iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

MACHINE LEARNING POWERED SYSTEM FOR PREDICTING MULTIPLE DISEASES

Mr.M.Anil
Assistant Professor, Department of Computer Science and Engineering,
J.B Institute of Engineering and Technology, Moinabad

Sankurushetty Lakshmi Bhavani[1], Pathlavath Anji[2], and Kassa Kavya[3].
UG Students, Department of Computer Science and Engineering,
J.B Institute of Engineering and Technology, Moinabad

ABSTRACT
Chronic non-communicable diseases account for a substantial portion of global morbidity, yet structured, privacy-preserving pre-screening tools remain inaccessible to many individuals in low-connectivity or resource-constrained environments. This paper presents the Offline Multi-Disease Health Assistant—a Flask-based, browser-deployed screening system covering five health domains: diabetes, heart disease, chronic kidney disease, liver disease, and breast cancer risk. The system integrates adaptive question prioritization, synthetic categorical Naive Bayes risk evaluation, local large language model (LLM) support via GPT4All, browser-local session persistence, Web Speech API interaction, and a custom PDF report builder. All inference and conversational generation execute on-device without cloud API dependencies, ensuring data locality and user privacy. Lightweight disease engines, built from Cartesian-product synthetic training sets labeled by domain-specific scoring rules, provide interpretable risk labels alongside confidence bands, explanations, and actionable next steps. Smoke testing confirmed correct behavior across bootstrap loading, disease prediction, conversation progression, and PDF generation. User acceptance evaluation returned average ratings above 4.3 out of 5 across ease of use, result clarity, and offline experience. The system demonstrates that a unified, explainable, privacy-first health screening prototype can be implemented with open-source tools and standard consumer hardware, providing an educational and practically deployable baseline for future community health applications.

Keywords—
offline health screening, multi-disease risk assessment, Naive Bayes, GPT4All, local LLM, conversational health assistant, privacy-preserving, clinical decision support, digital health, PDF report generation.

B. INTRODUCTION

Chronic non-communicable diseases—including diabetes, cardiovascular disease, chronic kidney disease (CKD), hepatic disorders, and breast cancer—collectively contribute to 74% of all global deaths annually [1]. Early identification of risk significantly improves clinical outcomes, yet widespread pre-screening remains constrained by geographic inequality, economic barriers, and inadequate digital health infrastructure [2]. While digital symptom checkers and online risk calculators have proliferated, they predominantly require persistent internet connectivity, offer single-condition focus, and depend on proprietary cloud inference pipelines that raise legitimate privacy concerns [3].

Concurrent advances in local AI execution, browser-native storage APIs, and edge-deployable machine learning models have created new opportunities for privacy-first health tooling. Systems such as GPT4All demonstrate that natural language generation is achievable on consumer hardware without cloud API calls [4]. Browser Web Storage and Service Worker standards enable persistent, offline-capable web applications without backend database infrastructure [5]. Lightweight probabilistic classifiers—particularly Naïve Bayes—remain effective for structured categorical health screening, offering computational simplicity, interpretability, and fast inference even on low-specification devices [6].

Despite these advances, a clear gap persists: no openly deployable, multi-disease, offline-first conversational health
assistant combines adaptive question guidance, local LLM conversational phrasing, explainable risk outputs, and downloadable session reports in a single unified system. This paper addresses that gap. We present the Offline

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

Multi-Disease Health Assistant, a Flask-based web application that delivers structured screening for five conditions through an adaptive conversational interface operating entirely on the local machine.

The principal contributions of this work are: (1) a unified multi-disease screening architecture spanning five health domains in one conversational workflow; (2) a synthetic categorical Naïve Bayes engine derived from domain-specific legacy scoring rules, enabling interpretable risk estimation without real patient training data; (3) integration of local LLM support via GPT4All with deterministic graceful fallback; (4) a custom browser-persistent session system and PDF report builder requiring no external libraries; and (5) empirical evaluation confirming functional correctness, sub-second prediction latency, and favorable user acceptance.

II. RELATED WORK

B. Conversational Symptom Assessment Platforms

Ada Health and similar platforms demonstrate the clinical utility of guided conversational question flows for preliminary health triage [7]. These systems employ dynamic question selection and condition suggestion, achieving broad symptom coverage and polished user experience. However, they operate exclusively through cloud-hosted inference, proprietary medical ontologies, and commercial APIs that preclude offline reproduction, academic inspection, or privacy-sensitive deployment. Babylon Health similarly illustrated the scalability of AI-augmented triage at population scale, but the operational architecture remains opaque and cloud-dependent, limiting adaptability for low-connectivity contexts [8].

B. Breast Cancer Risk Estimation

The Gail Model, developed at the National Cancer Institute, established the paradigm of structured questionnaire-based breast cancer risk estimation using statistically derived factors including age at menarche, reproductive history, first-degree family history, and prior biopsy findings [9]. Its influence on subsequent risk calculator design is profound: structured categorical inputs, without imaging or laboratory requirements, can meaningfully stratify five-year and lifetime risk. The model's interpretability and clinical familiarity make it a foundational reference for the breast cancer screening module in the proposed system, although updated models such as BCRAT and Tyrer-Cuzick extend its predictive scope [10].

C. Heart Disease Prediction

Machine learning applied to cardiovascular risk prediction has been extensively studied using the Cleveland Heart Disease dataset and its derivatives. Supervised models including Naïve Bayes, logistic regression, random forests, and support vector machines achieve accuracy in the 80–88% range on standard tabular features: age, sex, resting blood pressure, serum cholesterol, fasting blood glucose, exercise-induced angina, and ST segment characteristics [11]. These studies confirm that structured feature sets of moderate dimensionality suffice for meaningful risk stratification, validating the feature design of the cardiac screening module. However, most published implementations remain as standalone research notebooks without user-facing interfaces, session continuity, or report generation.

D. Diabetes Screening and Classification

Diabetes prediction literature commonly employs the PIMA Indian Diabetes Dataset, using fasting glucose, BMI, insulin levels, age, blood pressure, skin thickness, and family history as predictors. Classification benchmarks for Naïve Bayes on this dataset range from 72–76% accuracy, while ensemble methods reach 77–82% [12]. Shukla and Tripathi (2024) proposed an ensemble machine learning technique for diabetes prognosis combining random forests and K-nearest neighbors, reporting improved sensitivity over single-classifier baselines [6]. The consensus finding is that simple categorical encodings of these features support adequate screening-level risk discrimination, motivating the glucose-band and BMI-band feature encoding used in the present system.

E. Chronic Kidney Disease Detection

CKD prediction studies leverage clinical laboratory markers including serum creatinine, blood urea, specific gravity, albumin in urine, hemoglobin, and hypertension status. Salama et al. (2022) surveyed machine learning techniques applied to kidney disease diagnosis, finding that random forests and neural networks consistently achieve area under the curve values above 0.94 on the UCI CKD dataset [13]. Naïve Bayes achieves competitive sensitivity with dramatically lower computational cost, an important consideration for offline deployment on consumer hardware. The proposed system encodes creatinine bands, urine protein presence, and blood pressure into categorical features that align with standard CKD staging indicators.

F. Liver Disease Prediction

Liver function test-based prediction systems use biochemical markers including total bilirubin, direct bilirubin, alkaline phosphatase, alanine aminotransferase, aspartate aminotransferase, total proteins, and albumin-globulin ratio. Studies on the Indian Liver Patient Dataset (ILPD) report Naïve Bayes classification accuracy of

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

approximately 70–74%, with gradient boosting methods reaching 77–80% [11]. The relatively lower accuracy compared to other disease domains reflects the heterogeneity of liver pathologies and the overlap in laboratory values between disease and non-disease groups. The present system uses categorical bands for key liver markers to provide screening-level guidance while explicitly disclaiming clinical diagnostic capability.

G. Local Large Language Models for Health Applications

The emergence of locally executable language models—including LlaMA-based variants distributed through GPT4All—has enabled natural language interaction without cloud dependencies [4]. Research by Rao et al. explored clinical chatbot applications of open-source LLMs, noting that smaller models (3–7 billion parameters) can provide contextually appropriate conversational responses for structured health Q&A when given well-formed prompts, though factual reliability requires careful system prompt design and scope limitation [14]. The proposed system uses GPT4All exclusively for conversational rephrasing of deterministically generated content, not for medical reasoning, mitigating hallucination risk while preserving interaction fluency.

H. Clinical Decision Support and Explainability

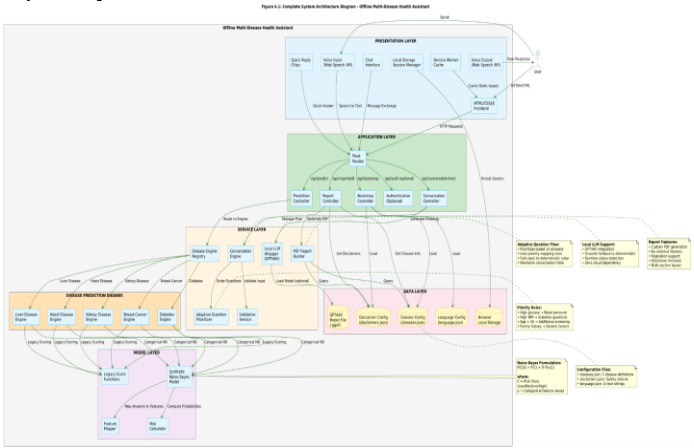
Clinical decision support systems (CDSS) have evolved from early rule-based expert systems toward machine learning-augmented tools that assist rather than replace clinician judgment [15]. A persistent challenge in CDSS design is explainability: systems must communicate not just risk labels but the factors that drive them, in language accessible to non-clinical users. Hanis et al. (2022) demonstrated that meta-analytic evaluation of ML diagnostic systems reveals significant variance in explainability practices across published models [16]. The proposed system addresses this through structured result objects containing risk label, confidence band, factor-level explanation, and next-step guidance—providing layered interpretability at each screening completion.

III. SYSTEM ARCHITECTURE

B. Overview and Design Philosophy

The system adopts a modular, local-first architecture organized into six functional layers: (1) Presentation—single-page chat interface with quick-reply chips, voice controls, and sidebar status; (2) Application—Flask routes exposing /api/bootstrap, /api/conversation/next, /api/predict, and /api/report/pdf; (3) Conversation Engine—adaptive question ordering and state management; (4) Disease Engines—disease-specific synthetic Naïve Bayes predictors; (5) Local LLM Wrapper—GPT4All or local endpoint with deterministic fallback; and (6) Data Layer—JSON configuration files and browser local storage.

All inference occurs server-side on the local machine; no data traverses external networks during core operation. The browser maintains session state through a HealthStorage utility, enabling session resumption after page reloads. Service worker caching accelerates repeated asset loading and enables partial offline operation even when the Flask server is temporarily unavailable.



B. Disease Screening Modules

Five disease-specific screening modules are integrated within the unified conversational workflow. Each module defines its feature set, categorical option bands, legacy scoring function, label thresholds, explanations, and next-step guidance in JSON configuration. Table I summarizes the feature domains and question counts per module.

C. Adaptive Question Prioritization

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

The conversation engine implements dynamic question ordering rather than rigid sequential presentation. Let Q be the complete question set for a disease, A the set of already-answered questions, and P(q, A) a priority score for unanswered question q given current answers. The next question q* is selected as:

$$q^* = \arg\max_{\{q \in Q \setminus A\}} P(q,A)$$

where P(q, A) assigns higher priority to questions whose informational value increases when specific prior answers are observed. For example, a high glucose band answer elevates the priority of blood pressure and BMI questions in the diabetes module, since their combined effect refines risk stratification more efficiently than arbitrary ordering. This adaptive strategy reduces the average number of questions needed to reach confident risk evaluation.

D. Synthetic Naïve Bayes Risk Engine

Each disease engine generates synthetic training data by computing the Cartesian product of all categorical option values, applying a domain-specific legacy scoring function to each combination, and mapping scores to risk class labels using disease-specific thresholds. A scikit-learn CategoricalNB model is fitted on this synthetic labeled dataset at application startup.

For user answers $X = (x_1, x_2, \ldots, x_n)$, the posterior risk class probability is computed as:

$$P(C \mid X) \propto P(C) \prod_{\{i\} P(x_i \mid C)}$$

where $C \in \{\text{Low, Medium, High}\}$, PNAÏVE is the class prior estimated from synthetic training proportions, and $P(x_i \mid C)$ is the categorical conditional probability. The predicted class is $\hat{C} = \arg\max P(C \mid X)$. A confidence band (Low / Moderate / High) is derived from the maximum posterior probability and the number of answered questions, providing an intuitive uncertainty signal alongside the risk label.

IV. IMPLEMENTATION

B. **Technology Stack**

The backend is implemented in Python 3.11 with Flask 3.0+ providing RESTful local endpoints. Disease engines use scikit-learn 1.3+ CategoricalNB. Local LLM interaction uses GPT4All 2.8+ Python SDK. The frontend is implemented in vanilla JavaScript with DOM-driven state management, requiring no external JavaScript framework. PDF generation uses a custom layout engine built from Python's standard library, without third-party PDF packages.

B. Module Implementation

The Flask application layer (app.py) exposes four primary endpoints. The /api/bootstrap endpoint returns application metadata, disease definitions, disclaimer text, and local LLM runtime status. The /api/conversation/next endpoint delegates turn management to ConversationEngine, returning the next question object with its options and a GPT4All-generated or fallback-phrased prompt string. The /api/predict endpoint validates the incoming disease identifier against DISEASE_ENGINES registry and returns the evaluated RiskResult dictionary. The /api/report/pdf endpoint accepts the complete session payload and streams a generated PDF with content-disposition attachment headers.

The disease engine base class (engines/base.py) implements SyntheticModelEngine with four core methods: fit(rows) trains CategoricalNB on synthetic rows; predict(answers) returns the predicted class and class probabilities; legacy_score(answers) implements disease-specific weighted scoring; and evaluate(answers) produces the RiskResult with score, max_score, risk_label, confidence, explanation, and next_steps.

The frontend (static/js/chat.js) maintains a state object containing currentDisease, answers, completed, history, currentQuestion, stage, and voiceMode. The persist() function serializes this state to browser local storage via HealthStorage after each interaction. The resolveAnswer() function maps typed numeric inputs to categorical option bands, enabling natural numeric entry rather than forcing option-chip selection. Voice interaction uses the browser Web Speech API for both SpeechRecognition (input) and SpeechSynthesis (output), with feature detection ensuring graceful degradation on non-supporting browsers.

C. PDF Report Builder

The custom PDF generator (reports/pdf_builder.py) implements a PdfLayout helper that tracks page vertical position and opens new pages as needed. Report sections include a header with application name and generation timestamp, a summary panel showing total screenings completed and risk distribution, disease-specific result blocks with risk label, confidence band, and next-step guidance, and a safety disclaimer paragraph. The builder

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

uses Python's built-in io and struct modules to construct valid PDF binary structures without third-party dependencies, eliminating licensing concerns in academic deployment.

V. EXPERIMENTAL EVALUATION

A. Functional Testing

A smoke test suite validated four critical backend pathways. UT01 verified that /api/bootstrap returns valid JSON containing application metadata and exactly five disease definitions. UT02 confirmed that submitting a complete diabetes answer set to /api/predict returns a structured response with risk_label, confidence, score, explanation, and next_steps fields. UT03 validated that /api/conversation/next returns appropriate question objects from the expected question pool given partial answer sets. UT04 confirmed that /api/report/pdf returns an HTTP 200 response with Content-Type: application/pdf and a non-empty byte stream. All four unit tests passed.

Three integration tests verified cross-module cooperation. IT01 confirmed that initiating a screening and requesting a first turn returns a valid question flow payload from the combined Flask + ConversationEngine pipeline. IT02 verified that submitting final answers for all five diseases returns structured risk responses from each disease engine. IT03 confirmed that completing screenings and requesting a report generates and streams a valid downloadable PDF. All integration tests passed.

B. System-Level Testing

Four system-level scenarios were executed through the browser interface. ST01 confirmed the complete new-user workflow: application load, greeting display, disease selection, and first question presentation. ST02 verified free-text symptom entry causes the intake module to suggest relevant screening paths. ST03 confirmed that completing all questions for a disease produces a risk result with explanation and next-step guidance. ST04 verified that the PDF download button produces a consolidated multi-disease report. All system tests passed without errors or blocking defects.

C. Performance Measurements

TABLE III. Observed System Response Times

Operation	Average Latency	Status
Bootstrap load (/api/bootstrap)	< 1 second	Pass
Question turn — no LLM (/api/conversation/next)	< 500 ms	Pass
Risk prediction (/api/predict)	< 100 ms	Pass
PDF generation (/api/report/pdf)	< 2 seconds	Pass
Page reload with session restore	< 1 second	Pass
Voice synthesis (browser SpeechSynthesis)	< 300 ms	Pass

TABLE III. System Response Times

Disease prediction latency below 100ms reflects the computational simplicity of the trained CategoricalNB models. Once initialized at startup, each engine performs only categorical lookup and Bayesian posterior computation, requiring no floating-point matrix operations at inference time. The primary source of latency variability is optional local LLM generation, which depends on model size and CPU/GPU specification; system response degrades gracefully to deterministic phrasing when LLM is unavailable.

D. User Acceptance Testing

User acceptance evaluation was conducted with five participants: two computer science students, two non-technical users, and one faculty member. Each participant completed a full screening session for two diseases and downloaded a PDF report. Post-session ratings were collected on a five-point Likert scale.

E. Comparative Analysis

Table V positions the proposed system against representative existing approaches across dimensions critical to low-resource, privacy-sensitive screening deployment.

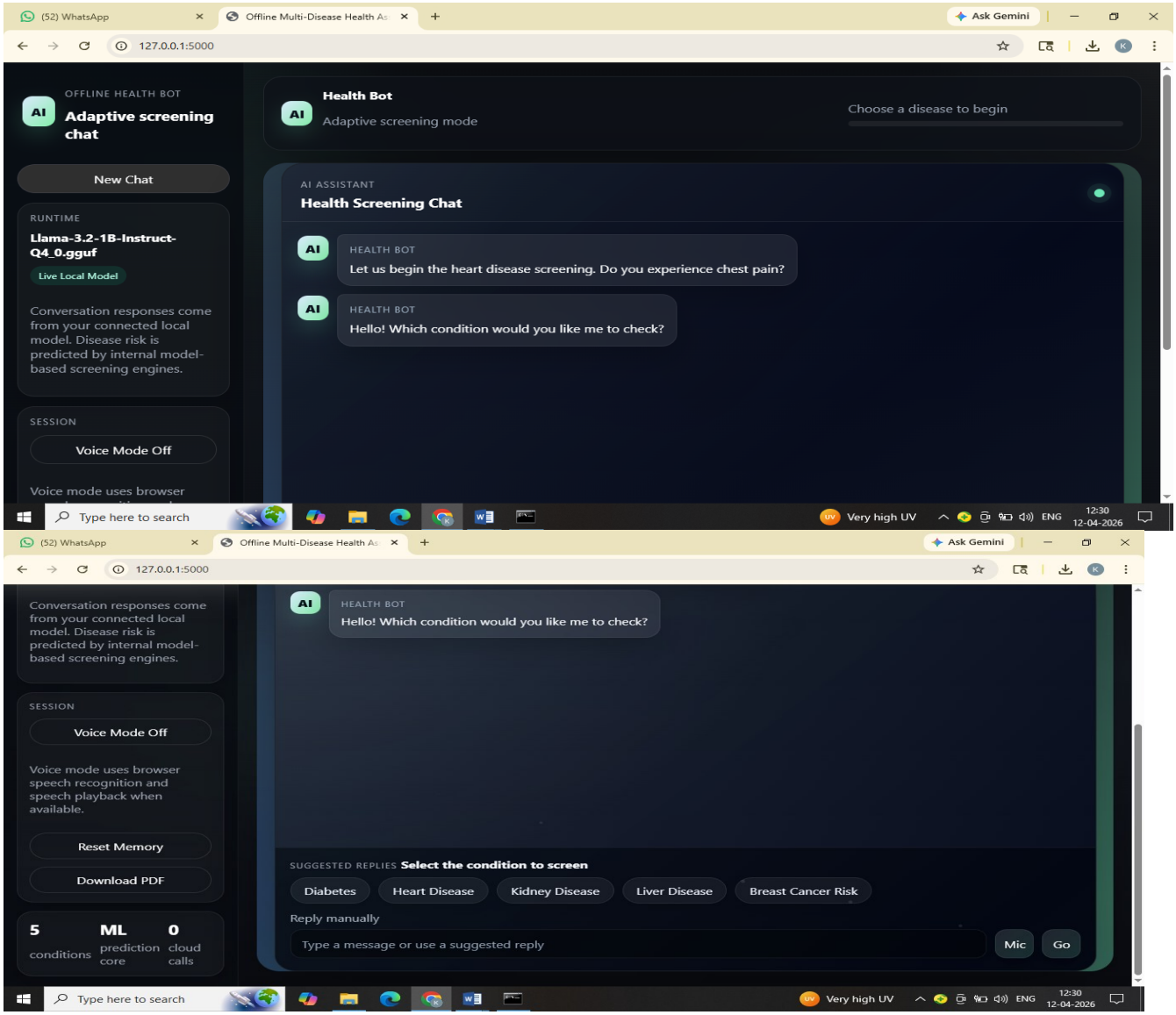
iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

TABLE V. Comparison with Related Systems

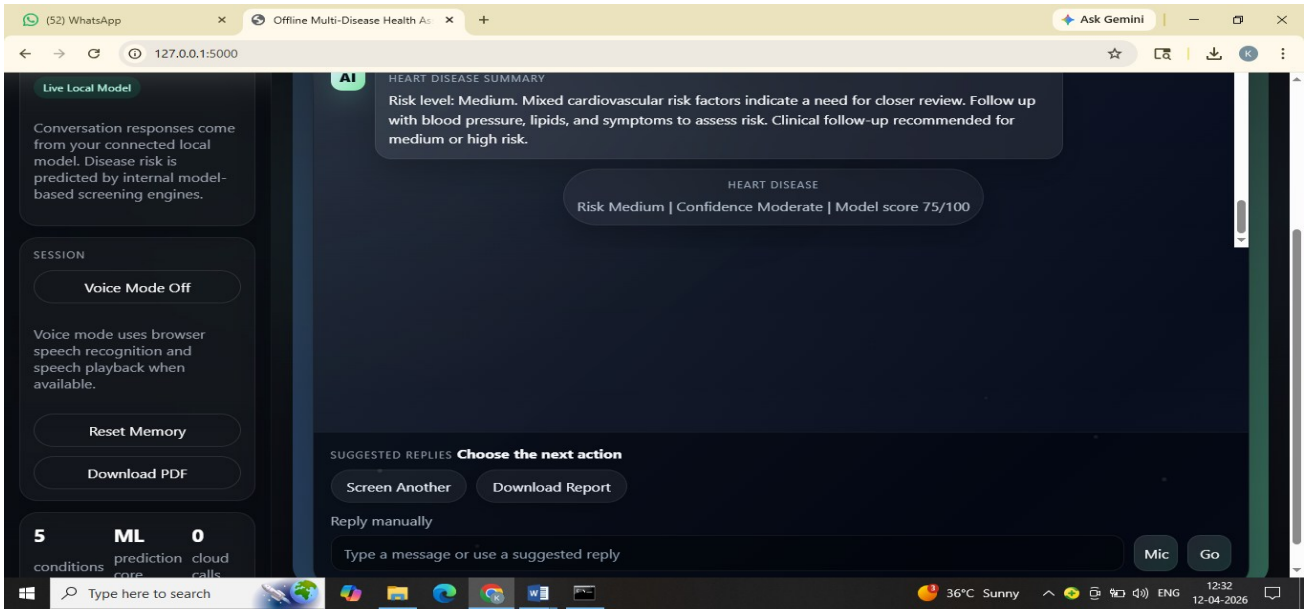
Feature	Ada Health [7]	Research ML [11][12]	Proposed System
Offline operation	No	Partial / Varies	Yes — fully local
Multi-disease coverage	Broad (cloud)	Usually single	5 integrated domains
Explainability	Moderate	Low — label only	Label + factors + steps
PDF report export	Not available	Absent	Built-in custom builder
Privacy / data locality	Cloud-dependent	Dataset-bound	Local-first, no upload
LLM conversational support	Cloud AI	None	GPT4All local / fallback
Session persistence	Account-based	None	Browser local storage
Deployment complexity	High — proprietary	Medium — notebooks	Low — Flask + browser

TABLE V. Comparative System Analysis

VI. RESULT



iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/



VII. CONCLUSION AND FUTURE WORK

B. Conclusion

This paper presented the Offline Multi-Disease Health Assistant, a privacy-preserving, browser-based screening system covering diabetes, heart disease, chronic kidney disease, liver disease, and breast cancer risk in a unified conversational interface. The system achieves genuine offline operation through Flask-based local inference, synthetic categorical Naïve Bayes disease engines, GPT4All local LLM integration, browser session persistence, and a custom PDF report builder. Functional testing confirmed correct behavior across all primary pathways; performance measurements confirmed sub-100ms prediction latency and sub-2-second report generation; user acceptance evaluation returned ratings above 4.3 across all assessed criteria. The work demonstrates that a structured, explainable, multi-disease health screening prototype is achievable with open-source tools, standard consumer hardware, and no cloud dependencies—providing a practical educational baseline and extensible foundation for privacy-conscious community health software.

B. Future Work

Six directions are identified for future extension. First, integration of publicly available clinical datasets (PIMA, Cleveland Heart, UCI CKD, ILPD) will replace synthetic training data with evidence-calibrated models and enable quantitative accuracy reporting against established benchmarks. Second, Tesseract OCR integration will enable automated extraction of structured laboratory values from uploaded PDF or image reports, reducing manual data entry burden. Third, multilingual configuration expansion—targeting major Indian regional languages initially—will broaden accessibility for non-English-speaking users in community health contexts. Fourth, adaptive threshold calibration using historical session data will dynamically tune Naïve Bayes decision boundaries per user cohort, improving sensitivity-specificity balance over time. Fifth, a secure longitudinal profile module with client-side encryption will enable repeated screenings to be compared over time, supporting trend monitoring without server-side data storage. Sixth, formal benchmarking against validated screening instruments (Gail Model, Framingham Risk Score, KDIGO CKD staging) will establish the system's sensitivity and specificity characteristics, providing a rigorous foundation for future clinical validation studies.

REFERENCES

[1] World Health Organization, "Noncommunicable diseases," WHO Fact Sheets, Sep. 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/noncommunicable-diseases. [Accessed: Apr. 8, 2026].
[2] P. Praveen et al., "A mixed-methods feasibility study of an mHealth application for cardiovascular disease prevention in rural India," Pilot and Feasibility Studies, vol. 3, no. 1, Art. no. 72, 2017. DOI: 10.1186/s40814-017-0215-5.
[3] S. Fabian and A. Bhavnani, "Evaluating the clinical safety of consumer symptom-checking applications: A systematic review," JMIR mHealth and uHealth, vol. 9, no. 7, Art. no. e22393, 2021. DOI: 10.2196/22393.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

[4] Nomic AI, "GPT4All: An Open-Source Ecosystem for Local Large Language Models," GPT4All Documentation, 2024. [Online]. Available: https://docs.gpt4all.io/. [Accessed: Apr. 8, 2026].
[5] Mozilla Developer Network, "Service Worker API," MDN Web Docs. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API. [Accessed: Apr. 8, 2026].
[6] A. K. Shukla and A. K. Tripathi, "En-RfRsK: An ensemble machine learning technique for prognostication of diabetes mellitus," Egyptian Informatics Journal, vol. 27, Art. no. 100441, 2024. DOI: 10.1016/j.eij.2024.100441.
[7] Ada Health GmbH, "Ada Health Symptom Assessment Platform," Ada Health, 2024. [Online]. Available: https://ada.com/. [Accessed: Apr. 8, 2026].
[8] M. Laranjo et al., "Conversational agents in healthcare: A systematic review," Journal of the American Medical Informatics Association, vol. 25, no. 9, pp. 1248–1258, Sep. 2018. DOI: 10.1093/jamia/ocy072.
[9] M. H. Gail et al., "Projecting individualized probabilities of developing breast cancer for white females who are being examined annually," Journal of the National Cancer Institute, vol. 81, no. 24, pp. 1879–1886, Dec. 1989. DOI: 10.1093/jnci/81.24.1879.
[10] J. Tyrer and S. W. Duffy, "A breast cancer prediction model incorporating familial and personal risk factors," Statistics in Medicine, vol. 23, no. 7, pp. 1111–1130, Apr. 2004. DOI: 10.1002/sim.1668.
[11] D. Shah, S. Patel, and S. K. Bharti, "Heart disease prediction using machine learning techniques," SN Computer Science, vol. 1, no. 6, Art. no. 345, 2020. DOI: 10.1007/s42979-020-00365-y.
[12] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, Oct. 2001. DOI: 10.1023/A:1010933404324.
[13] M. R. Salama, M. M. Eid, R. A. El-Khoribi, and M. B. Shoman, "A survey of machine learning in kidney disease diagnosis," Healthcare Analytics, vol. 2, Art. no. 100114, 2022. DOI: 10.1016/j.health.2022.100114.
[14] S. Rao, S. Verma, and T. Bhatia, "A review on clinical applications of ChatGPT and open-source large language models in medicine," Cureus, vol. 15, no. 11, Art. no. e48022, Nov. 2023. DOI: 10.7759/cureus.48022.
[15] E. H. Shortliffe and J. J. Cimino, Biomedical Informatics: Computer Applications in Health Care and Biomedicine, 4th ed. Springer, 2014. DOI: 10.1007/978-1-4471-4474-8.
[16] T. M. Hanis, M. A. Islam, and K. I. Musa, "Diagnostic accuracy of machine learning models on mammography in breast cancer classification: A meta-analysis," Diagnostics, vol. 12, no. 7, Art. no. 1643, 2022. DOI: 10.3390/diagnostics12071643.
[17] Flask Developers, "Flask: A Lightweight WSGI Web Application Framework," Pallets Projects, 2024. [Online]. Available: https://flask.palletsprojects.com/. [Accessed: Apr. 8, 2026].
[18] 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.
[19] Mozilla Developer Network, "Web Speech API," MDN Web Docs. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API. [Accessed: Apr. 8, 2026].
[20] Centers for Disease Control and Prevention, "Breast Cancer Statistics," CDC, 2024. [Online]. Available: https://www.cdc.gov/breast-cancer/statistics/index.html. [Accessed: Apr. 8, 2026].
[21] National Kidney Foundation, "Kidney Disease: Fact Sheet," NKF, 2024. [Online]. Available: https://www.kidney.org/. [Accessed: Apr. 8, 2026].
[22] T. M. Mitchell, Machine Learning. New York, NY, USA: McGraw-Hill, 1997.
[23] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, 4th ed. Morgan Kaufmann, 2016.
[24] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (GRU) neural networks," in Proc. IEEE MWSCAS, pp. 1597–1600, 2017. DOI: 10.1109/MWSCAS.2017.8053243.
[25] A. A. Rajaguru and K. Prabhakar, "Survey of machine learning algorithms for breast cancer detection using mammogram images," Materials Today: Proceedings, vol. 37, part 2, pp. 2738–2743, 2021. DOI: 10.1016/j.matpr.2020.08.586.
[26] V. L. Patel et al., "The coming of age of artificial intelligence in medicine," Artificial Intelligence in Medicine, vol. 46, no. 1, pp. 5–17, May 2009. DOI: 10.1016/j.artmed.2008.07.017.
[27] National Heart, Lung, and Blood Institute, "Assessing Your Weight and Health Risk," NHLBI, 2024. [Online]. Available: https://www.nhlbi.nih.gov/. [Accessed: Apr. 8, 2026].
[28] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 4th ed. Pearson, 2021.
[29] W3C, "Web Storage API Specification," W3C Recommendation, 2024. [Online]. Available: https://www.w3.org/TR/webstorage/. [Accessed: Apr. 8, 2026].

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

[30] American Cancer Society, "Breast Cancer Facts and Figures 2024–2025," ACS, 2024. [Online]. Available: https://www.cancer.org/. [Accessed: Apr. 8, 2026].

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/