "alert_count": 1
}
```

5. Environment & Tooling

Role	Tool	Version / notes
Firmware	Arduino IDE or PlatformIO	Latest stable
Firmware	ESP32 board support	Arduino core 2.0.x or ESP-IDF 4.4+
Edge / Backend / ML	Python	3.10 or 3.11
Backend DB	PostgreSQL	14 or 15
Frontend	Node.js	18 LTS or 20 LTS (for npm/Vite)
Frontend	npm or yarn or pnpm	Latest
API testing	Postman or curl / httpx	—
BLE testing	nRF Connect (mobile) or bleak CLI	—

6. Cost Summary

Category	Item	Qty	Est. cost (INR)
Wearable	ESP32 + MPU6050 + temp + power + band	1	750–1,250
Wearables (pilot)	Same	3–5	2,250–6,250
Edge	Raspberry Pi 4 + SD + PSU + BLE dongle	1	4,350–6,000
Backend (pilot)	Use same RPi or laptop	0	0
Backend (production)	Small server/VM or existing PC	1	0–5,000
Software	All open-source	—	0

Category	Item	Qty	Est. cost (INR)
Contingency	10–15%	—	700–1,500
Total (1 wearable + edge)			**₹5,800–₹8,750**
Total (5 wearables + edge)			**₹8,300–₹13,500**

This document is the single source of truth for hardware and software stack specifications. Update versions and part numbers as the project evolves.