VIT Vellore Institute of Technology (Deemed to be University under section 3 of UGC Act, 1956)	**Invention Disclosure Format (IDF)-B**	Document No.	02-IPR-R003
		Issue No/Date	2/04.09.2025
		Amd. No/Date	0/00.00.0000

1. **Title of the invention:** Recon: Wearable Device for Disease Detection Using Vitals and Breath Analysis (E-Nose) with Hybrid AI Framework

2. **Field /Area of invention:**
This invention relates broadly to the fields of biomedical sensing, wearable electronics, respiratory monitoring, electronic nose (e-nose) technology, Internet of Things (IoT), and edge/ cloud-based machine learning. It specifically targets healthcare applications involving non-invasive, continuous disease detection by integrating physiological monitoring and breath analysis, supported by secured telemedicine workflows.

3. **Prior Patents and Publications from literature (provide a table summarizing the prior art)**

Year	Patent ID	Title	Key points
2020	US 11,045,111 B1	Real-Time Breath Analyzer for Detecting Disease	1. Discloses breath analysis for disease detection using breath biomarkers and sensor(s), but not as a continuous wearable with multimodal vitals fusion and on-edge ML risk scoring. 2. Emphasizes diagnostic workflows rather than a unified wearable E-nose plus vitals system with patient/doctor dashboards and longitudinal personalization. 3. Does not detail hybrid MOX/NDIR/QCM arrays with on-body drift/temperature compensation in a single platform.
2020	CN 112394172 A	Acetone Monitoring Device for Exhaled Gas of Diabetic Patients	1.Targets acetone (diabetes marker) monitoring with portable/wearable form but focuses on a single biomarker rather than a heterogeneous E-nose array fused with vitals. 2. Lacks combined CO_2 (NDIR), ammonia, and ethanol/VOC channels and does not implement a 1–10 edge risk index with cloud-assisted learning. 3. Does not integrate clinician workflow or adaptive patient-specific baselines across time. .

2020–2022	US 2022/0240808 A1	Disposable Wearable Sensor for Continuous Monitoring of Breath Biochemistry	1. Describes electrochemical wearable sensing of single/multiple analytes for breath, but not a MOX-NDIR-QCM hybrid with vitals fusion in an edge ML pipeline on Raspberry Pi. 2. Limited coverage of drift correction, on-edge feature fusion, and secure telehealth prescription workflow. 3. Does not specify commodity parts aligned with your architecture (MQ-135/137, MH-Z19, MiCS-5524) within a coherent sensor-fusion system.
2007	US 2007/0062255 A1	Apparatus for collecting and analyzing human breath	1. Early handheld breath collection and analysis device for VOCs. 2. Not designed as a wearable continuous E-nose. 3. No edge ML, vitals fusion, or clinician workflow integration
2014	US 2015/0177224 A1	Breath acetones monitor and method of detecting acetone in human breath	1. Portable acetones monitor for noninvasive metabolic assessment. 2. Single-marker focus; lacks heterogeneous sensor array and vitals fusion. 3. No continuous wearable pipeline or cloud-assisted personalization
2017	BR 112018070768 A8	Breath analysis device	1. Discloses a breath analysis device for VOC detection. 2. Does not specify hybrid MOX/NDIR/QCM in a wearable form with drift/temperature compensation. 3. No integrated clinician dashboard or on-edge risk scoring

Non – Patent Literature:

Year	Citation	Title	Key points
2024	Kim, D., Lee, J., Park, M.K. *et al.* Recent developments in wearable breath sensors for healthcare monitoring. *Commun Mater* **5**, 41 (2024). https://doi.org/10.1038/s43246-024-00480-w	Recent developments in wearable breath sensors for healthcare monitoring	1) Reviews wearable breath sensors and biomarkers, emphasizing integration with IoT/ML and challenges (selectivity, humidity, power) addressed by hybrid arrays and compensation. 2) Supports the need for real-time on-body systems consistent with an edge-ML architecture
2024	Dang, Yu & Reddy, Yenugu & Cheffena, Michael. (2024). Facile E-nose based on single antenna and graphene oxide for sensing volatile organic compound gases with ultrahigh selectivity and accuracy. Sensors and Actuators B Chemical. 419. 10.1016/j.snb.2024.136409	Facile E-nose based on single antenna and graphene oxide for sensing volatile organic compound gases with ultrahigh selectivity and accuracy	1) Demonstrates compact e-nose performance comparable to arrays, highlighting materials/architecture trends for portable devices. 2) Underscores need for robust pattern recognition and compensation, aligning with wearable fusion and ML pipelines
2024	Yin, Z., Yang, Y., Hu, C. *et al.* Wearable respiratory sensors for health monitoring. *NPG Asia Mater* **16**, 8 (2024). https://doi.org/10.1038/s41427-023-00513-9	Wearable respiratory sensors for health monitoring	1) Reviews wearable gas/respiratory sensors, including paper-based and flexible platforms; notes feasibility and challenges of on-body gas sensing. 2) Highlights environmental and drift issues, motivating hybrid arrays and calibration strategies used in a medical wearable

Summary and Background of the Invention (Address the gap / Novelty)

Background:

The rapid increase in chronic disease prevalence worldwide, combined with rising healthcare costs and limitations of traditional reactive care models, has catalysed a global shift toward continuous, preventive, and patient-cantered healthcare solutions. Current wearable physiological sensors and IoT-enabled health monitoring devices provide access to vital parameters such as heart rate, blood pressure, oxygen saturation, and body temperature in real time. Despite their availability, these devices predominantly rely on single-modality sensing and lack integration with biochemical markers, limiting their diagnostic utility for early-stage disease detection.

Electronic nose (e-nose) technology has demonstrated great promise by identifying volatile organic compounds (VOCs) in exhaled breath, which can serve as non-invasive biomarkers indicative of early metabolic, renal, hepatic, respiratory, cardiac, and infectious diseases. However, existing e-nose systems are largely confined to research laboratories or single-disease diagnostic tools without wearable, real-time, or remote monitoring capabilities. This results in a disconnect between the potential of breath-based diagnostics and practical, continuous health monitoring in real-world settings.

Additional clinical and technical challenges still limit the utility of wearable health systems:

- **Limited Machine Learning Integration:** Most current wearables depend on simple threshold-based alerts rather than sophisticated, real-time multi-parameter disease risk scoring using advanced machine learning or deep learning methods.
- **Fragmented Data and Clinical Workflow:** Physiological and biochemical data are often siloed; systems lack unified fusion and analysis of these data streams, complicating telemedicine integration and reducing predictive accuracy.
- **Scalability, Privacy, and Security Concerns:** Existing solutions often lack robust data security, privacy protection, and compliance with healthcare regulations when managing sensitive patient data across cloud platforms and multiple user roles.
- **Data Scarcity for AI Models:** Machine learning in biomedical domains suffers from limited access to large, clinically-diverse datasets, especially for rare or early-stage conditions. This restricts generalization and necessitates synthetic data generation.

Technical Gaps in Prior Art

- **Limited Sensor Modalities:** Many commercial and academic wearable devices focus solely on physiological parameters and do not incorporate e-nose or VOC breath analysis sensors.
- **Lack of Miniaturized and Integrated E-Nose:** Standalone e-nose devices are typically non-wearable bench-top setups without integrated physiological monitoring or on-device real-time AI inference.
- **Suboptimal Cloud and UI Capabilities:** Most IoT health platforms provide delayed or summary-level information with poor support for dynamic clinical workflows such as programmable alerts, appointment scheduling, or patient-provider interaction.
- **Underutilized Machine Learning:** Multisensory fusion and event-driven personalized alerting are rare, reducing the potential for early intervention and clinical decision support.
- **Incomplete Privacy and Regulatory Compliance:** Many existing efforts inadequately address data ownership, privacy safeguards, and regulatory pathways necessary for clinical adoption.

Novelty and Technical Advancement

The disclosed invention presents a **comprehensive, multimodal wearable health monitoring system** uniquely combining the following advances:

- **Integrated Sensing Platform:** Combines standardized vital sign sensors (ECG, SpO_2, temperature, blood pressure) with advanced hybrid e-nose arrays (metal oxide MOX sensors, non-dispersive infrared NDIR sensors, quartz crystal microbalance QCM sensors, and chemocapacitive sensors), enabling dense, real-time acquisition of physiological and biochemical biomarkers in naturalistic settings.
- **Edge and Cloud-Based Machine Learning:** Implements efficient real-time inference via embedded platforms (Raspberry Pi 4) using diverse algorithms including Random Forests, Support Vector Machines, Deep Neural Networks with attention mechanisms, and LSTM recurrent networks for temporal pattern detection. Cloud-based models augment device inference by supporting aggregate data analytics and personalized model refinement.

- **Synthetic Dataset Generation:** Employs state-of-the-art data augmentation and synthetic data approaches, allowing high-quality AI model training despite restricted access to large-scale clinical datasets, improving model robustness, privacy, and generalization, especially for rare diseases.
- **Role-Based Telehealth Platform:** Provides a secure, user-friendly web interface tailored for patients, healthcare providers, and administrators with role-specific access controls, enabling real-time data visualization, historical trend analysis, alert management, remote prescription, and appointment scheduling.
- **Event-Tiered Alert Protocols:** Utilizes dynamic thresholds personalized to patient baselines and medical history to minimize false positives while ensuring rapid critical condition notification and prompt clinical response.
- **Scalability, Security, and Usability:** Designed for scalable cloud deployment, high wearability with optimized battery life, and universal accessibility in clinical and homecare environments, all built with privacy, security, and healthcare compliance in mind.
- **Regulatory-Centric Architecture:** Prepared for clinical translation via pathways incorporating device calibration, user acceptability, clinical validation, and adherence to medical device regulations and standards.

Objectives of the Invention

The principal objectives of the invention are to:

1. **Deliver seamless, continuous, non-invasive monitoring** of multiple vital signs and disease-specific breath biomarkers within a single wearable platform, enabling early and actionable detection of chronic diseases including diabetes, chronic kidney disease, hepatic disorders, respiratory ailments, cardiac abnormalities, and infections.
2. **Empower patients and healthcare providers** via a unified, interactive web-based dashboard that supports real-time analytics, historical data visualization, alert generation, prescription issuance, appointment handling, and administrative management.
3. **Achieve diagnostic accuracy greater than 90%** across multiple disease categories by employing advanced, explainable machine learning models both at the edge and in cloud environments with dynamic feature selection optimized for multimodal input.
4. **Ensure strict data privacy and security** using secure authentication and robust role-based access control mechanisms that comply with applicable healthcare data regulations (e.g., HIPAA, GDPR).
5. **Leverage synthetic data generation frameworks** to supplement limited clinical datasets, thus enabling accelerated development, benchmarking, and validation of AI models suitable for diverse patient populations including rare diseases.
6. **Design for scalability and reliability** by integrating cloud data lakes, secure wireless connectivity, and modular hardware/software architectures that accommodate future sensor expansions and analytic algorithm upgrades.
7. **Facilitate smooth clinical translation and regulatory compliance** by defining comprehensive user calibration protocols, conducting thorough usability testing, and planning extensive clinical trials aligned with international medical device standards (e.g., ISO 13485, FDA).

6. Working Principle:

1. Data Acquisition Layer (Wearable + Sensors)

- **Patient Identification:** The process begins by generating a unique patient ID for every user, ensuring that all subsequent data is securely linked and managed.
- **Sensor Data Collection:** Wearable sensors continuously collect vital physiological data (such as heart rate, blood pressure, SpO_2, temperature) alongside breath analysis data (VOCs and related biomarkers).

2. Cloud Data Management

- **Secure Transmission:** All collected data is encrypted and securely uploaded to the cloud, protecting patient privacy.
- **Longitudinal Storage:** Each patient's health records are systematically stored over time, enabling longitudinal tracking of health status and trends.
- **Data Synchronization:** The system automatically synchronizes health data across all users and authorized clinicians, ensuring that real-time updates are available for analysis and consultation.

3. Web App User Management

- **Role-Based Dashboards:** The platform provides dedicated dashboards for patients, doctors, and administrators. Each dashboard is tailored to user needs—patients view personal health histories and predictions, doctors review patient lists and results, and admins manage users and maintain data integrity.

4. AI and ML Prediction Layer

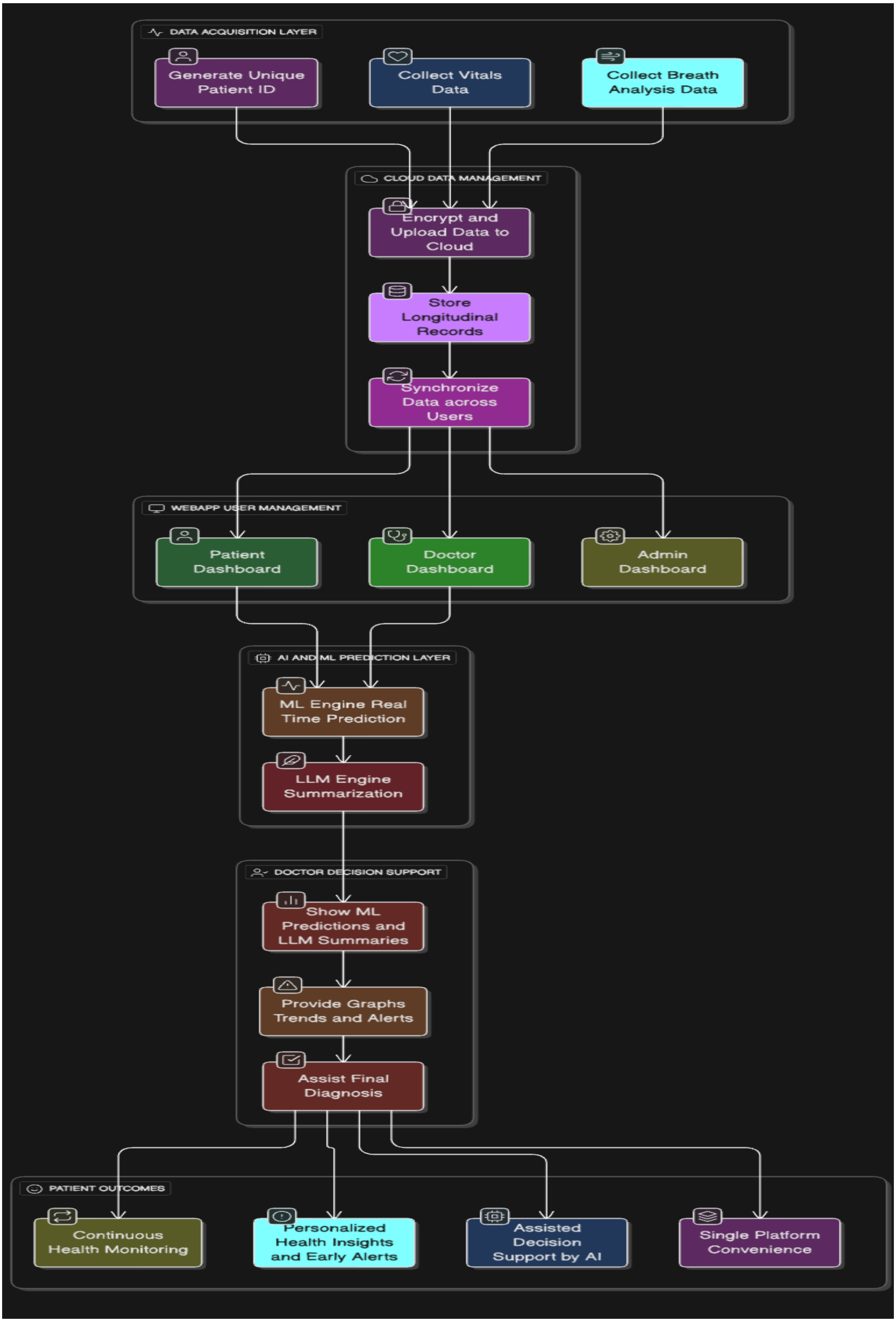
- **Machine Learning Engine:** Real-time data is processed by a machine learning engine, which predicts disease risks and flags significant health changes.
- **LLM Summarization:** A large language model (LLM) engine further processes historical and new data, summarizing trends, and synthesizing insights for both patients and providers.

5. Doctor Decision Support

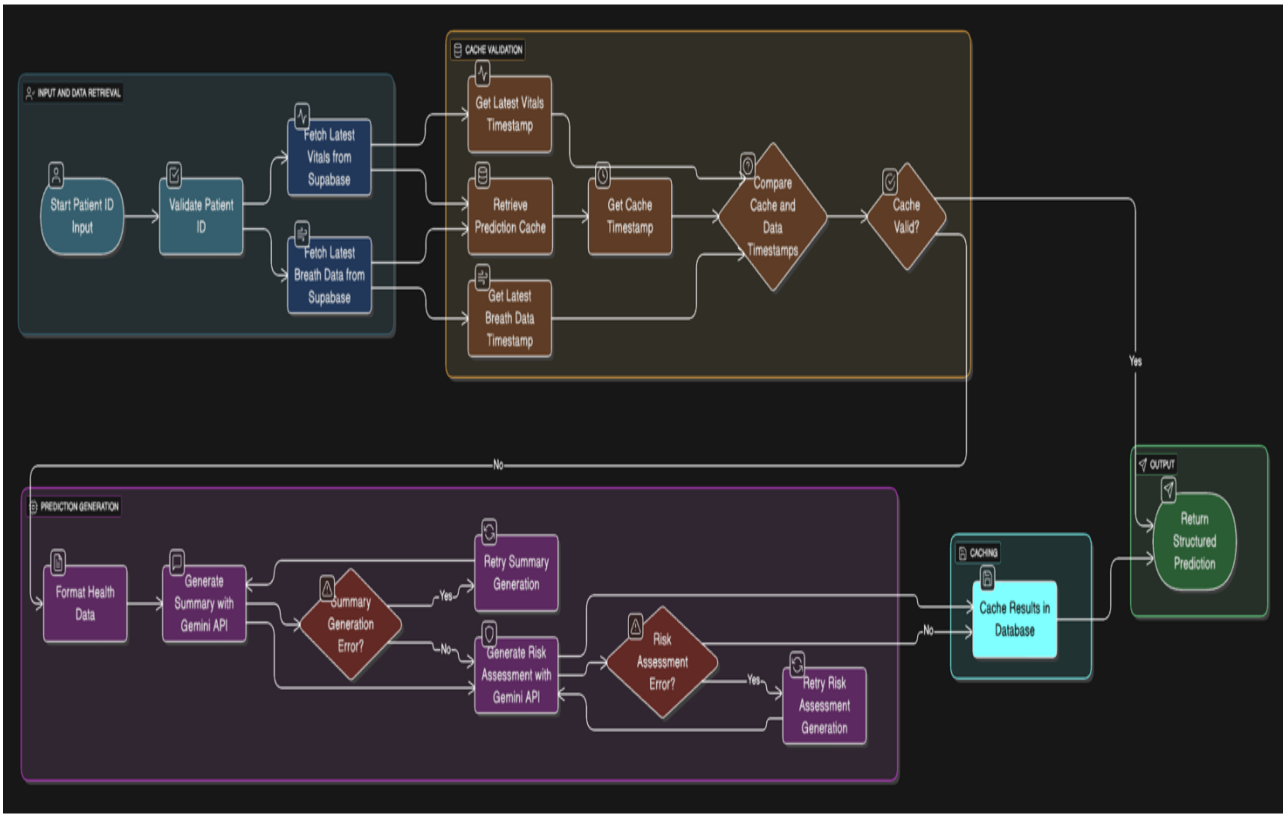
- **Insightful Presentation:** The system presents ML predictions and LLM summaries in an actionable format during clinical consultations.
- **Visualization:** It generates graphs, trends, and alert visualizations to assist doctors in quickly interpreting health trajectories.
- **Final Diagnosis:** The combination of real-time analytics and comprehensive summaries helps clinicians make informed, timely, and accurate diagnoses.

6. Patient Outcomes

- **Continuous Health Monitoring:** Patients benefit from ongoing, non-invasive health surveillance.
- **Personalized Insights and Early Alerts:** Automated personalized health feedback and early warnings improve patient engagement and prevention.
- **AI-Assisted Decision Making:** Doctors leverage AI-driven support for rapid and reliable decision-making.
- **Single Platform Convenience:** All processes—data collection, analysis, consultation, alerts, and historical review—are managed within a unified digital platform.


DATA ACQUISITION LAYER
Generate Unique Patient ID
Collect Vitals Data
Collect Breath Analysis Data
CLOUD DATA MANAGEMENT
Encrypt and Upload Data to Cloud
Store Longitudinal Records
Synchronize Data across Users
WEBAPP USER MANAGEMENT
Patient Dashboard
Doctor Dashboard
Admin Dashboard
AI AND ML PREDICTION LAYER
ML Engine Real Time Prediction
LLM Engine Summarization
DOCTOR DECISION SUPPORT
Show ML Predictions and LLM Summaries
Provide Graphs Trends and Alerts
Assist Final Diagnosis
PATIENT OUTCOMES
Continuous Health Monitoring
Personalized Health Insights and Early Alerts
Assisted Decision Support by AI
Single Platform Convenience

AI Prediction Generation Workflow:



1. Input and Data Retrieval

- **Start Patient ID Input:**
  The workflow initiates when a patient ID is entered.
- **Validate Patient ID:**
  The system validates the ID to ensure the user exists and is authorized.
- **Fetch Latest Health Data:**
  - The most recent vitals and breath analysis data are retrieved from Supabase (the cloud database).

2. Cache Validation

- **Get Timestamps:**
  Timestamps for the latest vitals data, breath data, and any previous prediction cache are fetched.
- **Validation Logic:**
  - The system compares the cache timestamp with the newest data timestamps.
  - If the data in cache is still valid (i.e., no new data has been added since the last prediction), the cached results are used.
  - If not, the workflow proceeds to generate a new prediction.

3. Prediction Generation (if cache is outdated or missing)

- **Format Health Data:**
  The incoming sensor data is formatted into a structure suitable for AI analysis.
- **Generate Summary with Gemini API:**
  The system calls the Gemini API to create a summary of the patient's health state.
- **Summary Error Handling:**
  - If summary generation fails, the process is retried until success or timeout.
- **Generate Risk Assessment with Gemini API:**
  On summary success, the next Gemini API call assesses patient disease risks.
- **Risk Assessment Error Handling:**
  - If an error occurs in risk assessment, the workflow retries the process.

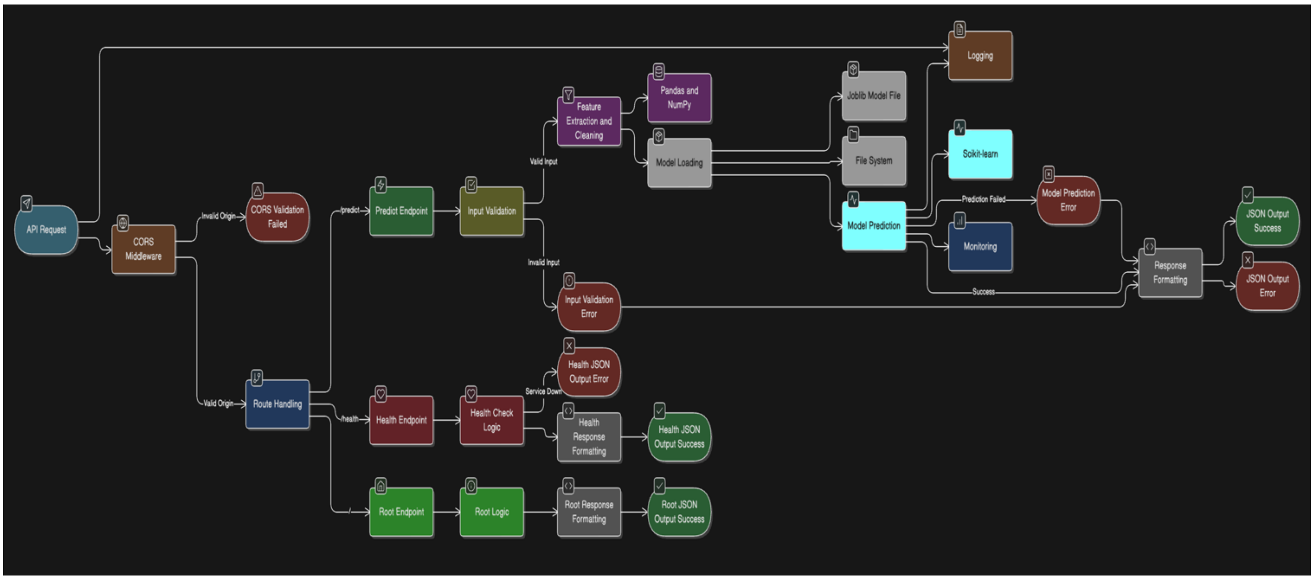
4. Caching

- **Cache Results in Database:**
  Once valid prediction and risk assessment results are generated, they are cached in the database for reuse if new data hasn't arrived.

5. Output

- **Return Structured Prediction:**
  The final structured prediction (encompassing summary and risk scores) is returned for display or further clinical use.

Fast API-ML-Service-Workflow:



1) Request ingress and security

- An external client sends an API Request, which first passes through CORS Middleware to validate origin; invalid origins are rejected with a CORS Validation Failed outcome, otherwise the request proceeds to Route Handling for the matched path (e.g., /predict, /health, /).

2) Routing and endpoints

- For /predict, the Predict Endpoint receives a JSON payload; the service also exposes a Health Endpoint (/health) and a Root Endpoint (/) with simple Root Logic and formatting to confirm liveness/readiness.

3) Input validation and preprocessing

- Input Validation checks schema, types, ranges, and required fields. Validation errors are surfaced as Input Validation Error; otherwise the pipeline triggers Feature Extraction and Cleaning implemented with Pandas and NumPy to coerce, impute, scale/encode, and assemble the model-ready feature vector.

4) Model loading and prediction

- At startup or on first call, Model Loading retrieves a serialized scikit-learn artifact (via Joblib Model File from the File System). The loaded estimator is used for Model Prediction; failures are reported as Model Prediction Error, while successful paths feed Monitoring hooks.

5) Monitoring and logging

- Logging captures request/response metadata, timing, and errors; Monitoring records prediction events and model status for drift/downtime dashboards and alerts. This is standard practice to track model performance, latency, and failures in production ML services.

6) Response formatting and output

- On success, Response Formatting constructs a Health/Root/Prediction JSON payload. If formatting fails, a JSON Output Error is returned; otherwise, clients receive JSON Output Success with the predicted class/score and any auxiliary fields (e.g., confidence, version).

7) Health checks and service diagnostics

- The Health Endpoint runs Health Check Logic (e.g., model-in-memory, filesystem and joblib availability, dependency checks) and returns formatted JSON. This enables orchestrators and load balancers to gate traffic only when the model and dependencies are ready.

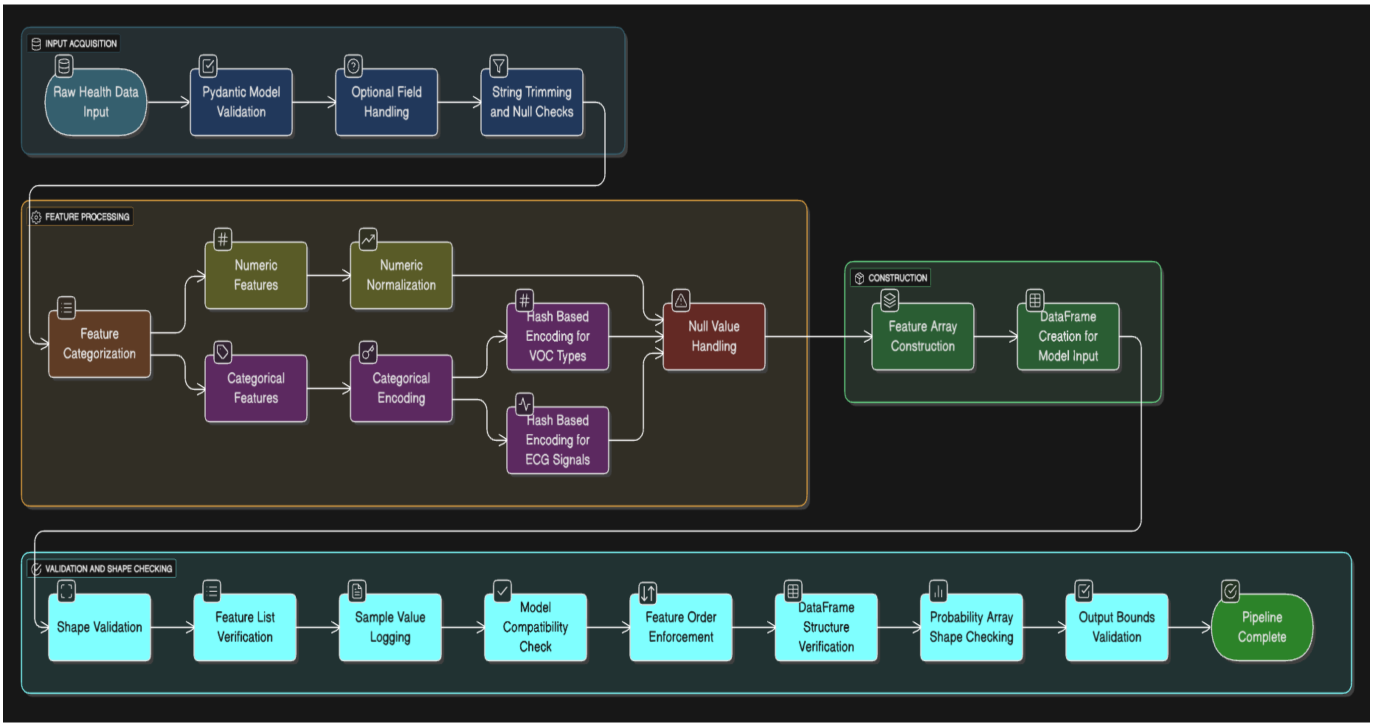
8) Failure handling and graceful degradation

- Distinct error nodes isolate issues:
    - CORS Validation Failed for disallowed origins.
    - Input Validation Error for malformed/invalid payloads.
    - Health JSON Output Error and Root JSON Output Error for formatting failures.
    - Model Prediction Error when the estimator or features fail. These explicit branches support targeted logging, alerting, and user-facing error codes.

9) Operational best practices reflected

- The workflow mirrors common production ML API guidance:
    - Serialize models with joblib, load once, and reuse.
    - Use FastAPI dependency/lifespan to ensure models are ready before serving.
    - Validate inputs strictly and normalize features with pandas/numpy.
    - Add logging/monitoring for prediction capture and drift tracking.
    - Provide health endpoints for readiness/liveness checks

Health-data-Feature-Engineering-Pipeline:



Input acquisition

- Raw Health Data Input → Pydantic Model Validation: Incoming JSON/payload is schema-validated (types, ranges, required fields) using a strict model, preventing malformed records from reaching the model stage.
- Optional Field Handling → String Trimming and Null Checks: Optional fields are defaulted, strings are stripped, and obvious nulls are flagged early to reduce downstream exceptions during encoding/normalization.

Feature processing

- Feature Categorization: Split variables into numeric (e.g., HR, SpO_2, temperature, BP) and categorical (e.g., device type, sampling mode) branches for appropriate transforms.
- Numeric Features → Numeric Normalization: Apply scaling/normalization to stabilize training/inference and align ranges across sensors (e.g., z-score/min–max).
- Categorical Features → Categorical Encoding: Encode nominal/ordinal attributes via one-hot/ordinal/target/frequency/binary encoders depending on cardinality and model needs.
- Hash-Based Encoding for VOC Types and ECG Signals: High-cardinality channel labels (VOC species or ECG segment/event IDs) can be hashed into a fixed-size representation to control dimensionality and handle unseen categories efficiently at inference time.
- Null Value Handling: Centralized imputation or masking after all per-branch transforms ensures consistent treatment of missing values before concatenation, preventing silent NaN propagation into the estimator.

Construction

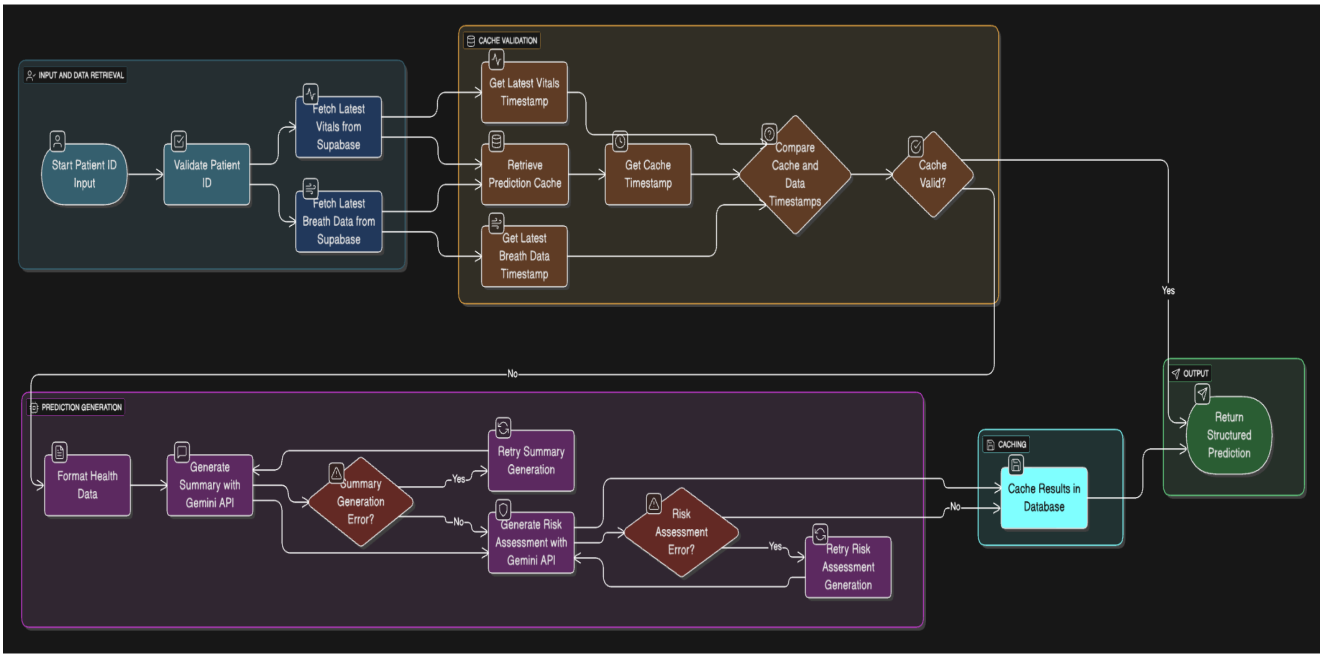
- Feature Array Construction → DataFrame Creation for Model Input: Concatenate numeric-normalized, categorical-encoded, and hashed vectors into a single ordered array/DataFrame that matches the model's training feature contract.

Validation and shape checking

- Shape Validation → Feature List Verification: Confirm expected feature count and names (contract checks) to prevent "column mismatch" failures in production.

- Sample Value Logging: Log a small sample of transformed rows for observability and debugging (e.g., drift, bad encodings).
- Model Compatibility Check → Feature Order Enforcement: Enforce deterministic column order that the serialized estimator expects (common with scikit-learn/joblib artifacts).
- DataFrame Structure Verification: Ensure dtypes and index alignment are correct prior to prediction to avoid runtime errors.
- Probability Array Shape Checking: After model inference, validate output dimensions (e.g., K-class softmax length) before response formatting to catch model/contract regressions.
- Output Bounds Validation: Sanity-check scores in and ensure they sum to ≤1 (classification) or meet expected ranges (regression), guarding against corrupted artifacts or overflow.
- Pipeline Complete: Emit a structured, validated payload to the next stage (e.g., API response or downstream alerting).

AI-Prediction-Generation-Workflow:



Flow overview

- The pipeline starts with a patient-scoped request, validates identity, fetches the latest vitals and breath data, checks whether a previous prediction is still valid via timestamps, and either serves a cached result or regenerates a new prediction with LLM calls. This pattern minimizes latency and cost while preserving freshness through timestamp comparison and cache TTLs.

1) Input and retrieval

- Start Patient ID Input → Validate Patient ID: Ensures the request references a known patient before any downstream work is done, reducing wasted compute on invalid IDs.
- Fetch Latest Vitals from Supabase and Fetch Latest Breath Data from Supabase: Pulls the most recent measurements for a consistent prediction snapshot tied to the same request time.

2) Cache validation (stale-or-fresh decision)

- Retrieve Prediction Cache + Get Cache Timestamp and compare with Get Latest Vitals Timestamp and Get Latest Breath Data Timestamp. If Cache Valid? is "Yes," return the prior result; otherwise proceed to prediction generation. Timestamp-driven invalidation is a simple, reliable approach to avoid recomputing when measurements haven't changed while assuring fresh output when they have.

3) Prediction generation with LLM and robust retries

- Format Health Data: Build a structured, normalized feature and context representation to prompt the LLM consistently.
- Generate Summary with Gemini API → on Summary Generation Error, Retry Summary Generation with bounded retries and backoff to withstand transient faults or rate limits—standard best practice in LLM integrations.
- Generate Risk Assessment with Gemini API → on Risk Assessment Error, Retry Risk Assessment Generation using the same retry/backoff policy. Explicit retry paths around LLM calls improve resilience to temporary errors (timeouts, 429s) and stabilize user experience.

Notes on LLM caching:

- Beyond application-level result caching, model-side context caching (implicit/explicit) can cut cost and latency for repeated prompts or shared long context, subject to the provider's TTL and cache semantics (e.g., Gemini context caching).

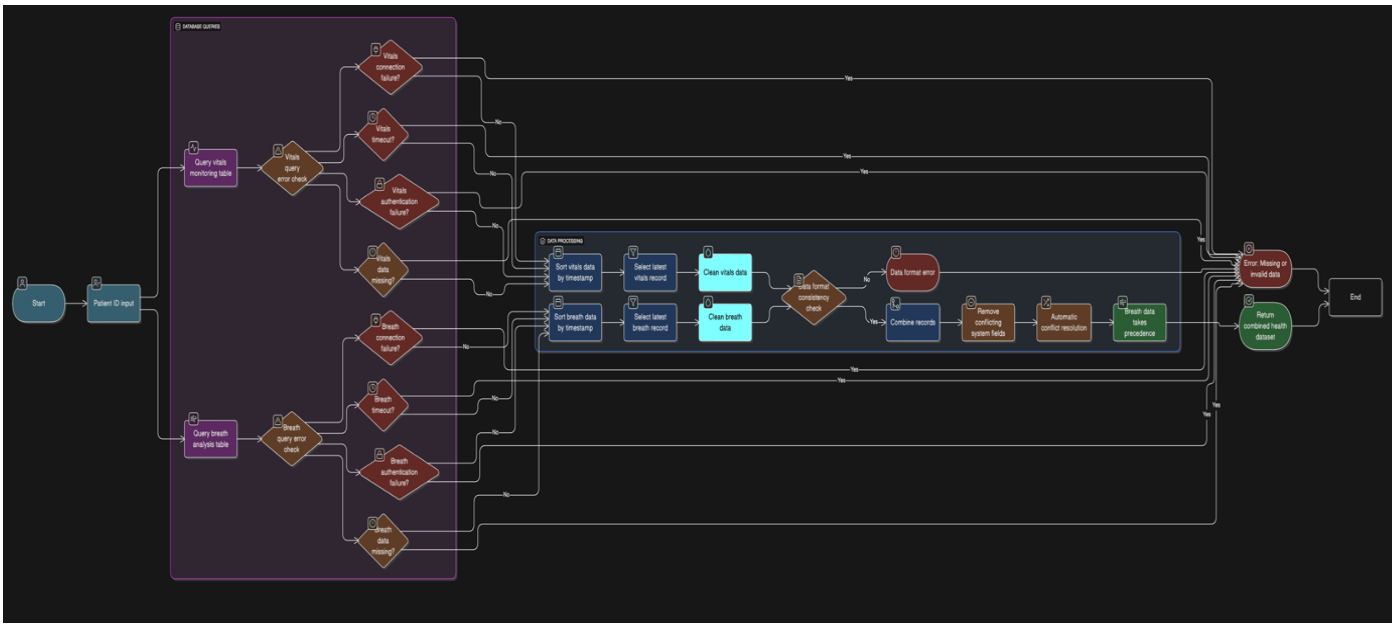
4) Cache results

- Cache Results in Database persists the successful prediction artifact (summary + risk vector) with the input timestamps, enabling instant responses on subsequent requests until upstream data changes. This application-layer cache complements any model-layer caching and keeps the final payload deterministic and auditable.

5) Output

- Return Structured Prediction: Respond with a stable JSON contract (patient_ID timestamps, model_version, narrative summary, risks/confidences, any alerts). A strict schema makes downstream rendering and alerting predictable and testable in clients and dashboards.

Patient Health Data Aggregation Flow:



1) Input and retrieval

- Start/Validate Patient ID, then pull the most recent vitals and breath records from storage to bind the request to a consistent state before inference; this is standard for production ML/health APIs to ensure deterministic predictions.

2) Cache validation (fresh vs stale)

- Retrieve prediction cache and cache timestamp, get latest data timestamps, then Compare Cache and Data Timestamps; if Cache Valid? is true, return cached output to avoid recomputation and reduce latency. Timestamp-based invalidation is simple and accurate for freshness without blind TTLs.

3) Prediction generation (if stale or missing)

- Format Health Data and call the LLM twice:
  - Generate Summary with Gemini API (clinical narrative).
  - Generate Risk Assessment with Gemini API (disease risk vector).
- On Summary/Risk errors, Retry Summary Generation and Retry Risk Assessment Generation using bounded retries/backoff—best practice for transient LLM/API failures.

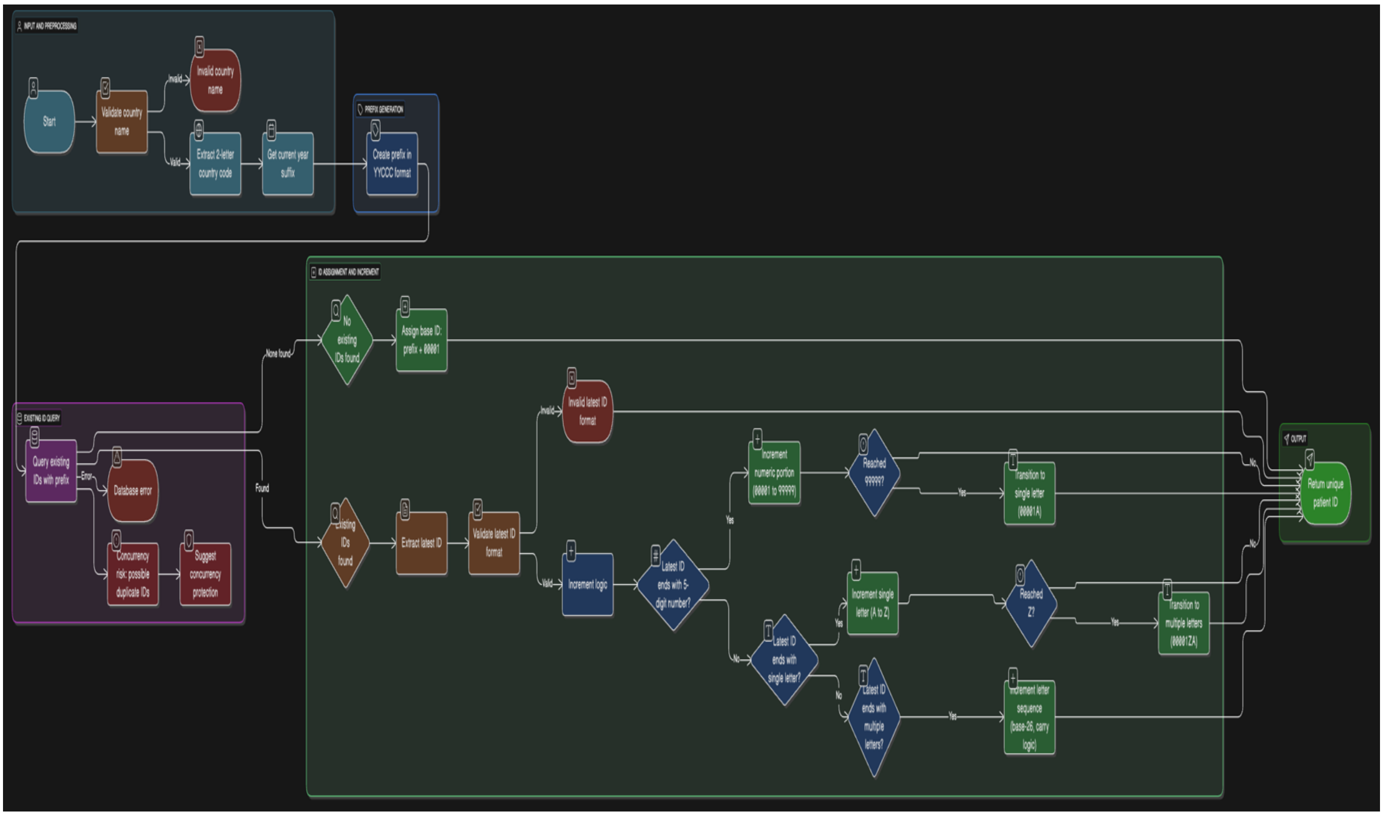
4) Cache results

- Cache Results in Database with the new summary, risk vector, model metadata, and the input timestamps. This complements any model-side context caching and gives deterministic application-layer reuse.

5) Output contract

- Return Structured Prediction with a strict JSON schema: patient_id, input timestamps, model_version, narrative summary, risk scores/confidences, and any alerts. Stable contracts enable safe dashboarding and alert triggers downstream.

Patient ID Generation Algorithm:



Major stages

A) Input pre-processing

- Start → Validate country name: Normalize and verify user-provided country; on failure, abort with invalid country error.
- Extract a stable country code (e.g., ISO country code) and determine a region or site suffix as needed for routing/analytics.

- Create a country profile in your “ID/COO domain” so subsequent steps can apply country-specific rules (prefix, encoding, reserved ranges). This mirrors standard input sanitation and normalization in production APIs.

B) Deduplication guard (preflight)

- Query existing IDs with the same prefix/profile; if the database operation fails, surface a database error.
- If a potential duplicate is detected (race/concurrency window), trigger a “suggest concurrency protection” branch (e.g., retry or switch to a sequence generator). This is a concurrency pre-check to reduce collision probability before constructing a candidate ID.

C) ID assembly and uniqueness

- Check existing ID count for the channel/profile: If none, Assign base ID prefix + region to initialize the namespace.
- Extract base ID (core components like YYYYMMDD/site or similar), then Validate base ID format to ensure it matches the schema.
- Increment logic: build a candidate suffix (e.g., sequence/entropy).
- Level checks to avoid orphan or digits-only number: ensure readability and mixed structure per your design policy (e.g., include alpha + numeric).
- If single-letter (weak) form happens, transition to single-letter error; if length short (≤ 2), transition to “insufficient token” error; in both cases, escalate to a stronger pattern (e.g., add more entropy or another segment) and continue.
- If candidate collides, increment again and repeat until a free slot is found. This mirrors “generate–check–bump” loops used widely for ID allocation.

D) Check digit and final validation

- Compute a check digit on the candidate using a well-known algorithm (e.g., Verhoef, Luhn, or ISO/IEC 7064 variant) to improve error detection when IDs are typed or transmitted manually. Verhoeff/Luhn/ISO-7064 are common for health and financial identifiers.
- Validate final ID format (prefix, length, allowed chars, and check digit).
- Optionally, log the assignment event for audit and traceability.

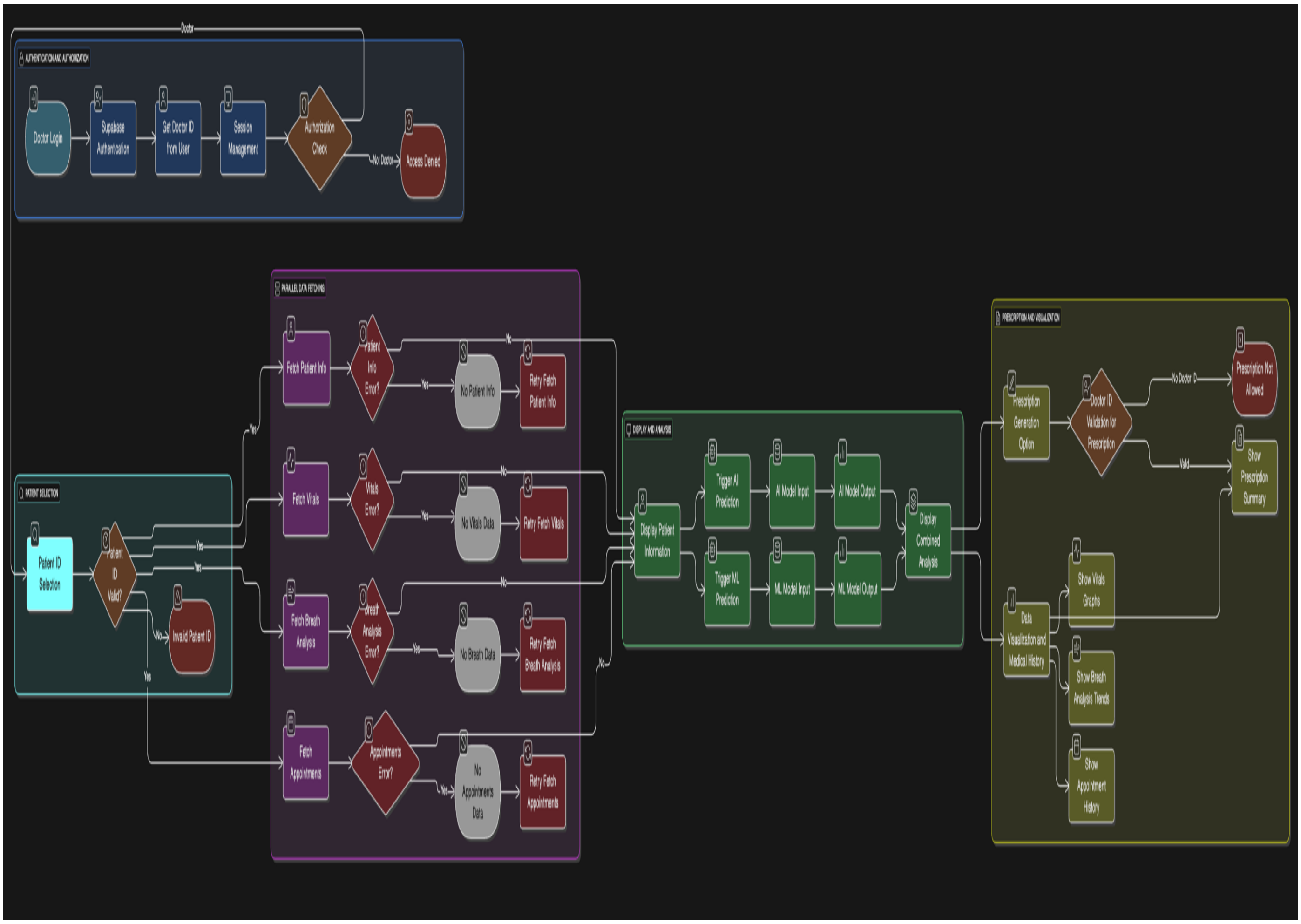
E) Concurrency-safe commit

- Persist via an atomic insert guarded by a unique index (e.g., UNIQUE(patient_id)); if the insert fails because another thread raced to the same ID, re-enter the increment/regen loop until success.
- This database-level uniqueness constraint is the single source of truth for collision prevention, complementing any preflight checks.

F) Output

- Return unique patient ID with canonical casing, separators, and optional metadata (country code, site, created at, checksum type) in a structured JSON response for clients.

Doctor Dashboard Patient Analysis Workflow:



Authentication and access

- Users create/login accounts with secure authentication, optional two-step verification, and session management before reaching protected areas; this mirrors common RPM/health portals and compliance guidance.

Intake and patient ID

- Intake verifies a Patient ID, audits identifiers, and persists a verified patient record; robust intake and ID hygiene are standard in RPM systems to ensure correct data linkage.

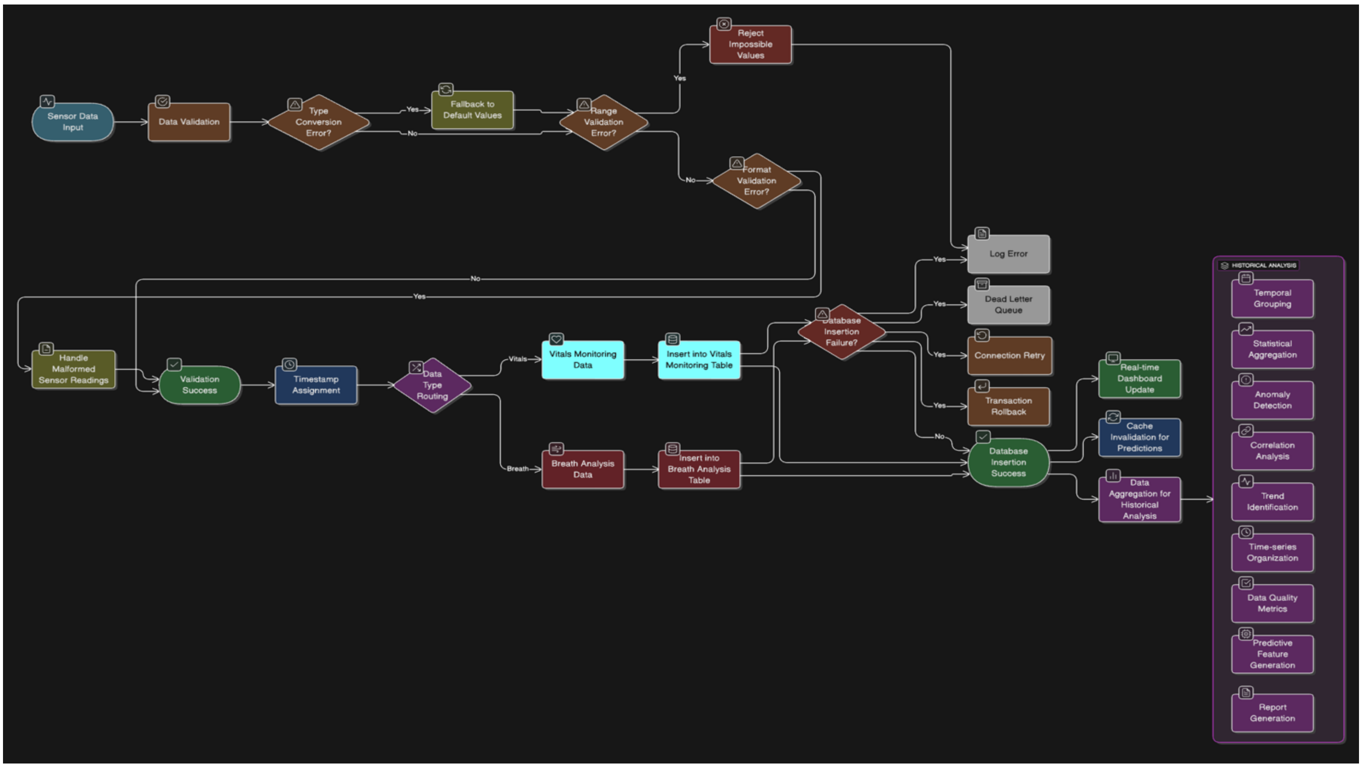
Data capture and update loop

- Workflows accept periodic inputs: Fetch today's vitals, fetch notes, fetch event analysis, handle appointments, and flag anomalies; failed fetches branch to errors, while valid inputs move to persistence and next steps. This mirrors IoT/RPM pipelines for continuous data collection and event logging.

ML analytics pipeline

- Trigger ML prediction, multi-model run, model ensemble/consensus, and multi-metric outputs; the diagram's "disease pattern monitoring" and consensus analysis reflect best practices for risk scoring and combining models for reliable decisions.
- Dashboards/visuals then expose "show vitals graphs," "data visualization of medical history," "show disease analysis trend," and "show appointment history," which matches clinical dashboarding recommendations for ML-CDS monitoring and end-user transparency.

Realtime-Health-data-Synchronization:



Ingestion and validation

- Sensor Data Input passes through schema/type checks with default fallbacks, range validation, and format validation; impossible values are rejected to protect downstream stages, aligning with data quality and schema enforcement best practices for real-time pipelines.
- Malformed readings are handled in a dedicated branch, while valid records receive authoritative timestamps before routing, which matches observability and data quality guidelines (freshness, volume, schema, distribution).

Routing and storage

- Data Type Routing splits flows into Vitals Monitoring Data and Breath Analysis Data, each inserted into its own table; enforcing typed destinations simplifies lineage, quality rules, and performance tuning per stream, a recommended pattern in robust data pipelines.

Fault tolerance on write

- On Database Insertion Failure, the pipeline Logs Error, sends payloads to a Dead Letter Queue, performs Connection Retry, and, if needed, Transaction Rollback; this mirrors retry-with-backoff, DLQ, and idempotent design guidance for reliable pipelines and reduced data loss.
- Successful inserts trigger Database Insertion Success, which fans out to Real-time Dashboard Update and Cache Invalidation for Predictions to keep UI/ML features consistent with the source of truth.

Real-time and historical analytics

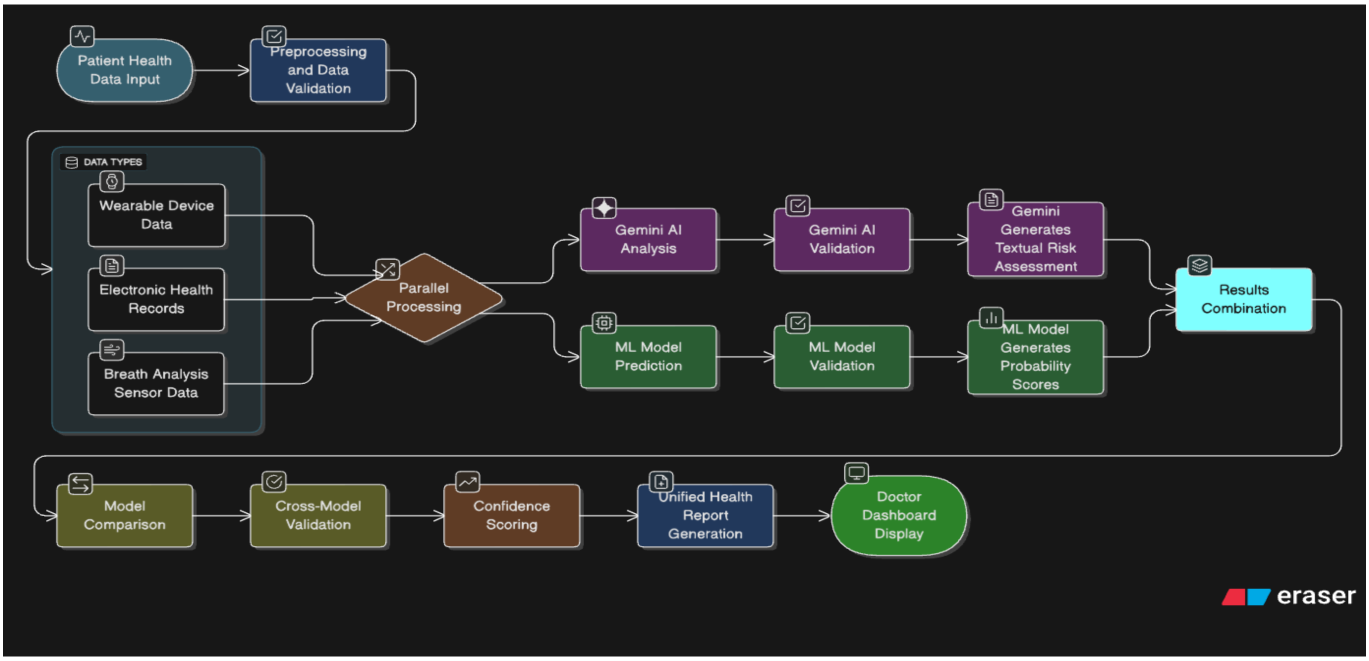
- A streaming path feeds real-time views; in parallel, Data Aggregation for Historical Analysis stores curated facts powering:
  - Temporal Grouping and Statistical Aggregation for cohort/time windows.
  - Anomaly Detection and Time-series Organization, used to highlight outliers and drifts in vitals/breath data.
  - Correlation Analysis and Trend Identification for clinical insights and monitoring model stability.
  - Data Quality Metrics and Predictive Feature Generation to enrich ML inputs and monitor pipeline health.
  - Report Generation for auditability and stakeholders.

Observability and governance hooks

- Centralized logging, metrics (throughput, latency, error rate), and automated alerts enable proactive monitoring and SLO enforcement, with dashboards consolidating pipeline health—key recommendations for data pipeline observability.

- Automated data quality checks (range, schema, distribution, freshness) at multiple points catch issues early, and lineage tracking clarifies dependencies for root-cause analysis, consistent with modern governance practices.

Multi-Model-Health-Assessment-Pipeline:



Data intake and validation

- Patient health inputs are pre-processed and schema-validated before dispatch into parallel branches for wearable streams, EHR data, and breath-sensor signals—consistent with safeguards recommended for clinical AI pipelines to reduce data quality errors upstream.

Parallel LLM + ML analysis

- LLM branch: Gemini AI Analysis → Validation → Textual Risk Assessment, reflecting growing use of multimodal LLMs to synthesize narrative risk summaries from diverse clinical inputs. Such M-LLMs are increasingly evaluated for clinical utility but require strict workflow evaluation and guardrails.
- ML branch: Model Prediction → Validation → Probability Scores, preserving calibrated numeric risk estimates so clinicians can compare likelihoods alongside LLM narratives.

Fusion and quality checks

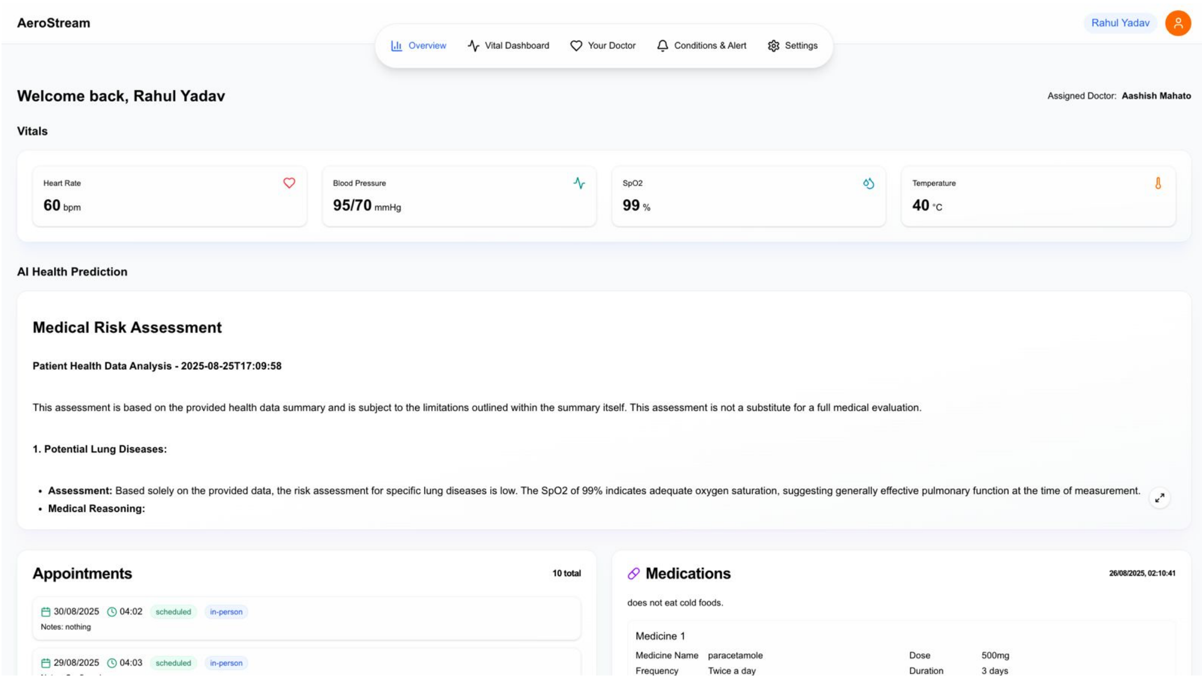
- Results Combination merges narrative outputs with probability vectors; then Model Comparison and Cross-Model Validation check agreement/divergence, a best practice to detect failure modes and quantify reliability of hybrid systems.
- Confidence Scoring computes a composite confidence using agreement patterns, calibration, and input quality, supporting safer triage of outputs for clinician review. Hybrid aggregation is consistent with evidence that ensembles and human–AI collectives can outperform either alone in open-ended medical diagnosis.

Reporting and display

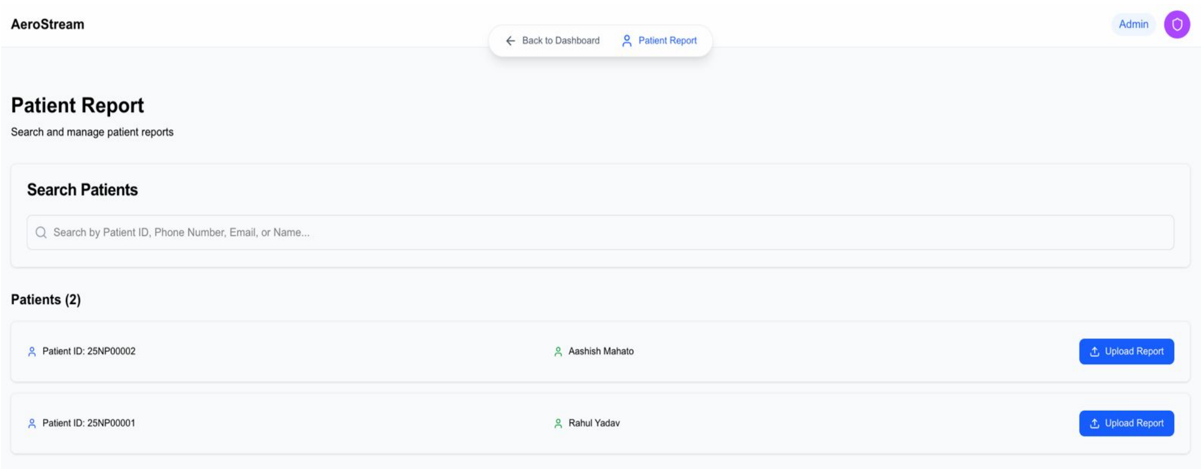
- Unified Health Report Generation produces a structured clinician-facing summary (text + numeric risk, rationale, and caveats), then the Doctor Dashboard Display surfaces it with explanations and supporting data. This mirrors recommended evaluation workflows for LLM-assisted clinical decisions, where transparency and validation findings are shown to end users.

8. Experimental validation results (Screenshot):

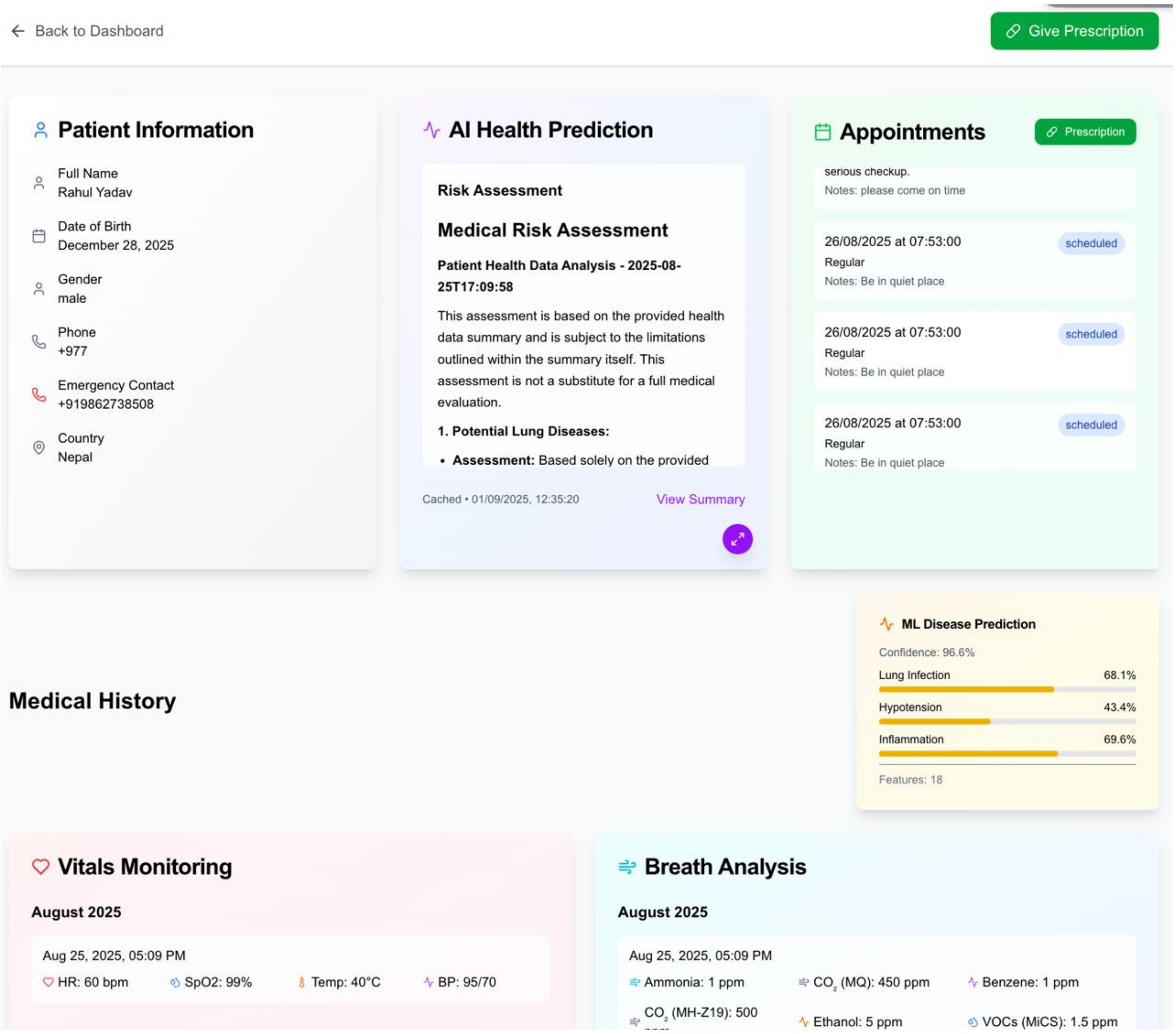
User:



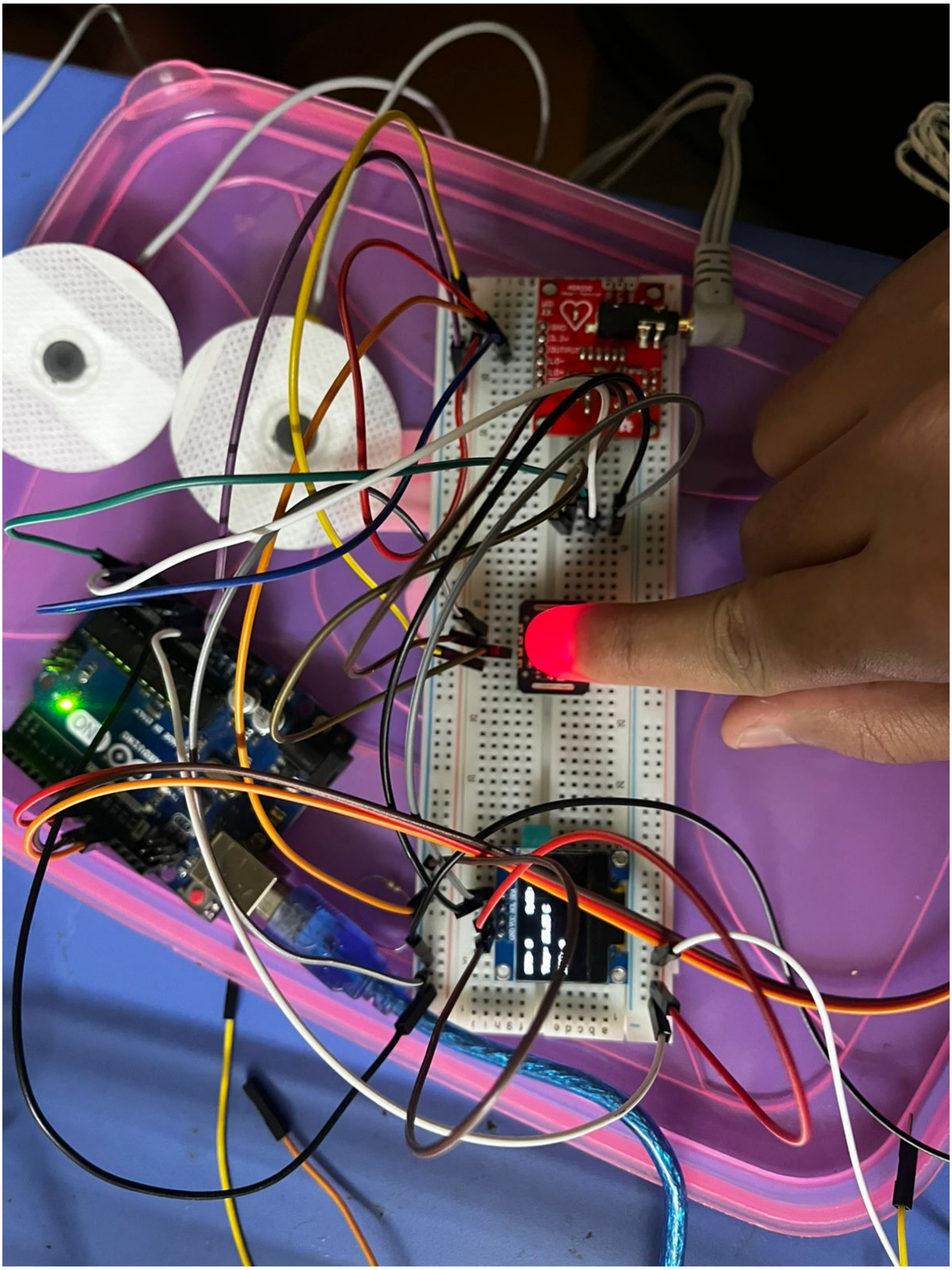
Admin:

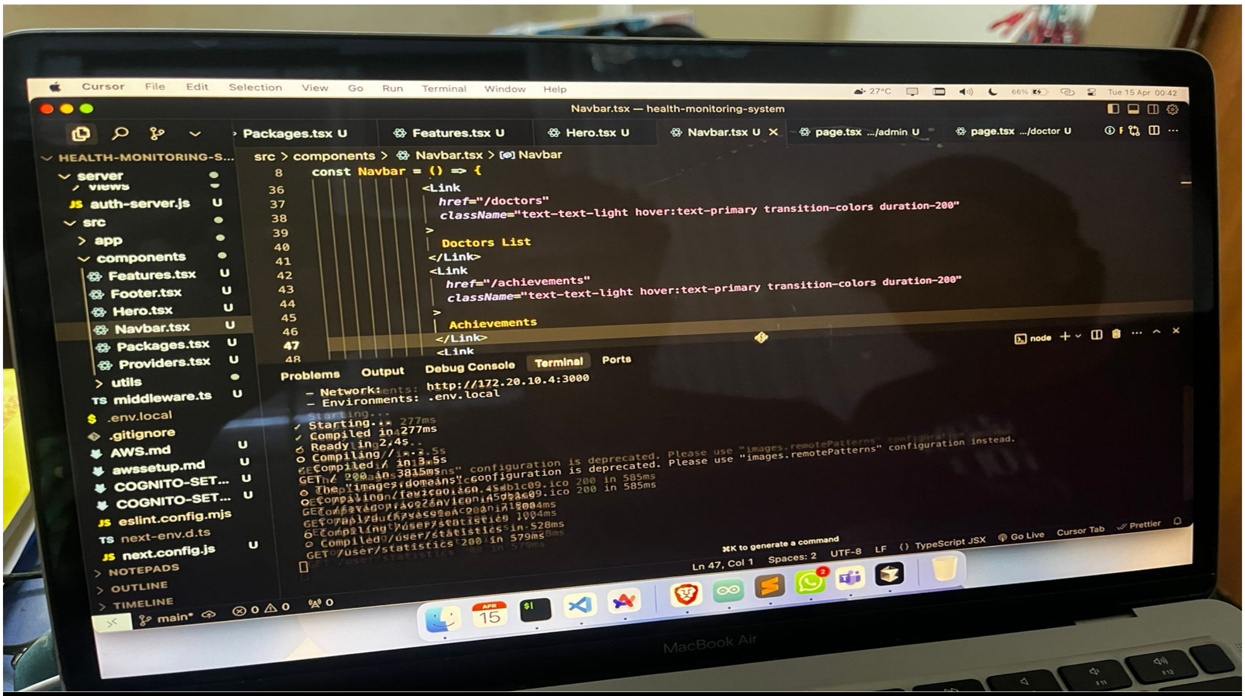


Doctor check-patient page:



Demo : (vital-sign)




Cursor File Edit Selection View Go Run Terminal Window Help
Navbar.tsx — health-monitoring-system
Packages.tsx U
Features.tsx U
Hero.tsx U
Navbar.tsx U
page.tsx .../admin U
page.tsx .../doctor U
HEALTH-MONITORING-S...
server
views
auth-server.js
src
app
components
Features.tsx
Footer.tsx
Hero.tsx
Navbar.tsx
Packages.tsx
Providers.tsx
utils
middleware.ts
.env.local
.gitignore
AWS.md
awssetup.md
COGNITO-SET...
COGNITO-SET...
eslint.config.mjs
next-env.d.ts
next.config.js
NOTEPADS
OUTLINE
TIMELINE
main*
src > components > Navbar.tsx > Navbar
const Navbar = () => {
<Link
href="/doctors"
className="text-text-light hover:text-primary transition-colors duration-200"
>
Doctors List
</Link>
<Link
href="/achievements"
className="text-text-light hover:text-primary transition-colors duration-200"
>
Achievements
</Link>
<Link
Problems
Output
Debug Console
Terminal
Ports
node
- Network: http://172.20.10.4:3000
- Environments: .env.local
✓ Starting...
✓ Compiled in 277ms
✓ Ready in 2.4s
Ln 47, Col 1
Spaces: 2
UTF-8
LF
TypeScript JSX
Go Live
Cursor Tab
Prettier
⌘K to generate a command
MacBook Air

Demo(Breath Analysis):

9. **What aspect(s) of the invention need(s) protection?**

Set 1: Hybrid E-Nose Sensor Array

Claim1.1:
An electronic nose (E-nose) system comprising a hybrid sensor array including MQ-135 and MQ-137 MOX sensors for detecting ammonia, carbon dioxide, and benzene associated with kidney, liver, and lung disorders.

Claim1.2:
As in Claim 1.1, the system further comprises an MH-Z19 NDIR sensor configured to measure CO_2 concentration in exhaled breath for respiratory and metabolic disorder detection.

Claim1.3:
As in Claim 1.1, the system further comprises a MiCS-5524 multi-gas sensor configured to detect ethanol and volatile organic compounds (VOCs) linked to liver dysfunction and lung diseases.

Claim1.4:
As in Claim 1.1, the system further comprises a quartz crystal microbalance (QCM) sensor configured to detect acetone as a biomarker for diabetes.

Claim1.5:
As in Claim 1.1, the system further comprises a chemo capacitor sensor configured to detect VOCs associated with early-stage infections and cancer markers.

Set 2: Machine Learning Risk Prediction

Claim2.1:
The system of Claim Set 1 processes fused outputs of the sensor array using embedded machine learning algorithms deployed on a Raspberry Pi 4 (or equivalent edge platform).

Claim2.2:
As in Claim 2.1, the machine learning pipeline generates a disease risk index along with a **confidence score**, wherein the confidence score is used to assist but not replace medical decision-making by clinicians.

Claim2.3:
The system of Claim 2.1 employs hybrid models comprising Random Forest, Support Vector Machine, and Long Short-Term Memory (LSTM) recurrent neural networks for temporal disease pattern detection.

Set 3: LLM Summarization & Historical Analysis

Claim3.1:
The system of Claim Set 2 integrates a large language model (LLM) configured to summarize historical and real-time patient health data into interpretable clinical reports.

Claim3.2:
As in Claim 3.1, the LLM provides longitudinal analysis of patient health trajectories, highlighting abnormal deviations and generating clinician-friendly narrative summaries.

Set 4: Telehealth Dashboard

Claim4.1:
The system of Claim Sets 1–3 provides a secure web platform with role-based dashboards for patients, doctors, and paramedics.

Claim4.2:
As in Claim 4.1, the patient dashboard displays personal health trends, alerts, and risk scores; the doctor dashboard displays patient lists, detailed risk summaries, and predictive analytics; and the paramedic/administrator dashboard manages user access and data integrity.

Claim4.3:
The system of Claim 4.1 ensures compliance with data privacy regulations (HIPAA, GDPR) through encrypted transmission, authentication, and role-based access control.

10. What is Technology readiness level of your invention? (Tick the appropriate TRL)

Research			Development			Deployment		
TRL 1	**TRL 2**	**TRL 3**	**TRL 4**	**TRL 5**	**TRL 6**	**TRL 7**	**TRL 8**	**TRL 9**
Basic Principles observed	Technology concept formulated	Experimental proof of concept	Technology validated in a lab	Technology validated in a relevant environment (industrially relevant in case of key enabling technologies)	Technology demonstrated in a relevant environment (industrially relevant in case of key enabling technologies)	System prototype demonstration in an operational environment	System complete and qualified	Actual system proven in an operational environment (competitive manufacturing in case of key enabling technologies, or in space)
			Applies					

Justification:
The invention has reached TRL 4 as it has been developed into a working prototype (Raspberry Pi–based hybrid E-nose with ML pipeline and web dashboard) and successfully validated in a controlled lab environment.

---------------------------END OF THE DOCUMENT----------------------------