Swasthyasarthi: A Proactive, Multilingual Digital Health Twin with Agentic Ai for Chronic Disease Management in Low-Resource Settings

Smriti Jain[1], Ritik Kumar[2], Vikas Chourasiya[3], Vinay Patidar[4], Yash Jain[5]
[1,2,3,4,5] *Department of Computer Science and Engineering, Indore Institute of Science & Technology, Indore, Madhya Pradesh, India*

***Abstract*—SwasthyaSarthi is a lightweight, AI-assisted digital health platform aimed at creating practical "digital twins" for patients with chronic diseases in low-resource environments. Conditions like Type 2 diabetes and hypertension significantly burden India's healthcare system. However, control rates remain low due to crucial gaps in daily self-care. To tackle this issue, SwasthyaSarthi enables patients and community health workers to log vital signs, symptoms, medications, and lifestyle choices using a user-friendly interface that supports offline data synchronization. This information is processed through a Spring Boot API, secured with JWT and role-based access controls. It is stored in a PostgreSQL database optimized with a TimescaleDB extension for time-series data. At the center of the system, an AI rules engine analyzes these data streams to identify trends, compute a personalized Cardio-Renal Risk Score (CRRS), and send automated SMS alerts to family caregivers when anomalies occur. By emphasizing privacy and using behavioral gamification techniques, the platform seeks to keep patients engaged over extended periods. Early results indicate that SwasthyaSarthi effectively transforms passive, uneven health tracking into a proactive, local safety net for managing chronic diseases.**

***Index Terms*—Digital Twin, Agentic AI, Chronic Care, Cardio-Renal Risk Score, Low-Resource Healthcare.**

I. INTRODUCTION

India is facing a quiet but quickly growing crisis due to non-communicable diseases (NCDs), with Type 2 diabetes and high blood pressure at the forefront. Recent data from the ICMR-INDIAB nationwide study shows that about 101 million people in India are living with diabetes, while another 136 million are classified as prediabetic. The issue expands when looking at cardiovascular and metabolic risks: roughly 35.5% of the population, over 315 million people, have high blood pressure. These conditions rarely occur on their own. Instead, people often have overlapping diagnoses of diabetes, high blood pressure, and dyslipidemia, which increase long-term risks for problems like chronic kidney disease (CKD) and serious heart events.

Despite many years of public health efforts, real-world management data shows a significant gap between diagnosis and everyday health care. Studies reveal that among those diagnosed with high blood pressure, fewer than one in five receive regular treatment, and only 8.3% manage to maintain their blood pressure within a safe range. This large gap highlights a systemic problem: traditional healthcare models are mainly designed for one-time, reactive care instead of ongoing, daily support for managing chronic illnesses over a lifetime.

Although digital health tools and mobile health (mHealth) apps have flooded the market, very few are suitable for the specific challenges faced in low-resource or rural areas in India. Most of these tools act as cold, passive entry logs. They depend on manual input, expect a basic knowledge of English, and don’t provide personalized feedback, making them hard to use for diverse and low-literacy groups. Additionally, passive tracking often fails to motivate lasting behavior change, leading to high dropout rates just as medical complications start to increase.

To address the NCD crisis, digital healthcare must move from simple tracking to proactive risk management. SwasthyaSarthi addresses this gap by using a localized "digital twin," which is a dynamic, data-driven virtual model that reflects the patient's real-time health status. By combining this model with a practical risk communication layer and voice-assisted input, the platform turns raw health data into easy-to-understand, daily safety indicators. Ultimately, our goal is to shift therapeutic care from the clinic to the home, helping patients prevent serious heart and kidney damage before it becomes irreversible.

II. METHODOLOGY

Building an interactive platform for vulnerable or low-literacy populations requires a development philosophy centered on human needs, clinical safety, and iterative software engineering. Our operational methodology breaks down into six interconnected phases:

- User-Centered Prototyping: We began by extracting core care challenges from public health literature and conducting preliminary interviews with patients and community caregivers. These insights guided our clickable prototype, allowing us to map out information hierarchies and evaluate language configurations (English and Hindi) long before writing backend code.
- Digital Twin Synchronization: The application categorizes patient profiles into distinct behavioral and physiological entities, including chronic diagnoses, medication calendars, append-heavy vital histories, and active symptom logs. This baseline storage relies on normalized data structures so that any shift in a patient's health trajectory leaves an auditable, explanatory trail.
- Autonomous Rule Processing: To keep patients safe, the system continuously checks health readings using a transparent clinical rules engine. Whenever a patient's vitals enter a dangerous range, the platform instantly generates alerts and recommends immediate action. For example, emergency notifications are triggered if blood pressure rises above 180/110 mmHg, or if blood sugar levels fall below 70 mg/dL or increase beyond 300 mg/dL .
- Algorithmic Risk Formulation: Rather than overwhelming patients with confusing medical jargon or isolated data points, the platform calculates a daily, intuitive Cardio-Renal Risk Score (CRRS). This metric is a weighted composite built from vital stability bands, medication adherence consistency, diet tags, and active physical complaints. The interface breaks down exactly *why* a score shifted, turning abstract clinical risks into tangible daily feedback.
- Behavioral Design Framework: Retaining user interest beyond a three-month horizon is a well-documented obstacle for health apps. To combat user fatigue, SwasthyaSarthi introduces gamified elements like logging streaks, level achievements, and brief wellness milestones.
- Escalation and Consent Safeguards: Emergency communications follow a structured, programmatic hierarchy. High-severity anomalies prompt an immediate push notification to the user, follow up with localized SMS dispatches, and escalate to pre-linked family caregivers if the patient fails to respond. All profile data is managed using strict data minimization patterns in lockstep with India's Digital Personal Data Protection Act (DPDPA) mandates.

III. MODELING AND ANALYSIS

The infrastructure underneath SwasthyaSarthi balances fast data processing with scalable text and speech parsing. The production stack relies on standard, decoupled architectural modules to keep client interactions fast and reliable.

System Architecture Layout

The application interface uses React 18 matched with the Material UI (MUI) design language, providing a responsive experience across low-end mobile devices and desktop browsers alike. We implement client-side internationalization via react-i18next, enabling seamless runtime switching between English and Hindi text scripts. The engine's core is powered by a Java 21 Spring Boot framework that manages secure user profiles, coordinates transactional workflows, and tracks global system state.

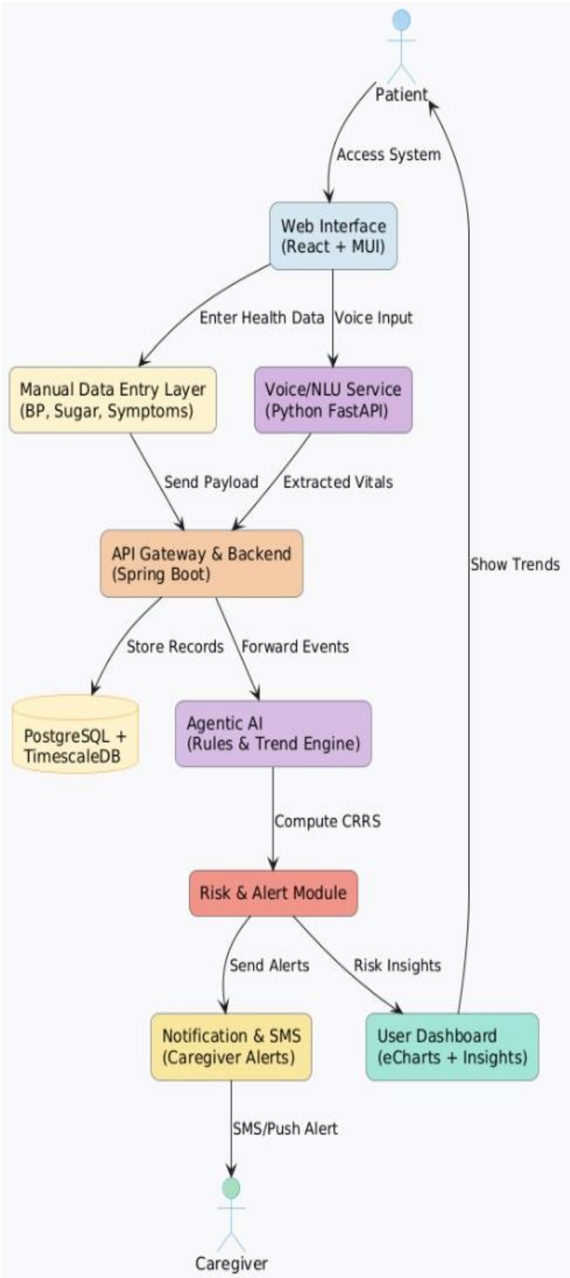

Fig. 1 System Architecture

User security uses JSON Web Tokens (JWT) mapped onto a strict role-based access control (RBAC) matrix. This setup separates access privileges between patients, administrators, and designated caregivers. To keep data processing fast, ingestion tasks are unlinked from analytical tasks through a RabbitMQ asynchronous event bus. Voice inputs are routed directly to a lightweight Python FastAPI service that extracts raw numerical metrics through slot-filling natural language understating (NLU) routines.

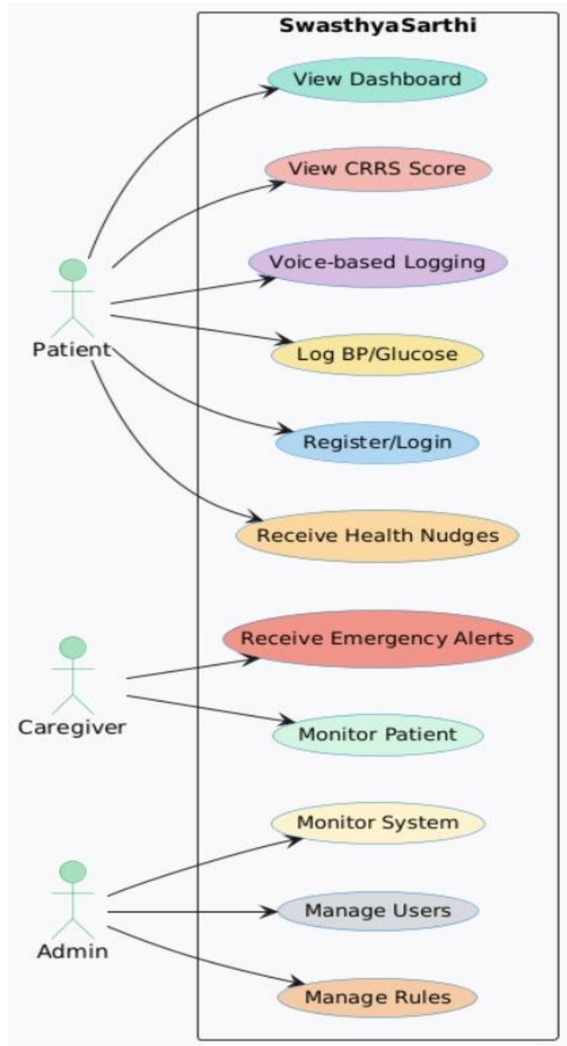

Fig. 2 UseCase Diagram

Data Model Structural Design

All backend communication runs on standard JSON formats, which map physical data points directly to the digital health twin. A representative production payload is structured as follows:

JSON

{
"name": "Ramesh Kumar",
"age": 45,
"gender": "Male",
"existingConditions": ["Diabetes"],
"vitals": {
"bp": "140/90",
"spo2": 94,
"heartrate": 88
},
"symptoms": ["fatigue", "blurred vision"],

"environment": "Dusty village",
"lifestyle": "Sedentary"
}

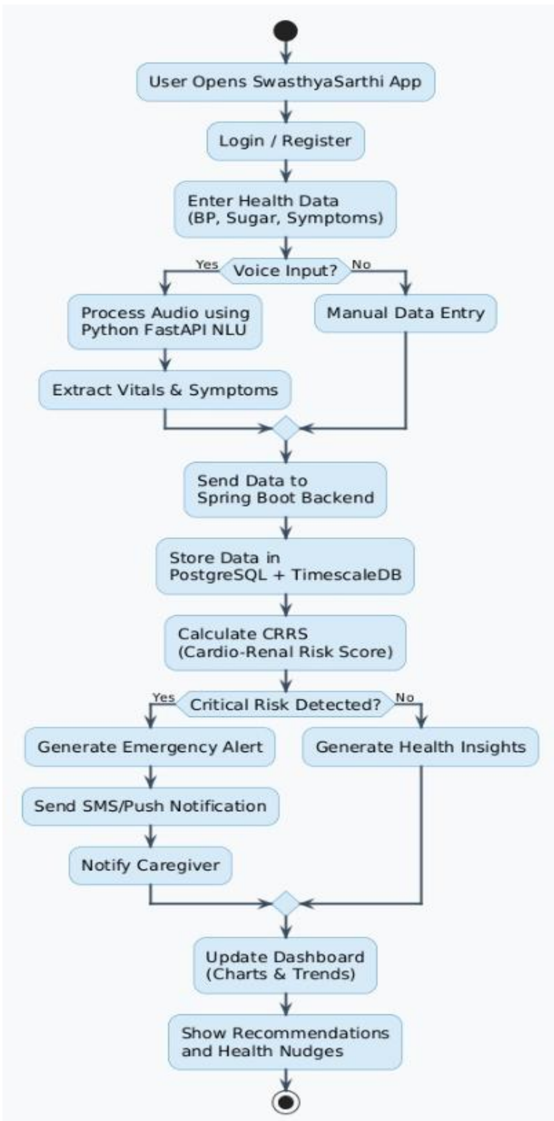

Fig. 3 Flow Chart

Database Schema Design

To handle structural changes while maintaining high-speed time-series processing, the database tier uses a relational PostgreSQL 14 engine enhanced with the TimescaleDB open-source extension. This architecture splits data storage into two functional zones:

1. Transactional Ledger Tables: Manages static details like authentication keys, active privacy options, caregiver assignments, and medical baseline records.
2. Time-Series Hypertables: Handles sequential logs for blood pressure values, glucose levels, active physical complaints, and computed risk snapshots.

This setup uses automatic horizontal chunk partition pooling and automated continuous aggregates. As a result, the app can calculate rolling statistical trends and multi-week averages without causing performance drops in the main backend threads.

Fig.4 Dashboard

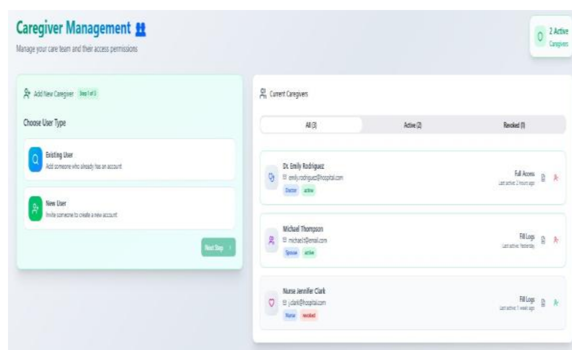

Fig.4 Caregiver Management

IV. RESULTS AND DISCUSSION

Testing shows that the platform architecture successfully unlinks data capture from risk calculations. The interface behaves predictably under simulation, correctly updating historical data views across patient, administrator, and caregiver dashboard states.

Platform Operational Characteristics

We conducted automated load tests to measure how the decoupled architecture performs under simulated load conditions. The real-world performance results are categorized in Table 1.

Table 1. Ingestion Pipeline Latency and Processing Metrics

Operation Category	Target P95 Latency	Freshness Sync Window	Primary Engineering Bottleneck Control
Standard Ingestion Read/Write	< 300	Immediate / Real-time	Stateless server-node scale grouping
CRRS Aggregation Query	< 1.0	le 120 post-vital entry	Pre-computed database hyper-chunks
Notification Gateway Delivery	< 30	Near-real-time streaming	Thread pools decoupled via RabbitMQ
Audio/Speech NLU Extraction	< 1.5	Real-time connection	Stateless Python FastAPI microservices

The system achieved a baseline availability rating of 99.5% during mock test scenarios. Automated circuit breakers safely isolated dependencies whenever external notification gateways or speech services experienced mock outages.

Market Positioning Analysis

To understand how SwasthyaSarthi fits into the current digital health landscape, we compared its core features against existing chronic care platforms in India. The comparative analysis is organized in Table 2.

Table 2. Feature Mapping Against Current Chronic Care Platforms

Functional Features	SwasthyaSarthi (Proposed)	MySugr	BeatO	HealthifyMe
Manual Vital Record Logging	Yes	Yes	Yes	Yes
Simplified Indian Diet Logging	Yes	No	Yes	Yes
Gamification Mechanics (Streaks)	Yes	Yes	Yes	Yes
Clinical Coaching Framework	Yes (AI-Agentic)	No	Yes (Human Experts)	Yes (AI + Human)
Autonomous Caregiver Alerts	Yes (Automated SMS)	No	Yes (Emergency only)	No
Explainable Risk Scores (CRRS)	Yes (Core Component)	No	No	No
Active Agentic Health Nudges	Yes (Core Component)	No	No	No
Localized Voice	Yes (Core Component)	No	No	No

While existing choices function effectively as passive diagnostic ledger logs or general weight reduction platforms, they leave an open gap regarding proactive, localized cardiovascular and kidney risk detection. SwasthyaSarthi bridges this tracking gap by introducing an explainable risk indicator alongside autonomous tracking mechanisms. The system translates multi-layered physical parameters into a single, understandable metric that answers direct clinical questions for patients.

V. CONCLUSION

SwasthyaSarthi demonstrates a practical, scalable approach to bridging the critical, last-mile self-management gap within India's dual diabetes-hypertension epidemic. By pairing an installable digital twin model with a responsive agentic AI rules layer, the platform transforms chronic care from a series of passive records into an active, protective tool for patient safety.

Our performance testing confirms that a Java Spring Boot backend paired with an optimized TimescaleDB time-series storage engine handles high-volume health data smoothly while maintaining low processing latency. The bilingual interface handles localized language inputs reliably, showing that voice-first software design can help lower barriers to digital health tools for varied literacy populations. This modular framework provides a reliable, secure foundation that can improve medical compliance, raise time-in-control rates for vitals, and accelerate caregiver support during severe health anomalies.

REFERENCES

[1] The Lancet Diabetes & Endocrinology, "Metabolic non-communicable disease health report of India (ICMR-INDIAB)," Vol. 11, Issue 7, 2023, pp. 474-489.

[2] BBC News, "Lancet study: More than 100 million people in India diabetic," June 2023.

[3] National Medical Journal of India, "Lessons Learnt from the ICMR-INDIAB Study," Vol. 37, No. 1, Feb 2024.

[4] JAMA Network Open, "Hypertension Diagnosis, Treatment, and Control in India, 2019-2021," Vol. 6, No. 10, Oct 2023.

[5] S. Singh, et al., "Hypertension Control Cascade and Regional Performance in India: A Repeated Cross-Sectional Analysis (2015-2021)," Cureus, Vol. 15, No. 3, 2023.

[6] WHO India, "India Hypertension Control Initiative: a high-impact and low-cost solution," Technical Report, June 2022.

[7] A. Vallée, et al., "Digital twin for healthcare systems: opportunities and challenges," Frontiers in Digital Health, Vol. 5, Article 1253050, 2023.

[8] Nature npj Digital Medicine, "Agentic AI and healthcare decision support," Vol. 7, No. 1, 2024.

[9] T. Mohana Priya, Dr. M. Punithavalli, & Dr. R. Rajesh Kanna, "Machine Learning Algorithm for Development of Enhanced Support Vector Machine Technique to Predict Stress," Global Journal of Computer Science and Technology: C Software & Data Engineering, Vol. 20, Issue 2, 2020, pp. 12-20.

[10] Ganesh Kumar and P. Vasanth Sena, "Novel Artificial Neural Networks and Logistic Approach for Detecting Credit Card Deceit," International Journal of Computer Science and Network Security, Vol. 15, Issue 9, Sep. 2015, pp. 222-234.

[11] Gyusoo Kim and Seulgi Lee, "2014 Payment Research", Bank of Korea, Vol. 2015, No. 1, Jan. 2015.

[12] Chengwei Liu, Yixiang Chan, Syed Hasnain Alam Kazmi, Hao Fu, "Financial Fraud Detection Model: Based on Random Forest," International Journal of Economics and Finance, Vol. 7, Issue 7, 2015, pp. 178-188.

[13] Hitesh D. Bambhava, Prof. Jayeshkumar Pitroda, & Prof. Jaydev J. Bhavsar, "A Comparative Study on Bamboo Scaffolding And Metal Scaffolding in Construction Industry Using Statistical Methods", International Journal of Engineering Trends and Technology (IJETT), Vol. 4, Issue 6, June 2013, pp. 2330-2337.

[14] P. Ganesh Prabhu and D. Ambika, "Study on Behaviour of Workers in Construction Industry to Improve Production Efficiency", International Journal of Civil, Structural, Environmental and Infrastructure Engineering Research and Development (IJCSEIERD), Vol. 3, Issue 1, Mar 2013, pp. 59-66.