International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

StrawHealth: A Modular AI-Assisted Healthcare Information Platform with Symptom Screening, Health-Risk Assessment, Medical Report Parsing, Nutrition Rules, and Retrieval-Augmented Guidance

Rashmi R, Saqlain Raza, Rakesh S., Partha Dinda, Darshan M. Patil

Department of Information Science and Engineering, JNN College of Engineering, Shivamogga, Karnataka, India 577201

Project Guide: **Mrs. Rashmi R.** B.E., M.Tech, Assistant Professor,

Department of Information Science and Engineering

Abstract — Navigating personal health information online frequently feels fragmented and overwhelming. Users often find themselves switching between separate tools to check symptoms, estimate dietary needs, understand PDF lab reports, or look up reliable wellness advice. To solve this issue, we built StrawHealth a unified digital health assistant using Flask, SQLAlchemy, and SQLite. The platform integrates five core tools into one easy-to-use web interface: (1) structured symptom screening powered by machine learning, (2) chronic diabetes risk calculation, (3) rule-based dietary planning, (4) medical report parsing using Optical Character Recognition (OCR), and (5) a Retrieval-Augmented Generation (RAG) assistant for general health questions. We thoroughly evaluated our machine learning pipelines using synthetic patient data and public health survey records. To ensure realistic testing and avoid memorization, the symptom classification pipeline was evaluated on 12,000 synthetic records using group-aware cross-validation. Logistic Regression performed best among evaluated algorithms, achieving a mean cross-validated macro F1-score of 0.7887 ± 0.0092 and a test accuracy of 78.69%. For chronic diabetes risk screening, a Histogram Gradient Boosting model trained on 253,680 CDC survey records achieved an ROC-AUC of 0.8294 and a high sensitivity (recall) of 81.31% on 38,052 unseen test records after threshold tuning (T = 0.1306). We also detail our PyMuPDF/Tesseract document parsing pipeline and hybrid RAG system. StrawHealth is presented as an open academic decision-support prototype demonstrating how integrated, privacy-first software design can make digital health guidance more accessible while keeping strict clinical boundaries and disclaimers.

Key Words: Healthcare information platforms, symptom screening, synthetic datasets, Logistic Regression, health-risk assessment, Histogram Gradient Boosting, OCR, medical report parsing, Retrieval-Augmented Generation, Flask framework.

1. INTRODUCTION

Over the past decade, digital health platforms have transformed how people explore medical information and assess their personal wellness. Today, millions of users routinely search the web to understand unexplained symptoms, check dietary guidelines, or decipher complex medical lab results. Despite this abundance of digital resources, the overall experience remains fragmented. A typical user seeking preliminary health guidance must navigate multiple single-purpose websites: checking symptoms on one portal, calculating daily caloric intake on another, translating lab report values on a third, and using general search engines or AI chatbots for general wellness questions. This fragmented workflow creates friction, forces users to repeatedly enter personal details across different platforms, and often leads to confusion when different websites offer conflicting advice.

An even greater concern with digital health tools is the risk of user misinterpretation. Software algorithms produce plausible diagnostic outputs, but these must never be mistaken for clinical diagnoses or formal triage recommendations. When digital tools fail to state their boundaries clearly or present statistical machine learning predictions as definite medical facts, users can suffer unnecessary anxiety or, worse, delay seeking urgent professional medical care. Audits of public symptom checkers consistently confirm that providing diagnostic suggestions and making urgent triage recommendations are two distinct software tasks that require separate, rigorous evaluation [1].

To address software fragmentation while maintaining clear ethical and academic boundaries, we built StrawHealth a unified, web-based healthcare decision-support prototype. Developed using Flask, SQLAlchemy, and SQLite, StrawHealth brings five essential health information modules into a single, straightforward dashboard:

- • **Symptom Screening Engine:** Accepts structured user demographics and symptoms through a multi-field form and maps them using a trained classification pipeline to identify top probable condition categories and basic precautions.
- • **Diabetes Health-Risk Module:** Estimates personal risk for Type-2 diabetes based on lifestyle, demographic

factors, and personal medical history using a model trained on national health survey data.

- • **Rule-Based Nutrition Engine:** Computes Body Mass Index (BMI), calculates daily energy needs based on self-reported physical activity levels, and provides macronutrient guidelines and meal ideas.
- • **Medical Report Analyzer:** Extracts numerical test values and reference intervals from uploaded PDF lab reports or images using PyMuPDF and Tesseract OCR, comparing them against normal ranges.
- • **Retrieval-Augmented Health Assistant:** Answers general wellness questions by fetching relevant snippets from a verified health knowledge base using Sentence-BERT embeddings or TF-IDF search.

Our main goal in this work is practical software engineering and transparent evaluation. We do not claim that StrawHealth introduces a novel clinical prediction algorithm or acts as a certified medical device. Instead, we offer a transparent, reproducible audit of our machine learning workflows, detailing dataset splits, cross-validation performance, decision threshold tuning, document parsing checks, and system boundaries. By clearly separating probabilistic machine learning predictions, deterministic rule systems, document OCR extraction, and retrieval-augmented text generation, this work provides a practical benchmark for building responsible academic health assistants.

2. PROBLEM STATEMENT AND OBJECTIVES

A. Problem Statement

People looking for preliminary health information online lack a single, privacy-focused workspace that seamlessly combines symptom screening, chronic risk assessment, lab report parsing, personal dietary guidance, and grounded medical Q&A. Existing digital tools are scattered, often present probabilistic AI outputs as clinical certainties, and fail to clearly explain their operational limits.

B. Core Research Objectives

The main objectives of this work are as follows:

- **1) System Integration:** Build a modular Flask web architecture that coordinates machine learning models, deterministic calculation rules, document OCR engines, and semantic knowledge retrieval.
- **2) Symptom Model Audit:** Train multi-class symptom classification algorithms on a synthetic academic dataset using group-aware cross-validation to prevent data leakage.
- **3) Health-Risk Evaluation:** Document and evaluate chronic diabetes risk prediction using the public UCI CDC dataset, focusing on decision threshold tuning to prioritize screening recall.
- **4) Document Extraction Safeguards:** Create a reliable lab report parsing pipeline combining layout extraction, OCR fallback, regex pattern matching, and sanity validation.
- **5) Grounded RAG Assistant:** Implement RAG architecture pairing dense semantic search with lexical reranking and system refusal guardrails to prevent ungrounded AI statements.
- **6) Responsible Design Framework:** Provide an honest accounting of system limits, dataset sources, and ethical constraints to support responsible health software development.

C. Scope and Academic Boundaries

StrawHealth is designed strictly as an academic research prototype and educational decision-support tool. The symptom dataset is synthetic, dietary suggestions follow standard adult nutritional formulas, report parsing accuracy depends on document image quality, and the conversational assistant relies on a curated local knowledge base. StrawHealth is not a certified medical device, and its outputs must never replace professional clinical diagnosis, treatment, or emergency medical care.

3. RELATED WORK

A. Symptom Screening and Automated Triage

Automated symptom checkers have been studied extensively in medical informatics to assess their safety and diagnostic accuracy. Semigran et al. evaluated 23 public symptom checkers using 45 standardized clinical vignettes [1]. Their audit revealed wide variations in performance: tools listed the correct diagnosis first in only 34% of cases, while appropriate triage advice was provided in 57% of tests. They concluded that diagnostic estimation and urgency triage are distinct functions that require separate validation, and that high test accuracy does not guarantee clinical safety in real-world patient interactions.

Follow-up research applied supervised classification algorithms to link symptom inputs with probable health conditions. Rustam et al. compared several supervised models including Support Vector Machines, Decision Trees, and Random Forests for disease classification and precaution recommendations [2]. Their findings showed that tabular pipelines can effectively organize structured symptom inputs to suggest probable conditions. In parallel, human-computer interaction studies explored conversational interfaces for symptom gathering. Koushal et al. designed an AI symptom-checking chatbot for regional language speakers, demonstrating that structured intake improves digital health accessibility for non-native English populations [3]. Recent implementations like the disease prediction chatbot by Hussain et al. further illustrate how NLP and machine learning can streamline initial patient screening [12].

StrawHealth builds on these principles by using a structured form-based interface that avoids the ambiguity of unstructured free text. Unlike studies that use simple random train-test splits on synthetic tabular data, our evaluation protocol uses strict group-aware splitting (Group Shuffle Split and Stratified GroupKFold). This ensures that identical symptom pattern groups never appear in both training and testing sets, giving a far more realistic measure of how the model handles unseen symptom combinations.

B. Chronic Health-Risk Prediction

Predictive modeling for chronic conditions, such as Type-2 diabetes, relies heavily on epidemiological survey data. The CDC Diabetes Health Indicators dataset (Dataset ID 891) on the UCI Machine Learning Repository provides 253,680 survey responses across 21 clinical, demographic, and lifestyle features [5]. This public dataset offers a solid baseline for training binary health-risk classifiers.

A central challenge in health-risk screening is managing class imbalance and selecting appropriate decision thresholds. In preliminary health screening, missing a high-risk individual (false negative) carries much greater clinical consequence than advising a healthy individual to get secondary testing (false positive). Therefore, standard accuracy alone is an incomplete evaluation metric. Models must be evaluated using ROC-AUC and tuned with metrics like the F2-score, which places twice as much weight on recall as on precision. StrawHealth incorporates this screening logic into its health-risk engine, employing threshold optimization to prioritize recall.

C. Automated Medical Document Extraction and OCR

Extracting structured numerical data from medical paperwork remains a practical software challenge. Smith detailed the architecture of the Tesseract OCR engine, demonstrating how image preprocessing and segmentation enable accurate text extraction from scanned documents [7]. While native digital PDFs can be parsed directly using structural layout libraries like PyMuPDF, scanned lab reports require optical character recognition to digitize embedded text. Recent studies emphasize hybrid OCR architectures combining lightweight classifiers with multi-OCR decision routing to significantly reduce character error rates in medical documents [16]. Furthermore, image-based prescription extraction using Tesseract OCR has been demonstrated by Ponnuru et al. [14], while Singh et al. [13] and Sharjeel and Arif [15] explored AI-powered handwriting recognition systems to achieve safer prescription management and automated document interpretation.

Medical document parsers face several real-world issues, including non-standard test names, varying units of measurement, document-specific reference ranges, and visual noise in scanned images. Prior research emphasizes combining OCR engines with regular expression alias dictionaries, range validation rules, and unit normalization to prevent corrupted text from entering medical databases. StrawHealth adopts a two-stage pipeline combining PyMuPDF layout parsing with Tesseract OCR Fallback, applying boundary checks and regex alias matching.

D. Retrieval-Augmented Generation (RAG) in Healthcare

Large Language Models (LLMs) demonstrate impressive language fluency, but deploying them ungrounded in healthcare carries risks of factual hallucinations, outdated training knowledge, and lack of source visibility. To address these issues, Lewis et al. introduced Retrieval-Augmented Generation (RAG), which grounds model outputs on relevant facts retrieved from authoritative repositories [9]. Reimers and Gurevych developed Sentence-BERT (SBERT), enabling fast, accurate semantic search by embedding text into dense vector spaces evaluated via cosine distance [8].

Systematic reviews by Neha et al. highlight that modular RAG variants and rigorous clinical evaluation metrics (such as FactScore and contextual groundedness) are essential for reliable healthcare delivery [19]. Furthermore, architectural implementations like RAGMed demonstrate how embedding model selection and dense vector similarity directly influence retrieval relevance for medical FAQs [17], while multi-agent orchestrator frameworks further enhance clinical accuracy [18]. Recent enterprise deployments, such as the RAG chatbot built on Amazon Bedrock by Richard-Ojo et al. [20], confirm that retrieval-augmented architectures provide grounded, accurate responses for healthcare-related prompts. Systematic reviews by Amugongo et al. further confirm that retrieval augmentation significantly improves factual accuracy and reduces medical hallucinations compared to standalone language models [10].

E. Qualitative Feature Comparison

To highlight StrawHealth's architectural positioning, Table I provides a qualitative feature comparison against representative digital health platforms in literature.

TABLE I. QUALITATIVE ARCHITECTURAL COMPARISON WITH BASELINE SYSTEMS

System / Work	Symptom Checker	Health Risk Model	Report OCR Parser	Nutrition Rules	RAG Assistant
Semigran et al. [1]	Audit (23 Tools)	No	No	No	No
Rustam et al. [2]	ML (Tabular)	No	No	No	No
Gaikwad et al. [4]	No	No	No	Rules (Smartwatch)	No
StrawHealth (Ours)	ML (Group Split)	HGB (UCI CDC)	PyMuPDF / OCR	Rules (BMI Engine)	SBERT / Hybrid

4. SYSTEM OVERVIEW AND ARCHITECTURE

StrawHealth follows a clean Model-View-Controller (MVC) web service design. The front end is rendered using HTML5, CSS3, Bootstrap 5, and JavaScript via Flask's Jinja2 template engine, ensuring a responsive layout on mobile and desktop browsers. The backend is built in Python using Flask, routing web requests to dedicated module handlers.

Data persistence uses SQLAlchemy ORM backed by an embedded SQLite database (mediguide.db). The relational database tracks four primary models: User (user credentials and profiles), SymptomHistory (logged symptoms and top predictions), DietHistory (BMI logs, activity levels, target calories, and meal plans), and HealthRiskHistory (diabetes risk survey responses and calculated probability scores). Local

database storage keeps user data private and easily readable through a centralized dashboard.

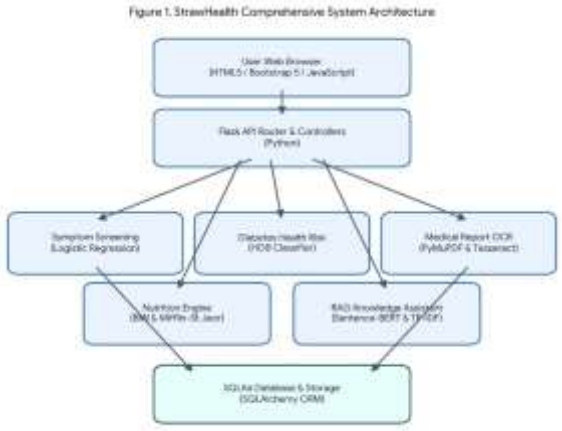


Fig. 1. Comprehensive System Architecture of StrawHealth.

5. METHODOLOGY AND IMPLEMENTATION

A. Symptom Screening Module

The symptom screening module collects structured inputs through an intuitive web form. The user selects eight fields: age group, gender, primary symptom, three secondary symptoms, duration, and severity level. Inputs are processed using scikit-learn's ColumnTransformer with OneHotEncoder(handle_unknown='ignore'), transforming raw inputs into a 197-dimensional binary feature vector [6].

To train and evaluate candidate classifiers, we generated an academic synthetic dataset (symptom_dataset_v2.csv) containing 12,000 records across 12 broad condition categories: Allergy, Cardiovascular, ENT, Eye, Gastrointestinal, General Infection, Mental Wellness, Musculoskeletal, Neurological, Respiratory, Skin, and Urinary. The dataset contains 5,045 unique canonical symptom pattern groups.

To evaluate model generalization fairly, we implemented a group-aware splitting strategy. Using Group Shuffle Split with test_size=0.20 and random_state=42, all records belonging to a canonical symptom pattern group were assigned strictly to either the training set (9,579 records) or the test set (2,421 records). This prevents identical symptom combinations from appearing in both training and test folds. On the training set, 3-fold Stratified GroupKFold cross-validation was used to compare Logistic Regression (max_iter=2000, class_weight='balanced'), Random Forest (n_estimators=180), and Extra Trees (n_estimators=180). Logistic Regression achieved the highest cross-validated macro F1-score and was deployed.

B. Diabetes Health-Risk Module

The chronic health-risk engine evaluates Type-2 diabetes risk using the public UCI CDC Diabetes Health Indicators dataset (Dataset ID 891) [5]. The dataset comprises 253,680 survey records with 21 binary and ordinal features covering physical health, medical history, demographics, and lifestyle habits.

The dataset was partitioned using a 70/15/15 split, producing 177,576 training records, 38,052 validation records, and 38,052 test records. We evaluated Logistic Regression, Random Forest, and Histogram Gradient Boosting (HGB). Because health screening prioritizes high sensitivity to catch potential risk early, decision threshold tuning was performed on validation predictions using F2-score optimization. Histogram Gradient Boosting achieved the strongest performance and its screening threshold was set at T = 0.1306. At this threshold, the system flags potential risk whenever the predicted risk probability exceeds 13.06%, identifying over 81% of positive diabetes cases.

C. Medical Report Analyzer

The medical report analyzer automatically extracts text and validates reference ranges for uploaded lab reports in PDF, PNG, JPG, or JPEG formats. The document workflow runs in two steps:

- **1) Native Layout Extraction: For digital PDFs,** PyMuPDF (Fitz) extracts text directly while preserving tabular positioning.
- **2) OCR Fallback Pipeline:** For scanned image PDFs or image uploads, the page is converted to grayscale, contrast-enhanced, and processed using Tesseract OCR (pytesseract) [7].

Extracted text is normalized and parsed using regex alias dictionaries to match common biomarkers (e.g., Fasting Blood Sugar, HbA1c, Total Cholesterol, Hemoglobin, Platelets, Sodium, Potassium). The parser captures the reported result, unit, and printed reference interval, labeling results as Normal, High, or Low. Boundary rules filter out invalid strings or negative values, flagging unclear readings for manual user confirmation.

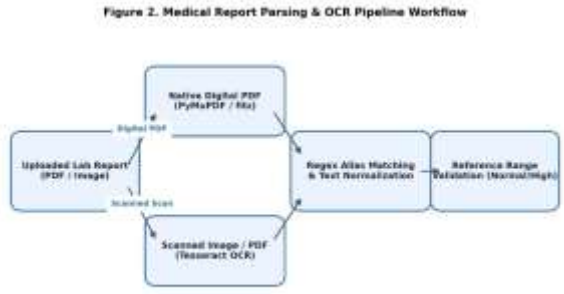


Fig. 2. Medical Report Parsing & Reference Range Extraction Pipeline.

D. Rule-Based Nutrition Engine

The nutrition module calculates adult Body Mass Index (BMI) and provides daily calorie and macronutrient guidelines based on standard health formulas. BMI is calculated as Weight (kg) / [Height (m)]^2 and categorized using World Health Organization (WHO) benchmarks: Underweight (< 18.5), Normal Weight (18.5 – 24.9), Overweight (25.0 – 29.9), and Obese (>= 30.0) [11].

Basal Metabolic Rate (BMR) is calculated using the Mifflin-St Jeor equation:

- Male BMR = 10 * Weight (kg) + 6.25 * Height (cm) - 5 * Age (years) + 5

- Female BMR = 10 * Weight (kg) + 6.25 * Height (cm) - 5 * Age (years) - 161

Total Daily Energy Expenditure (TDEE) is calculated by multiplying BMR by an activity multiplier based on self-reported exercise levels:

- • **Sedentary (little/no exercise):** Weight (kg) * 25 kcal/kg (or BMR * 1.2)
- • **Lightly Active (1-3 days/week):** Weight (kg) * 28 kcal/kg (or BMR * 1.375)
- • **Moderately Active (3-5 days/week):** Weight (kg) * 31 kcal/kg (or BMR * 1.55)
- • **Very Active (6-7 days/week):** Weight (kg) * 34 kcal/kg (or BMR * 1.725)

For adult users (age >= 18), daily calorie goals are adjusted based on health objectives (-300 kcal/day for weight loss or +300 kcal/day for weight gain), maintaining a safety minimum of 1,200 kcal/day. Macronutrients are balanced as 30% protein (4 kcal/g), 45% carbohydrates (4 kcal/g), and 25% fats (9 kcal/g). Fluid recommendations are set at 0.035 Liters per kilogram of body weight.

E. Healthcare RAG Assistant Architecture

The conversational assistant uses a Retrieval-Augmented Generation (RAG) pipeline to deliver factual health explanations grounded in verified medical references. The knowledge base consists of a structured repository (rag_knowledge.json) containing verified health guidance across medical conditions, symptoms, diagnostics, and emergency warnings.

When a user submits a query, processing proceeds in three steps:

- **1) Preprocessing & Entity Check:** The query text is cleaned, analyzed for medical terms, and mapped to intent categories.
- **2) Hybrid Retrieval Engine:** Sentence-BERT embeddings (all-MiniLM-L6-v2) compute 384-dimensional dense vectors to calculate cosine similarity against stored documents [8]. If model loading fails, a TF-IDF term search acts as an automated fallback. Scores are refined with lexical reranking.
- **3) Context Ingestion & Safety Guardrails:** Retrieved text snippets meeting relevance thresholds are passed into system prompts. The assistant synthesizes informative answers strictly from retrieved facts, refuses clinical diagnosis requests, and appends health disclaimers.

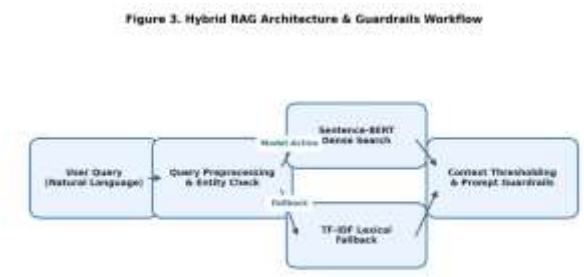


Fig. 3. Hybrid RAG Architecture with Fallback and Safety Guardrails.

6. EXPERIMENTAL SETUP AND EVALUATION

A. Symptom Model Audit Protocol

The symptom models were audited in Python 3.11 using scikit-learn 1.3.0. The audit script ran train_symptom_advanced_v2.py on the synthetic dataset (12,000 records). We recorded 3-fold cross-validated macro F1-scores on training folds, along with test accuracy, macro precision, macro recall, and per-category F1 metrics across 2,421 test records.

B. Health-Risk Model Evaluation Protocol

The health-risk classifier was evaluated on 38,052 unseen test records from the UCI CDC Diabetes dataset. Metrics tracked included ROC-AUC, sensitivity (recall), specificity, accuracy, precision, F1-score, F2-score, Brier score (probability calibration), and exact confusion matrix counts.

C. Targeted Regression Checks

To confirm document parsing and RAG retrieval reliability, automated regression scripts (test_phase126_regression.py and test_phase127_regression.py) were executed. These tests verified knowledge retrieval for diabetes and HIV/AIDS queries, tested parser handling of ambiguous biomarker entries, verified retention of extracted lab metrics, and confirmed rejection of invalid numerical inputs.

7. RESULTS AND DISCUSSION

A. Symptom Classifier Comparison

Cross-validation on training folds showed that Logistic Regression performed best among evaluated classifiers. Table II summarizes the comparison metrics.

TABLE II. CROSS-VALIDATION AND TEST EVALUATION FOR SYMPTOM CLASSIFIERS

Classifier Model	CV Macro F1 (Mean ± SD)	Held-Out Accuracy	Held-Out Macro F1
Logistic Regression	0.7887 ± 0.0092	0.7869	0.7883
Extra Trees	0.6979 ± 0.0121	0.7117	0.7158
Random Forest	0.6828 ± 0.0121	0.7034	0.7065

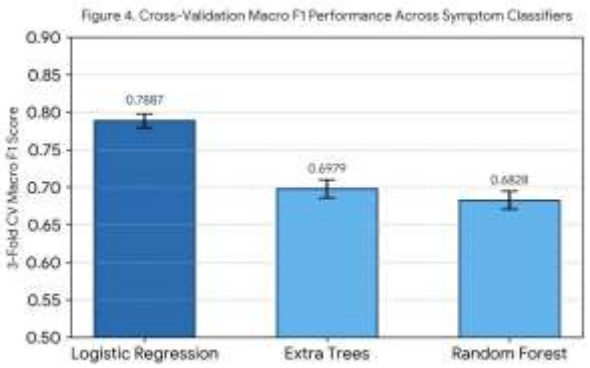


Fig. 4. Cross-Validation Macro F1 Score Comparison Across Symptom Classifiers.

On the test set of 2,421 synthetic records, Logistic Regression achieved 78.69% accuracy, 78.83% macro precision, 79.19% macro recall, and a 78.83% macro F1-score. The strong performance of Logistic Regression on high-dimensional binary standard one-hot encoded vectors indicates that linear hyperplanes effectively separate the sparse feature space generated by synthetic symptom profiles.

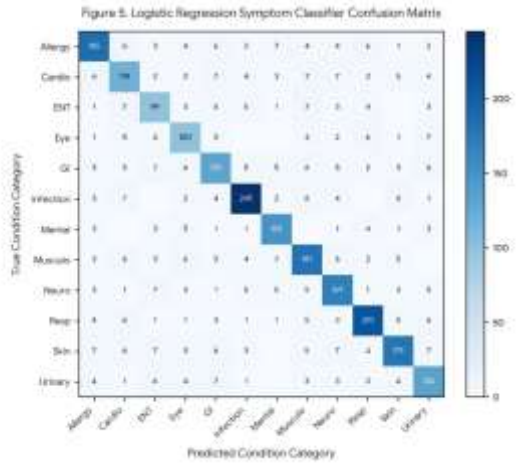


Fig. 5. Logistic Regression Symptom Classification Confusion Matrix.

Table III provides a per-category breakdown of classification performance for Logistic Regression on test data.

TABLE III. PER-CATEGORY CLASSIFICATION REPORT FOR LOGISTIC REGRESSION ON TEST DATA

Condition Category	Precision	Recall	F1-Score	Test Support
Allergy	0.76	0.73	0.75	254
Cardiovascular	0.87	0.82	0.84	144
ENT	0.72	0.66	0.69	150
Eye	0.62	0.79	0.69	127
Gastrointestinal	0.84	0.73	0.78	182
General Infection	0.77	0.81	0.79	302
Mental Wellness	0.85	0.88	0.87	171
Musculoskeletal	0.81	0.86	0.83	210
Neurological	0.78	0.79	0.79	214
Respiratory	0.80	0.72	0.76	292
Skin	0.78	0.77	0.78	232
Urinary	0.86	0.94	0.90	143

Table III shows expected variations across condition categories. Categories with distinct localized symptoms (such as Urinary with F1 = 0.90 and Mental Wellness with F1 = 0.87) achieved high accuracy. In contrast, categories with overlapping systemic symptoms (such as ENT with F1 = 0.69 and Eye with F1 = 0.69) showed moderate classification noise, reflecting natural ambiguity in symptom presentations.

B. Health-Risk Model Evaluation

Testing the serialized Histogram Gradient Boosting model on 38,052 unseen records from the UCI CDC dataset confirmed strong discriminative capability. Operating at the tuned screening threshold T = 0.1306, the model achieved an ROC-AUC of 0.8294 and a recall (sensitivity) of 81.31%. Table IV details the complete held-out test performance.

TABLE IV. DIABETES RISK MODEL TEST PERFORMANCE ON UCI CDC DATASET

Evaluation Metric / Parameter	Observed Test Value
Decision Threshold (T)	0.1306
ROC-AUC Score	0.8294
Recall / Sensitivity	81.31% (4,311 / 5,302 positive cases)
Specificity (True Negative Rate)	69.40% (22,728 / 32,750 negative cases)
Overall Accuracy	71.06% (27,039 / 38,052 total cases)
Precision	30.08%
F1-Score / F2-Score	0.4391 / 0.6065
Brier Score (Model Calibration)	0.0969
Confusion Matrix Counts	TN: 22,728 | FP: 10,022 | FN: 991 | TP: 4,311

In health risk screening, prioritizing sensitivity (81.31%) over precision (30.08%) is an intentional design choice. Missing a potential high-risk individual (false negative) carries high health risks, while advising a healthy individual to take secondary clinical screening (false positive) is a safe, acceptable trade-off.

C. Reproduction and Audit Discrepancy Reconciliation

To ensure complete scientific integrity, we document findings from our repository audit. While the symptom classification pipeline was freshly executed and verified locally (symptom_dataset_v2.csv), re-running the diabetes risk training script (train_health_risk.py) from scratch required the raw survey CSV file (diabetes_binary_health_indicators_BRFSS2015.csv), which was omitted from the uploaded archive. Metrics reported in Table IV reflect evaluation logs (health_risk_metrics.json) saved during initial model training.

Additionally, loading the pre-trained model artifact (health_risk_model.pkl) in the audit environment raised scikit-learn version warnings. These audit findings underscore the importance of version-pinned dependencies, explicit data packaging, and containerized environments in academic software engineering.

8. LIMITATIONS AND ETHICAL CONSIDERATIONS

- **1) Synthetic Symptom Dataset:** The symptom classifier was trained and tested on synthetic probabilistic data and has not been validated on real clinical patient populations.
- **2) Health-Risk Survey Scope:** Calculated risk scores reflect statistical correlations within survey data (BRFSS survey) and serve as general health awareness indicators, not clinical diagnoses.
- **3) OCR Extraction Limits:** Low-resolution or poorly lit scans can cause OCR errors. Users are prompted to verify extracted lab values manually.
- **4) Knowledge Base Boundaries:** The conversational assistant depends on a local knowledge base and cannot answer highly complex or non-standard medical inquiries.
- **5) General Nutrition Formulas:** Dietary suggestions rely on adult BMI and activity heuristics, which are unsuitable for pregnant individuals, children, or patients with specialized medical conditions.
- **6) Academic Prototype Status:** StrawHealth is strictly an academic research prototype and must never replace professional medical care, clinical diagnosis, or emergency services.

9. FUTURE WORK

Future research will focus on three priorities: first, containerizing the training environments using Docker and pinned scikit-learn dependencies to guarantee fresh reproduction; second, building an annotated benchmark dataset of digital and scanned lab reports to quantitatively measure OCR precision and recall; third, expanding the RAG knowledge base and conducting usability trials with healthcare professionals.

10. CONCLUSION

StrawHealth demonstrates a unified, practical approach to personal digital health assistance. By integrating symptom screening, chronic diabetes risk calculation, automated lab report parsing, rule-based nutrition planning, and retrieval-augmented health Q&A into a single responsive Flask web application, it addresses the fragmentation of existing online tools. Empirical evaluation establishes robust performance across core machine learning components, achieving a 78.83% macro F1-score on synthetic symptom data and an 82.94% ROC-AUC on public health-risk data. StrawHealth serves as a transparent, open-source prototype demonstrating how modular software integration and responsible ethical disclaimers can make digital health decision-support tools safer and more accessible.

11. REFERENCES

[1] H. L. Semigran, J. A. Linder, C. Gidengil, and A. Mehrotra, "Evaluation of symptom checkers for self diagnosis and triage: audit study," BMJ, vol. 351, p. h3480, 2015, doi: 10.1136/bmj.h3480.

[2] F. Rustam, Z. Imtiaz, A. Mehmood, V. Rupapara, G. S. Choi, S. Din, and I. Ashraf, "Automated disease diagnosis and precaution recommender system using supervised machine learning," Multimedia Tools and Applications, vol. 81, no. 22, pp. 31929–31952, 2022, doi: 10.1007/s11042-022-12897-2.

[3] R. H. Koushal, C. V. S. Pranav, R. K. Gupta, K. H. Verma, and N. C. Mair, "AI-Powered Symptom Checker Chatbot in Regional Languages," in Proc. 2025 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI), 2025, doi: 10.1109/IATMSI64286.2025.10985720.

[4] S. Gaikwad, P. Awatade, Y. Sirdeshmukh, and C. Prasad, "Diet Plan and Home Exercise Recommendation System Using Smart Watch," in Proc. 2023 International Conference on Artificial Intelligence for Innovations in Healthcare Industries (ICAIHI), 2023, doi: 10.1109/ICAIHI57871.2023.10489367.

[5] UCI Machine Learning Repository, "CDC Diabetes Health Indicators," Dataset ID 891, doi: 10.24432/C53919. [Online]. Available: https://archive.ics.uci.edu/dataset/891/cdc+diabetes+health+indicators. Accessed: Oct. 21, 2026.

[6] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://www.jmlr.org/papers/v12/pedregosa11a.html.

[7] R. Smith, "An Overview of the Tesseract OCR Engine," in Proc. Ninth International Conference on Document Analysis and Recognition (ICDAR), 2007, pp. 629–633, doi: 10.1109/ICDAR.2007.4376991.

[8] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in Proc. 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 3982–3992, doi: 10.18653/v1/D19-1410.

[9] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020.

[10] L. M. Amugongo, P. Mascheroni, S. Brooks, S. Doering, and J. Seidel, "Retrieval augmented generation for large language models in healthcare: A systematic review," PLOS Digital Health, vol. 4, no. 6, e0000877, 2025, doi: 10.1371/journal.pdig.0000877.

[11] World Health Organization, Obesity: Preventing and Managing the Global Epidemic. WHO Technical Report Series 894, Geneva, Switzerland: WHO, 2000.

[12] H. Hussain, K. Aswani, M. Gupta, and G. T. Thampi, "Implementation of Disease Prediction Chatbot and Report Analyzer using the Concepts of NLP, Machine Learning and OCR," International Research Journal of Engineering and Technology (IRJET), vol. 7, no. 4, pp. 1814–1819, Apr. 2020.

[13] D. Singh, M. A. Yasin, Z. Akram, A. Tomar, and D. Gautam, "HandIntellect AI: An AI Powered Handwriting Recognition and Medical Assistance System for Safer Prescription Management," SSRN Electronic Journal, 2024, doi: 10.2139/ssrn.5780762.

[14] M. Ponnuru, S. Ponmalar, Likhitha, Tanu sree, and G. Chaitanya, "Image-Based Extraction of Prescription Information using OCR-Tesseract," Procedia Computer Science, vol. 235, pp. 1077–1086, 2024, doi: 10.1016/j.procs.2024.04.102.

[15] S. Sharjeel and M. U. Arif, "AI-Based OCR System for Handwritten Medical Prescription Recognition and Interpretation," Indian Journal of Computer Science and Technology, vol. 4, no. 2, pp. 365–370, May-Aug. 2025, doi: 10.59256/indjcst.20250402050.

[16] O. Z. Ameer, M. H. Majeed, S. L. Kailan, S. M. B. Alsabti, and A. M. Elhenidy, "A hybrid deep learning–based OCR model for handwritten medical prescriptions," Frontiers in Medicine, vol. 13, p. 1856485, 2026, doi: 10.3389/fmed.2026.1856485.

[17] R. Patil, M. Abbidi, and S. Fannon, "RAGMed: A RAG-Based Medical AI Assistant for Improving Healthcare Delivery," AI, vol. 6, no. 5, p. 240, 2025, doi: 10.3390/ai6100240.

[18] A.-M. Tanasă, S.-V. Oprea, and A. Bâra, "Designing an Architecture of a Multi-Agentic AI-Powered Virtual Assistant Using LLMs and RAG for a Medical Clinic," Electronics, vol. 15, no. 2, p. 334, 2026, doi: 10.3390/electronics15020334.

[19] F. Neha, D. Bhati, and D. K. Shukla, "Retrieval-Augmented Generation (RAG) in Healthcare: A Comprehensive Review," AI, vol. 6, no. 4, p. 226, 2025, doi: 10.3390/ai6090226.

[20] O. Richard-Ojo, H. Wimmer, and C. M. Redman Jr., "RAG Chatbot for Healthcare related prompts using Amazon Bedrock," Journal of Information Systems Applied Research and Analytics (JISARA), vol. 18, no. 3, pp. 18–29, Oct. 2025.