PersonHealth: A Healthcare Website with Chatbot Integration Using Django and Language Models

Lagudu Thrilok Naidu[1], Dr. N. Padmaja Lavanya Kumari [2]

[1]*M.Tech Student, Department of Computer Science and System Engineering, Andhra University College of Engineering(A), Visakhapatnam, Andhra Pradesh, India*

[2]*Assistant Professor, Department of Computer Science and System Engineering, Andhra University College of Engineering(A), Visakhapatnam, Andhra Pradesh, India*

Abstract: **This study introduces *PersonHealth*, an integrated web-based healthcare platform designed to offer both general and personalized medical support. The system leverages advanced Natural Language Processing (NLP) and machine learning models to enhance user engagement and response accuracy. For general medical queries, a fine-tuned BioGPT model is employed to deliver medically relevant answers. Meanwhile, a Retrieval-Augmented Generation (RAG) architecture, utilizing FAISS for semantic similarity search and the Gemma model for generation, powers profile-based, symptom-only, and hybrid queries. This setup ensures context-aware, user-specific recommendations based on inputs like age, weight, height, and diet. Key interactive modules include mood tracking, nutrition summary fetching, water intake logging, and BMI calculation, each contributing to a comprehensive health overview. The platform also integrates a specialist finder through symptom-to-specialist machine learning mapping, enhanced with Google Maps for real-world connectivity. Wearable simulator support enables real-time monitoring of heart rate, steps, and sleep, accompanied by 7-day graphical trends of vitals and weight. System evaluation across all modules indicated high accuracy and contextual alignment. Personalized health tips matched user health contexts effectively, and symptom-to-specialist predictions were consistent and precise. The PersonHealth platform exemplifies a scalable, AI-driven solution for structured, accessible, and intelligent digital healthcare—combining retrieval, reasoning, tracking, and navigation in one unified interface.**

Keywords: Biomedical NLP, Django REST Framework, Healthcare Chatbot, Wearable Integration, ReactJS Frontend, Retrieval-Augmented Generation (RAG)

I.INTRODUCTION

Access to reliable, personalized health guidance remains a challenge in many healthcare systems, where users often rely on fragmented online resources or generalized advice that lacks individual context. While other platforms provide doctor listings and appointment booking, they may not combine comprehensive health information retrieval, lifestyle tracking, and specialist search in one cohesive framework. PersonHealth addresses this gap by integrating general medical Q&A, context-aware symptom assessment [7], lifestyle monitoring, and nearby specialist recommendations into a unified platform built with ReactJS and Django. A chatbot is integrated into the platform, powered by a fine-tuned BioGPT model and RAG using FAISS and Gemma, building on foundational GPT-based conversational agent architectures that enable responses tailored to each user's profile for greater precision and relevance [4],[8],[9],[10]. Supplementary modules track mood, nutrition, water intake, BMI, and wearable-derived metrics such as heart rate, step count, and sleep patterns, which are visualized over the last seven days [12]. A wearable simulator (a) shown in Figure 5 ensures functionality without physical devices, allowing continuous testing and demonstration. Evaluation of all modules showed consistently strong contextual relevance, accurate lifestyle tips, and precise specialist matching (see Table 1) indicating the system's potential as a scalable, user-centric healthcare support tool. The aim of this work is to demonstrate that a unified, retrieval-powered, and profile-aware system can enhance accessibility, accuracy, and personalization in digital healthcare delivery.

II. PROBLEM STATEMENT

In the current digital healthcare landscape, users often face fragmented solutions where medical information, lifestyle tracking, and doctor consultations are offered through separate platforms. This lack of integration results in incomplete health guidance, limited personalization, and delays in accessing relevant care. While some existing platforms focus on doctor search and appointment scheduling, they rarely incorporate context-aware symptom assessment or lifestyle-based recommendations. Moreover, most systems do not leverage structured data from wearable devices to adapt advice dynamically to a user's health status. The absence of a unified, retrieval-powered framework limits the effectiveness and efficiency of online healthcare delivery.

The challenge lies in creating a single, scalable platform that combines intelligent health information retrieval, personalized symptom evaluation, lifestyle monitoring, and real-time specialist recommendations, while maintaining accuracy, contextual relevance, and user-friendliness.

III.METHODOLOGY

The proposed healthcare assistant system was developed using a structured, modular methodology to ensure clarity, maintainability, and real-world applicability. The process was divided into the following phases:
1. Data Collection and Preparation:
Domain-specific datasets were curated for different modules:

- Doctor Search: doctor dataset containing names, specialties, and gender
- Medicine & Remedies: medicine dataset .and natural remedies dataset for medicine lookups and natural remedy recommendations.
- Wellness & Nutrition: food nutrition. dataset and wellness quotes dataset for nutrition tracking and motivational quotes.
- Contextual Chatbot Data:
- General health QA, profile-based QA, symptom-based suggestion, profile symptom-based suggestion datasets used for general and personalised methods
- Symptom-to-Specialist Mapping: Mapping dataset linking symptoms to relevant medical specialists for fast triage suggestions.
- Health-Tips-Dataset: Curated personalized health tips linked to mood, nutrition, water intake and wearable data.

All datasets underwent cleaning, deduplication, and structured formatting to ensure fast retrieval and avoid cross-module interference.

2. System Design and Backend Logic:
The backend was implemented using Django and Django REST Framework with PostgreSQL as the primary database. Key modules included:

- Secure authentication using JWT with OTP-based password recovery.
- APIs for profile management, symptom tracking, mood logging, water/nutrition intake, appointment booking, trends, medicine/natural remedies search, health tips and AI chatbot queries.
- Google Maps API integration for location-based doctor filtering.
- Wearable simulator integration via Web Bluetooth to collect real-time or simulated heart rate, steps, and sleep data.

3. AI Model Integration:
Two main AI pipelines were deployed:

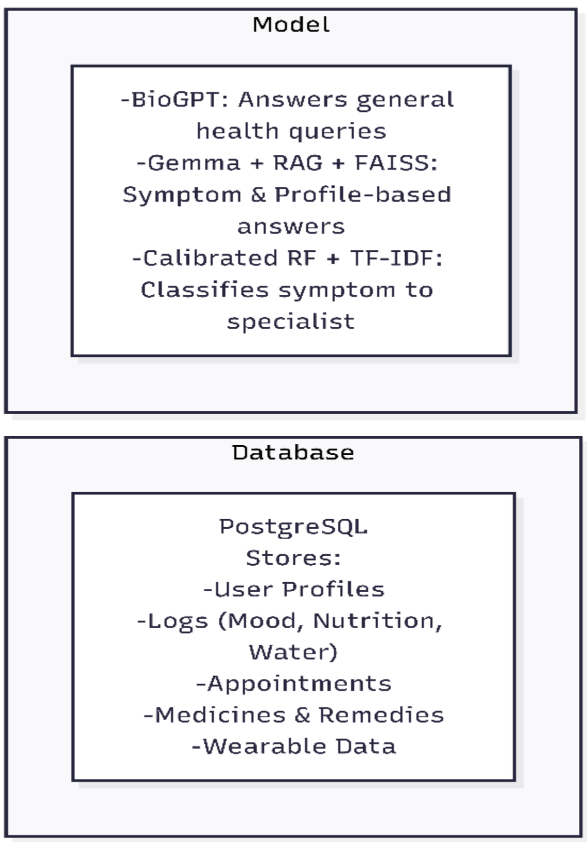
- BioGPT: Fine-tuned for general with the model from, open-ended health Q&A [8].
- Gemma + RAG Pipeline: Uses FAISS for semantic retrieval, [9],[10],[13],[14] followed by contextual response generation based on symptoms or user profiles.
- Health Tip Prediction: Cosine similarity-based recommendation engine matching user health metrics (mood, nutrition, water intake, heart rate, steps, sleep) with the most relevant health tip from the curated dataset.
- Symptom Specialist: A Random Forest classification model was employed to train and test a predictive system capable of identifying the most appropriate medical specialist for a given symptom with high accuracy. The deployed specialist predictor loads a calibrated Random Forest (calibrated_rf_model.joblib) with a combined word/character TF-IDF vectorizer and a label encoder. Given symptom text, the vectorizer generates features (tfidf_vectorizer_combined.joblib), the classifier outputs an encoded class, and the label encoder (label_encoder.joblib) returns the final specialty.

Prompt tuning and embedding validation were performed to ensure accuracy, reduce hallucination, and maintain contextual relevance.

4. Frontend Development and Interaction:
The frontend, built using ReactJS and Vite, [12] was styled with Tailwind CSS for a responsive and modern UI. Core features included:

- Chatbot interface supporting general [8],[9],[10], symptom-based, profile symptom and profile-based queries. Check the interaction flow in Figure 2 and UI in

- Figure *4*
- Health dashboard with visual analytics for wearable and logged data. Check the UI (b), (c) from Figure 5
- Appointment booking system with real-time doctor filtering. Check the process in Figure 3 and UI in Figure *6*
- Interactive modules for mood tracking, nutrition logging, water intake, and wellness quotes.



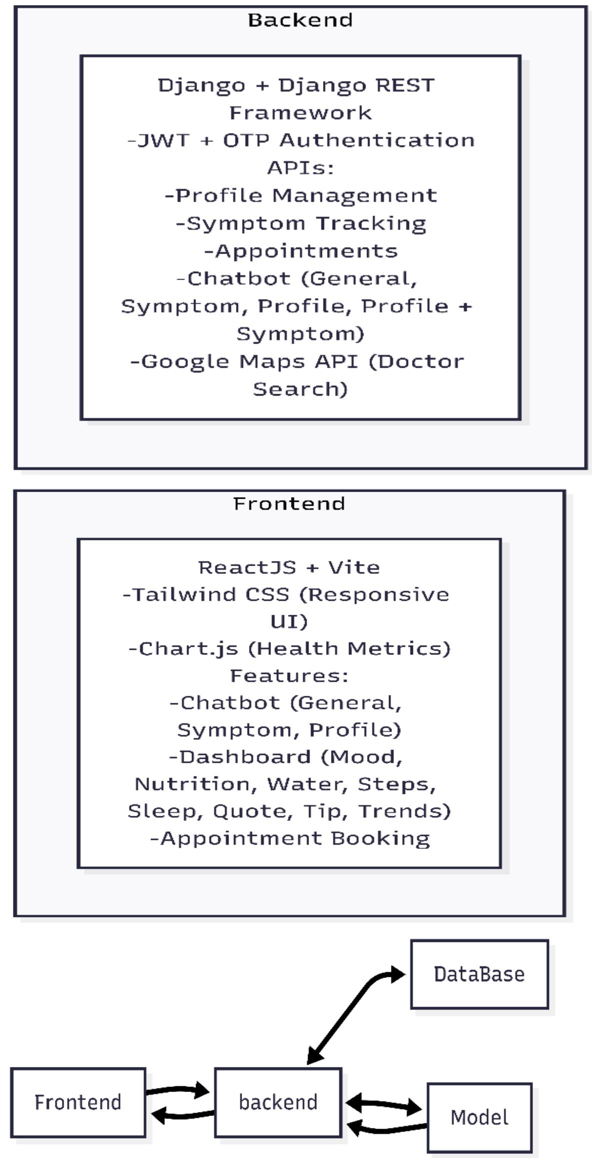


Figure 1: System Architecture

5. Evaluation Framework:

The AI components of the system were evaluated using a curated 100-sample test set.

- Responses from the chatbot and health tip generator were rated according to Good, Partial, and Poor criteria.
- Accuracy was calculated as the proportion of Good and Partial responses relative to the total test samples.
- Based on the evaluation outcomes, iterative refinements were applied to the datasets, embedding generation, and prompt formulations to improve response quality.

IV. TECHNOLOGIES USED

This section summarizes the technologies, tools, and frameworks employed in the development of the healthcare assistant system. The stack spans across frontend, backend, AI modelling, and development environment layers to ensure a robust, scalable, and maintainable solution.

1. Frontend Technologies:

- ReactJS: A component-based JavaScript library used for building dynamic and responsive user interfaces.
- Vite: A modern frontend build tool that enhances development speed and performance in React projects.
- Tailwind CSS: A utility-first CSS framework adopted for designing clean, responsive layouts.
- JavaScript (ES6+): Applied for frontend logic, interaction, and state management.
- Chart.js & React Icons: Used to display health analytics and enhance UI with iconography.

2. Backend Technologies:

- Python: The primary backend language for implementing logic, models, and APIs.
- Django: A high-level Python framework that manages server-side operations and database interactions.
- Django REST Framework (DRF): Enables creation of RESTful APIs for communication between frontend and backend.
- PostgreSQL: A robust relational database system used to store user profiles, logs, metrics, and chatbot records.
- JWT Authentication: JSON Web Token mechanism used for secure session handling and protected API access.

3. AI and NLP Tools:

- BioGPT: A transformer-based large language model fine-tuned for biomedical and health-related question answering.
- Gemma: A lightweight transformer used in conjunction with retrieval-based prompts for context-aware response generation.
- FAISS: Facebook AI Similarity Search, a vector search library used for indexing and retrieving semantically similar health entries.
- Sentence Transformers: Applied to generate dense embeddings from user inputs for effective semantic retrieval in RAG-based pipelines.
- Scikit-learn (Random Forest): Used to train and deploy the symptom-to-specialist classification model.

4. Development and Deployment Environment:

- Anaconda (Conda Environment): Used for environment isolation, dependency control, and reproducibility of the backend pipeline.
- Web Bluetooth API: Integrated with the wearable simulator to transmit real-time heart rate, step count, and sleep data.

V. SYSTEM SETUP AND IMPLEMENTATION

A. Project Setup

1. Development Environment Setup:

The backend environment was configured using Anaconda to ensure clean dependency management and reproducibility. A dedicated Conda environment was created, and essential libraries such as Django, scikit-learn, transformers, and FAISS were installed via pip and conda.

```
conda create --name env1 django=5.2.1 python=3.11.11
conda activate env1
```

2. Backend System Setup (Django Project): check the official document from [11]

A Django project named backend was initialized, within which a main app called api was created. The PostgreSQL database connection was configured in settings.py, and the Django REST Framework (DRF) was used to modularize APIs for authentication, user profile, chatbot interaction, health logging, and other services.

```
django-admin startproject backend
cd backend
python manage.py startapp api
```

3. Frontend Setup (React + Vite + Tailwind CSS): check the official document from [12]

The frontend was developed using Vite with ReactJS for component-based UI and Tailwind CSS for responsive styling. The following command sequence initializes the frontend and sets up Tailwind:

```
npm create vite@latest frontend -- --template react
cd frontend
npm install
npm install tailwindcss postcss autoprefixer
npx tailwindcss init -p
```

Custom components were developed for chatbot, dashboard, wearable device interface, health logging, doctor booking, and user profile.

4. PostgreSQL Integration:
PostgreSQL served as the primary database, configured as follows in settings.py:

```
DATABASES = {
  'default': {
    'ENGINE': 'django.db.backends.postgresql',
    'NAME': 'healthuser',
    'USER': 'postgres',
    'PASSWORD': 'rollno21',
    'HOST': 'localhost',
    'PORT': '5432',
  }
}
```

Tables were automatically generated using Django models with makemigrations and migrate.
The illustration of the system setup is shown in Figure 1

B. System Implementation
The system was implemented as a complete end-to-end health monitoring and assistant platform, combining wearable integration, AI-powered chatbot interaction, and structured user data logging. The major components are described below:

1. User Authentication and Profile Management:
A secure JWT-based login system enables user registration, login, and OTP-based password recovery [5]. post-login, users can edit their age, weight, height, and dietary preferences — essential for personalized chatbot interaction and BMI calculations.

2. Health Dashboard and Wearable Device Integration:
Users can connect to a wearable device via Web Bluetooth. For development purposes, a custom Android-based wearable simulator shown in Figure 5 was created to mimic heart rate, steps, and sleep data, simulating real wearable behaviour. The dashboard visualizes trends using Chart.js, and real-time alerts are triggered for abnormal readings (e.g., heart rate > 120 bpm).

3. Health Logging and Visual Feedback:
Users can log mood, water intake, nutrition, weight, BMI, and wearable data. Whenever a wearable is connected, new readings are saved to the backend and later plotted on trend graphs in the dashboard using timestamped data.

4. Medicine and Natural Remedies Lookup:
Users can search for medicines or home remedies by typing common ailments. Pre-processed datasets are used to return safe usage guidelines and natural solutions.

5. Wellness Quotes and Engagement:
The dashboard displays rotating motivational wellness quotes, dynamically changing based on time or refresh triggers. These are fetched from a curated dataset to maintain user engagement.

6. Chatbot Interaction and Modes:
The chatbot supports three distinct modes [8],[9],[10]:

- General Mode: Handles open-ended queries using a fine-tuned BioGPT model.
- Symptom Mode: Accepts structured inputs (Symptom, Severity, Duration, Description) and returns contextual responses via a FAISS + Gemma RAG pipeline.
- Personalized Mode: Enhances symptom or general responses using profile features like age, weight, height, and diet.

The RAG mechanism ensures the AI uses semantically relevant data, reducing hallucination and improving medical accuracy [10].
See the Chatbot UI with sample inputs in

Figure *4*

7. Appointments and Doctor Search:

- Users can search for doctors and book appointments via:
  - Nearby (current location using Google Maps),
  - State-wise (e.g., Telangana, Kerala),
  - Regional/National (e.g., South India, Pan India).
- Symptom → Specialist: A model maps user symptoms to medical specializations.
- Nearby Search: Uses GPS coordinates with Google Maps Nearby API.

- Text Search: Converts text like "South India" to city list and uses Google Text Search API.
- Doctor names are extracted via regex from clinic titles. If missing, fallback doctors are used.
- Diagnostic centres are filtered using keywords like "lab", "scan", etc.
- Results include doctor name, clinic, specialization, photo, address, and location.
- Users can view, book, and cancel appointments via the frontend React app.
- Results displayed with doctor photo, clinic, and address. See in figure 6
- Appointments can be viewed/booked/cancelled via the React UI.

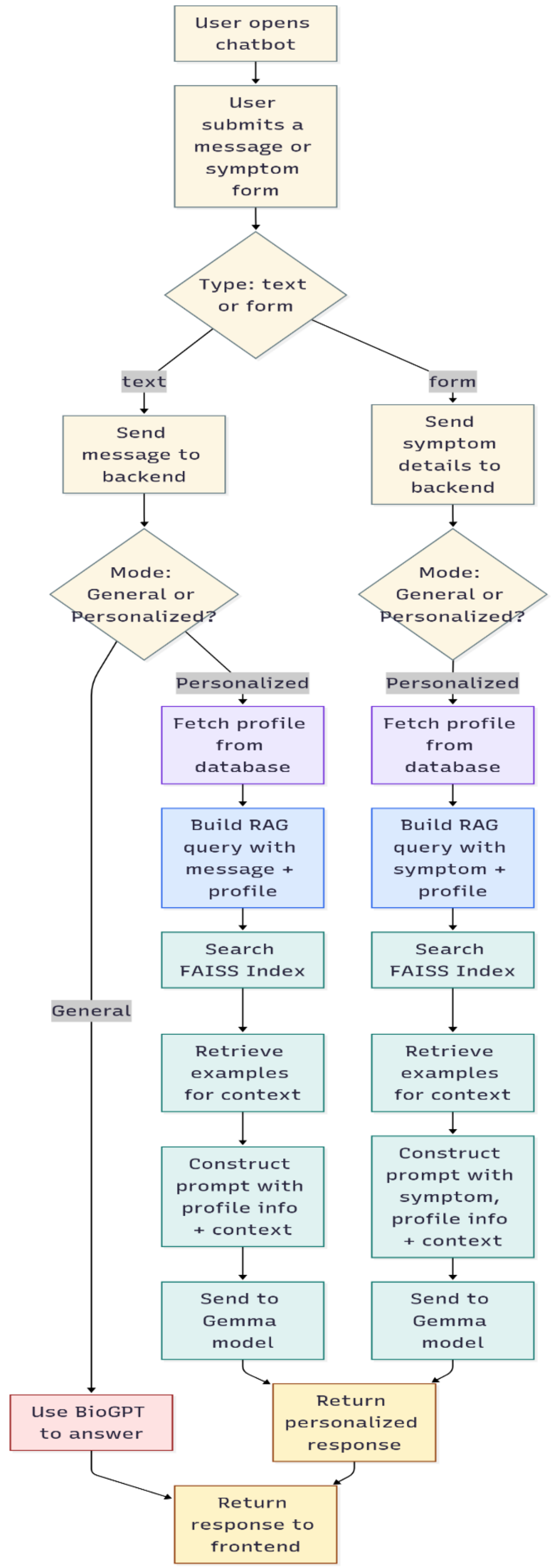


Figure 2: Chatbot Interaction Flow for General and Personalized Health Queries

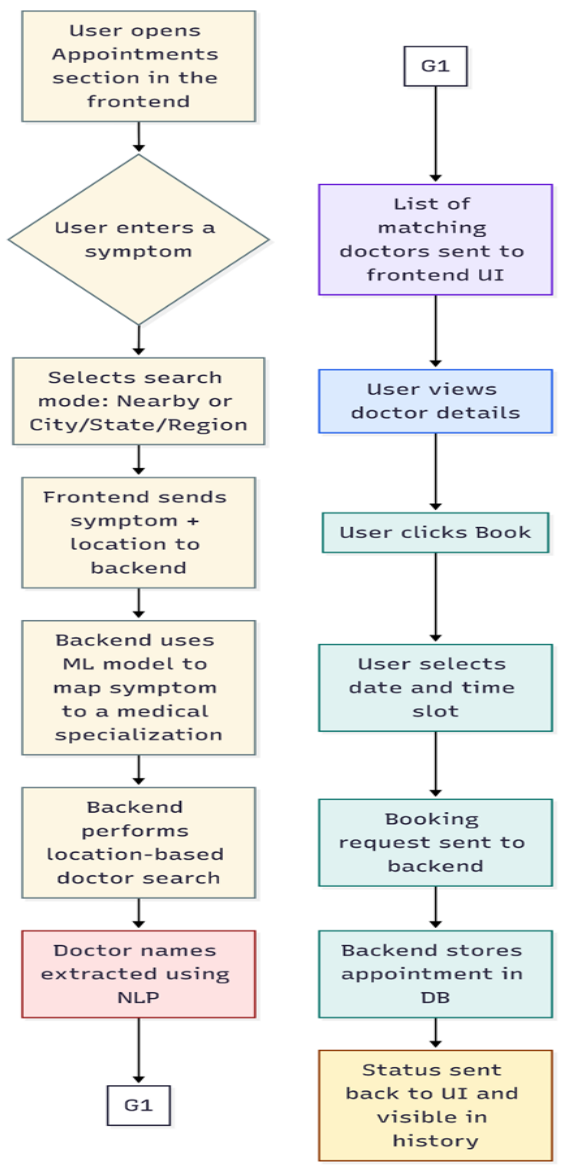


Figure 3: End-to-End Flow of Symptom-Based Appointment Booking System

8. Health Tips (Personalized Recommendations):
Upon clicking "Generate Tip", the system computes a contextual health tip using the latest:

- Mood
- Water intake
- Nutritional summary
- Heart rate
- Steps and sleep (from wearable)

A cosine similarity model matches this profile to the most relevant tip from a dataset of 2,000+ entries.

9. Responsive Design and User Feedback:
Developed with Tailwind CSS, the platform is fully responsive across screen sizes. User experience is enhanced with real-time feedback mechanisms such as toast messages, loading spinners, validation prompts, and visual alerts.

VI. RESULTS AND EVALUATION

This section presents the evaluation results of all six subsystems of the healthcare assistant. Each system was tested on a designated dataset with responses manually reviewed and classified into Good, Partial, or Poor categories, reflecting relevance, factual correctness, and contextual alignment. In addition to textual analysis, a consolidated performance table is provided for cross-system comparison.

1. General Q&A Evaluation:
The BioGPT-based general QA model was tested on 110 health-related queries. Manual evaluation revealed:

- Good: 86 responses (78.18%)
- Partial: 16 responses (14.55%)
- Poor: 8 responses (7.27%)

The relevant match rate (Good + Partial) was 92.73%, showcasing the model's strong grasp of factual relevance and question intent. However, some responses lacked detail or slightly misaligned with nuanced phrasing, suggesting room for further fine-tuning.

2. Health Tip Recommendation Evaluation:
Using 100 vector-based inputs combining mood, nutrition, vitals, and sleep, the cosine similarity model yielded:

- Good: 71 responses (71.0%)
- Partial: 21 responses (21.0%)
- Poor: 8 responses (8.0%)

The overall relevance score was 92.0%, with most tips well-aligned with user conditions. Weak predictions stemmed from vague tip phrasing or mild context mismatches.

3. Profile-Based QA Evaluation:
Based on 100 profile-only inputs, the updated results are:

- Good: 63 responses (63.0%) – *all personalized*
- Partial: 37 responses (37.0%) – *all non-personalized*
- Poor: 0 responses (0.0%)

100% of responses were relevant, and 63% were explicitly personalized, indicating high contextual reliability when handling demographic inputs. The full correction of mislabelled entries enhanced accuracy and consistency.

4. Specialist Mapping Model Evaluation

Evaluated on 3,129 test samples, this classification system (Random Forest + TF-IDF) achieved:

- Overall Accuracy: 84.63%
- Top-3 Accuracy: 95.46%

Strongest results appeared in Dermatology, Ophthalmology, and Orthopaedics (F1 ≥ 0.91). General Physician and Endocrinologist showed weaker recall, warranting class-wise augmentation.

5. Symptom-Only QA Evaluation

On a 100-sample test set with no profile data, manual scoring showed:

- Good: 87 responses (87.0%)
- Partial: 8 responses (8.0%)
- Poor: 5 responses (5.0%)

Most outputs addressed severity, duration, and description effectively, yielding a 95.0% relevant match rate. Poor matches stemmed from generalization or omission of symptom attributes.

6. Symptom + Profile Hybrid QA Evaluation

The hybrid model's 100 outputs were categorized as:

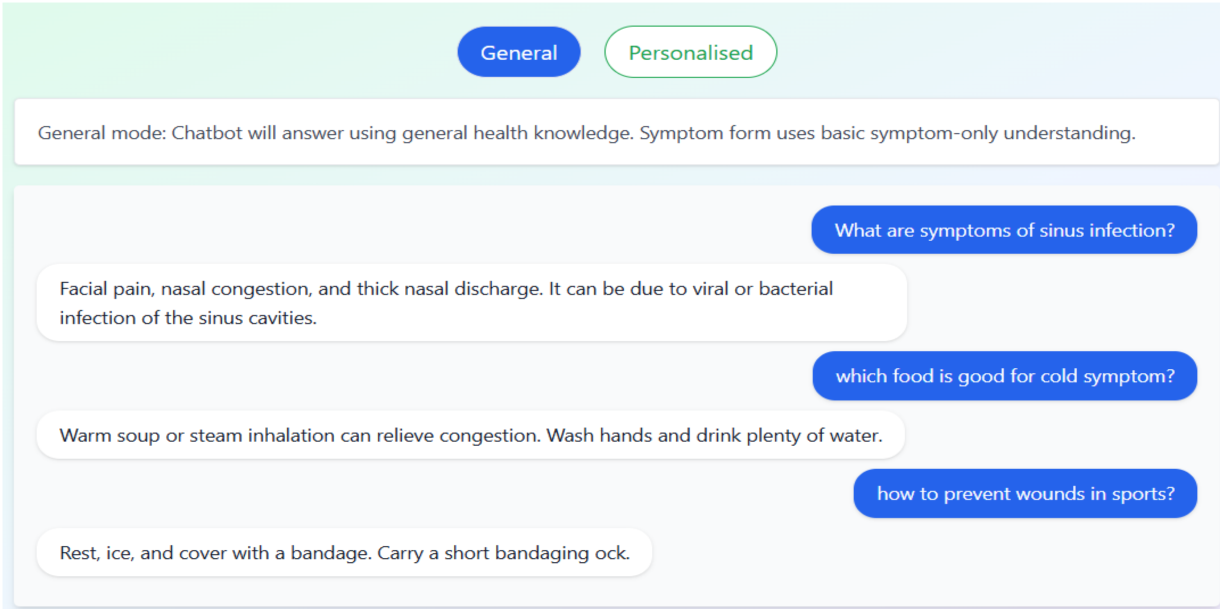
- Good: 51 responses (51.0%)
- Partial: 40 responses (40.0%)
- Poor: 9 responses (9.0%)

Relevance rate was 91.0%, but only 64% of responses explicitly mentioned symptoms. Personalization appeared in 36 responses, of which 33 were relevant. These figures suggest performance gaps in merging both input streams with consistency

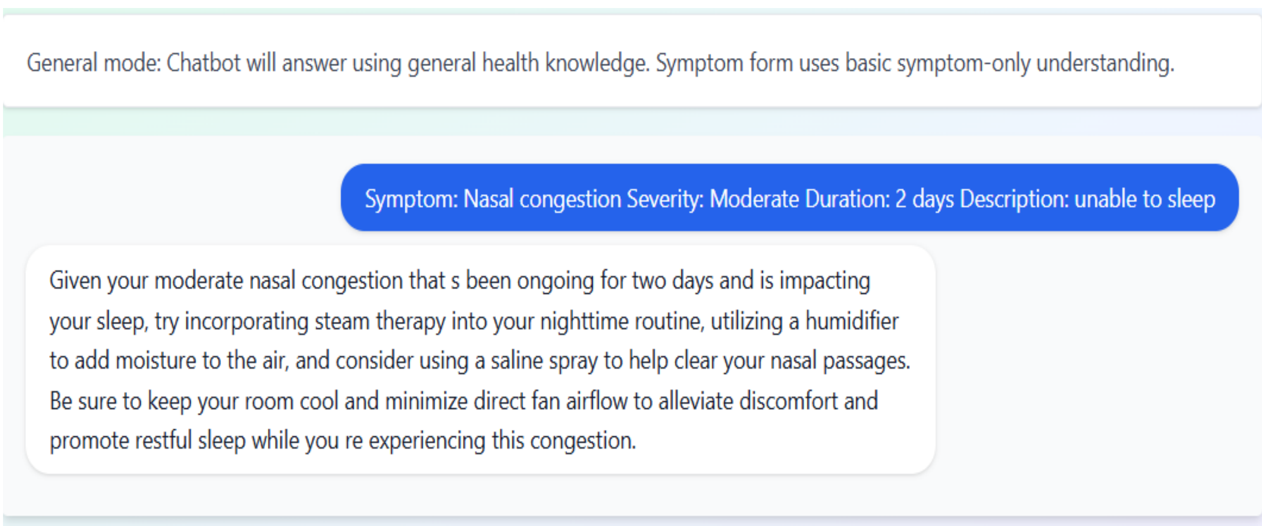
Table 1: Summary of Evaluation Results

Module	Good	Partial	Poor	Good (%)	Partial (%)	Poor (%)	Relevant (G+P) %	Personalised (%)	Non-Personalised (%)
General Q&A	86	16	8	78.18	14.55	7.27	92.73	Not Tracked	Not Tracked
Health Tip Recommendation	71	21	8	71.00	21.00	8.00	92.00	Not Tracked	Not Tracked
Profile-Based QA	98	2	0	98.00	2.00	0.00	100.00	63.00	37.00
Specialist Mapping	—	—	—	—	—	—	Accuracy: 84.63	—	—
Symptom-Only QA	87	8	5	87.00	8.00	5.00	95.00	Not Applicable	Not Applicable
Symptom + Profile Hybrid QA	51	40	9	51.00	40.00	9.00	91.00	36.00	64.00

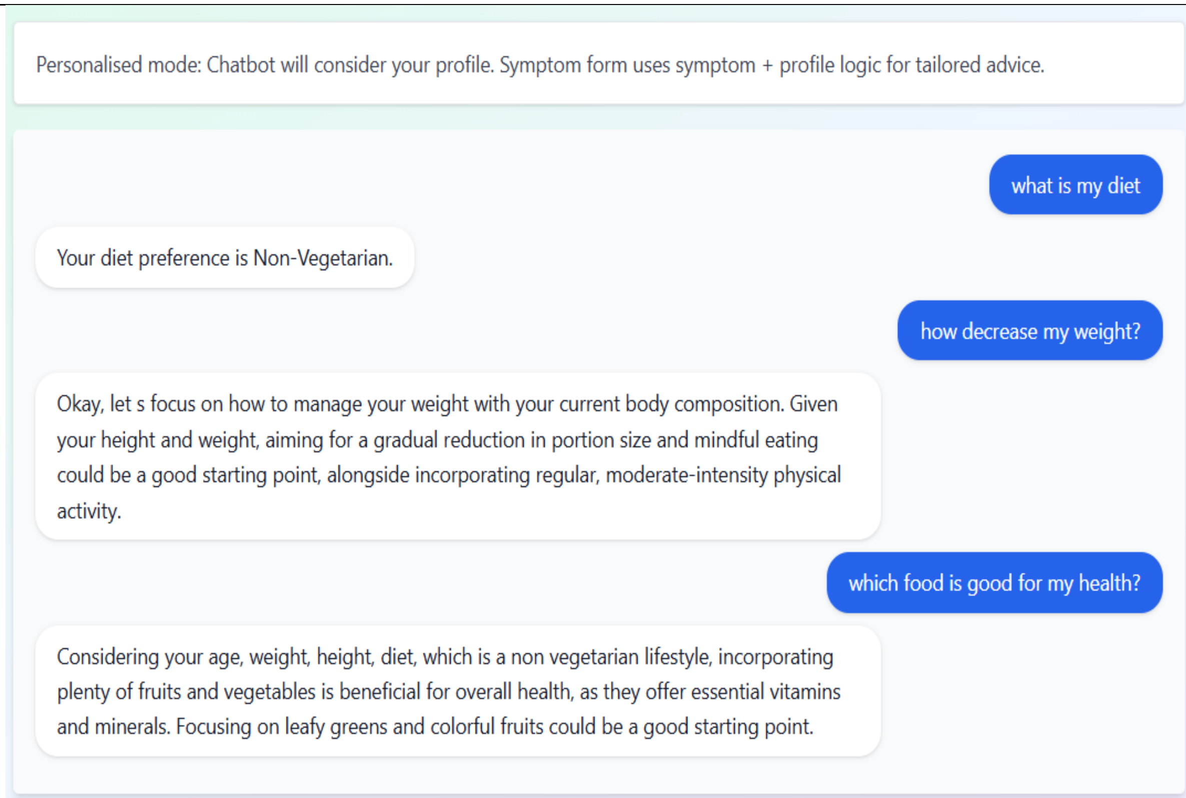
Chatbot:


General
Personalised
General mode: Chatbot will answer using general health knowledge. Symptom form uses basic symptom-only understanding.
What are symptoms of sinus infection?
Facial pain, nasal congestion, and thick nasal discharge. It can be due to viral or bacterial infection of the sinus cavities.
which food is good for cold symptom?
Warm soup or steam inhalation can relieve congestion. Wash hands and drink plenty of water.
how to prevent wounds in sports?
Rest, ice, and cover with a bandage. Carry a short bandaging ock.

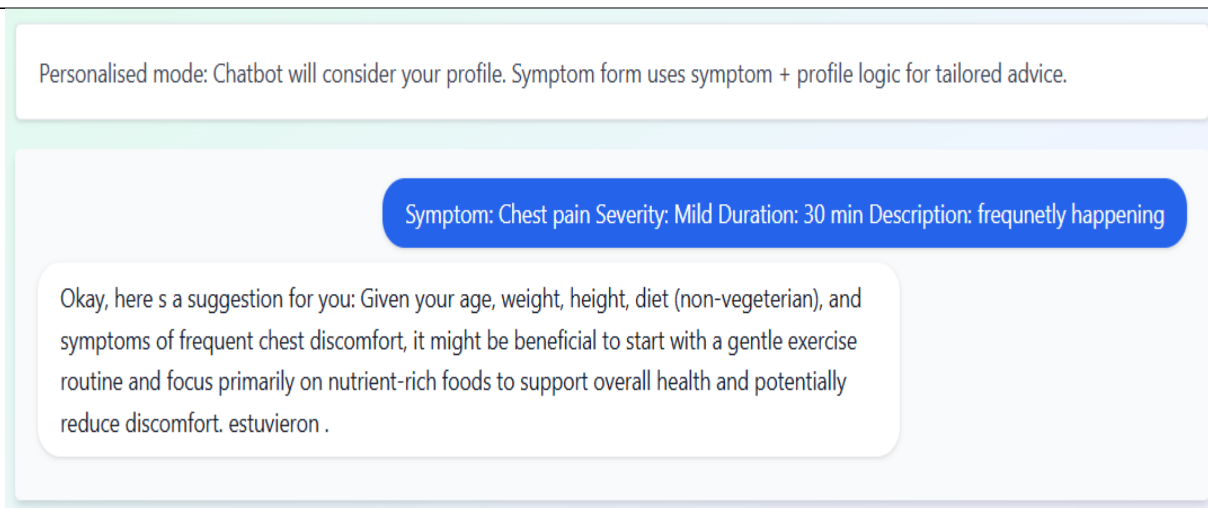
(a)


General mode: Chatbot will answer using general health knowledge. Symptom form uses basic symptom-only understanding.
Symptom: Nasal congestion Severity: Moderate Duration: 2 days Description: unable to sleep
Given your moderate nasal congestion that s been ongoing for two days and is impacting your sleep, try incorporating steam therapy into your nighttime routine, utilizing a humidifier to add moisture to the air, and consider using a saline spray to help clear your nasal passages. Be sure to keep your room cool and minimize direct fan airflow to alleviate discomfort and promote restful sleep while you re experiencing this congestion.

(b)



(c)

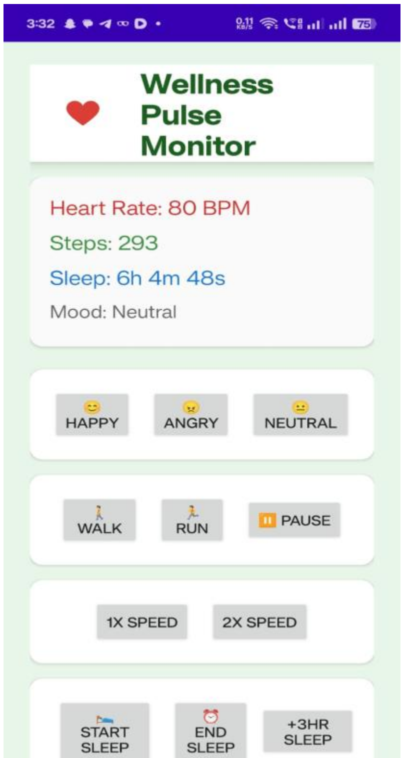


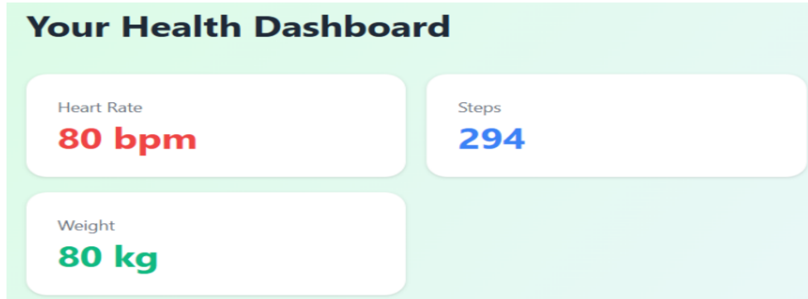
(d)

Figure 4: Chatbot interface screenshots demonstrating user interface flow

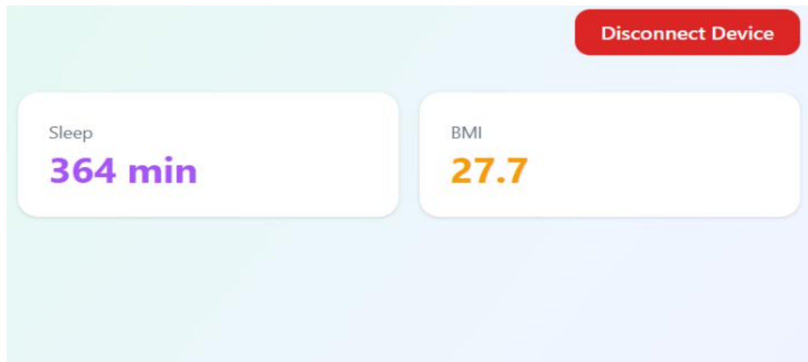
(a) General QA Chatbot, (b) Symptom based Chatbot (c) Personalized QA, (d) Personalized Symptom based Chat-bot

Wearable simulator:





(b)

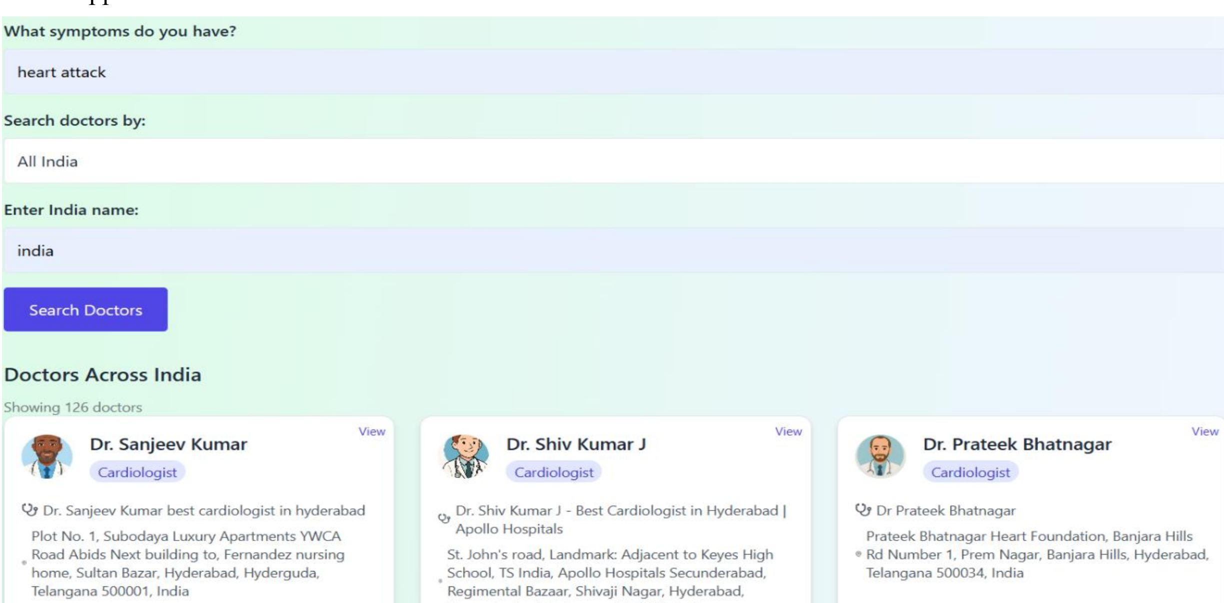


(c)

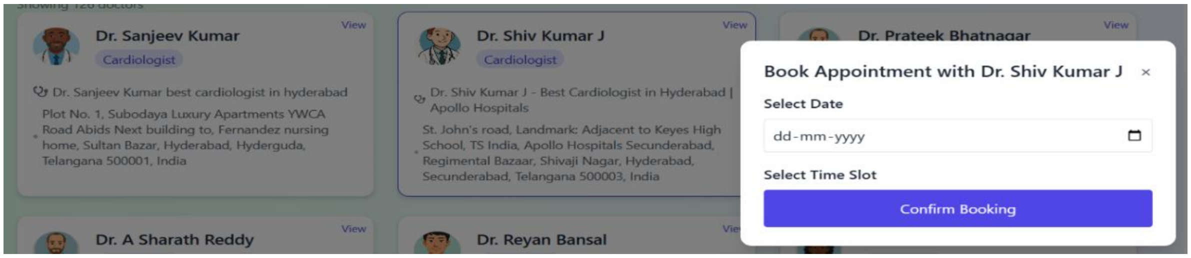
(a)

Figure 5: wearable simulator app data display in dashboard UI with Bluetooth
(a) wearable simulator app, (b) User wearable data with weight display in Dashboard UI (left part), (c) User wearable data with BMI display in dashboard UI (right part)

Doctor Appointment:



(a)



(b)

Figure 6: End-to-End Flow of Symptom-Based Appointment Booking System
(a) User Symptom search and show doctor list with the recommended Specialist, (b) Appointment booking UI

VII.CONCLUSION

The proposed PersonHealth platform successfully demonstrates a scalable, integrated approach to digital healthcare delivery by combining general medical Q&A, personalized and symptom-specific responses, health tracking, and specialist mapping. The use of fine-tuned language models and retrieval-based architectures ensures both contextual relevance and adaptability to individual user profiles. Modules such as wearable simulation, health tip generation, and chatbot personalization further enhance user engagement and proactive care. Evaluation results across various subsystems indicate high accuracy and user relevance, especially in health tip recommendations and specialist prediction. Unlike existing fragmented solutions, PersonHealth offers a unified, intelligent, and interactive platform that bridges the gap between lifestyle data and actionable healthcare insights and Ethical considerations for AI in healthcare were also observed, [6] thereby establishing a robust foundation for further innovation in scalable, personalized digital healthcare systems [1],[2],[3].

REFERENCE

[1] Brown, P., & Jones, M. (2021). *AI and NLP in Healthcare: Trends and Challenges*. Journal of Medical Informatics, 45(3), 201–217.

[2] Smith, R., & Lee, T. (2020). *Chatbot-based Healthcare Assistance: A Review*. International Journal of AI Research, 12(4), 89–105.

[3] Williams, K. (2019). *Machine Learning for Medical Chatbots: Approaches and Outcomes*. Healthcare AI Journal, 8(2), 44–60.

[4] OpenAI. (2023). *GPT Models for Conversational Agents*. Technical Report on AI-driven Chatbots, 67–82.

[5] Patel, S., & Gupta, N. (2022). *Security Considerations for Healthcare Chatbots*. Journal of Cybersecurity in Medicine, 19(1), 31–48.

[6] World Health Organization. (2021). *Ethical Guidelines for AI in Healthcare*. WHO Publications, 1–15.

[7] Kim, Y., & Park, H. (2021). *Context-Aware NLP for Medical Chatbots*. Journal of Computational Medicine, 22(5), 189–204.

[8] Hugging Face. (2025). *BioGPT Model Card*. Retrieved from: https://huggingface.co/microsoft/biogpt

[9] Hugging Face. (2025). *Gemma Model Card*. Retrieved from: https://huggingface.co/google/gemma-3-1b-it

[10] Lewis, P., Perez, E., Piktus, A., et al. (2020). *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. Advances in Neural Information Processing Systems (NeurIPS).

[11] Django Software Foundation. (2025). *Django Documentation (v4.x)*. Retrieved from: https://docs.djangoproject.com

[12] ReactJS. (2024). *React – A JavaScript Library for Building User Interfaces*. Retrieved from: https://react.dev

[13] Johnson, J., Douze, M., & Jégou, H. (2017). Billion-scale similarity search with GPUs. Facebook AI Research.

[14] FAISS White paper. Retrieved from: https://github.com/facebookresearch/faiss