Medical Report Processing Pipeline using Llama 3.3 70b, Celery and Redis

Nishikant Raut[1], Mrs. Fatima A. Ansari[2], Vivek Chouhan[3], Rehan Sayyed[4], Rohit Deshmukh[5]
[1,3,4,5]*Student, Department of Computer Engineering, M. H. Saboo Siddik College of Engineering*
[2]*Professor, Department of Computer Engineering, M. H. Saboo Siddik College of Engineering*

***Abstract*—Efficient processing of medical reports enhances clinical decision-making and patient engagement. This system uses Llama 3.3 70b with Celery to asynchronously extract key health indicators such as hemoglobin, PCV, RBC, and leukocyte levels and etc. from uploaded reports. Reports are stored on AWS S3, and tasks are managed via Redis and Celery workers for scalable processing. Extracted data is visualized through clear, interactive charts, helping patients and doctors track health trends. A personalized chatbot is generated using the user's last five reports, enabling meaningful, context-aware conversations. The system integrates PostgreSQL for metadata, MongoDB for chatbot memory, and Twilio for real-time status updates, ensuring fast, reliable, and user-friendly healthcare data interaction.**

***Index Terms*—AI, Celery, Redis, Health, Reports, Chatbot, Charts, Twilio, MongoDB, AWS, PostgreSQL**

I. INTRODUCTION

The growing digitization of healthcare has resulted in a vast volume of unstructured medical data, particularly in the form of diagnostic reports. Patients frequently receive multiple reports over time but often lack the medical expertise to interpret them or track their health trends. Simultaneously, healthcare providers face the challenge of extracting relevant clinical insights from these documents efficiently. Automating this process can significantly improve both patient understanding and clinical decision making.

This work presents a system that applies Generative AI (Llama 3.3 70b) to extract essential health indicators such as hemoglobin levels, PCV (Packed Cell Volume), RBC (Red Blood Cells), WBC (White Blood Cells), and leukocyte counts from uploaded reports. The processing is managed asynchronously using Celery, allowing the system to scale effectively and respond in near real time. Reports are stored on AWS S3, and the structured results are saved in PostgreSQL for further analysis. Redis is used as a task broker to manage job distribution across multiple workers.

A key feature of the system is a personalized chatbot that provides users with context aware responses based on insights derived from their last five medical reports. Extracted information is visualized using interactive charts, helping users and clinicians understand trends without requiring manual interpretation. Additionally, Twilio is integrated to send real time status updates, improving user engagement throughout the processing pipeline.

By combining AI-driven extraction, scalable task handling, and user focused design, the proposed system offers an efficient and intelligent solution for medical report analysis and visualization.

II. LITERATURE REVIEW

Visual tools like graphics and videos help healthcare professionals simplify complex data, improving communication, decision-making, and patient safety [1]. Abudiyab and Alanazi's work shows how visuals support better outcomes by making clinical information easier to interpret [1].

Austin, Mathiason, and Monsen used bubble charts and alluvial diagrams with Omaha System data to explore health patterns in older adults [2]. Their visual approach uncovered new insights and validated multiple hypotheses, showcasing the power of exploratory analytics [2].

Menon et al. created a Django-based tool combining LSTM models with data visualization to predict pharmacy needs [3]. It supports uploads, interactive views, and exportable charts, improving hospital operations and planning [3].

Rajabiyazdi and colleagues addressed patient-collected data as a design challenge, using visualizations tailored to both clinical needs and user expectations [4]. Their flexible approach improves clarity and communication between patients and providers [4].

Zulkafli, Ariffin, and Zakariya built a real-time monitoring system using IoT sensors and dashboards to track vital signs [5]. Their Agile-built platform enables early responses by visualizing patient data in Power Apps [5].

Liu et al. developed a 3D digital human model for displaying electronic health records with anatomical accuracy [6]. Using OCR and interactive visuals, their system helps both patients and doctors better engage with health data [6].

Hossain and team created a Gantt chart–based timeline tool for patient histories [7]. It simplifies longitudinal data and aids clinicians in making informed decisions quickly [7].

Meloncon and Warner reviewed health visualizations and found simple formats like bar charts and icon arrays most effective for public communication [8]. They stress the importance of accessible and user-friendly design [8].

Rajasagi developed a VR tool using Unreal Engine to visualize pandemic data in 3D [9]. This immersive approach supports policymakers in understanding and responding to public health trends [9].

III. PROPOSED SYSTEM

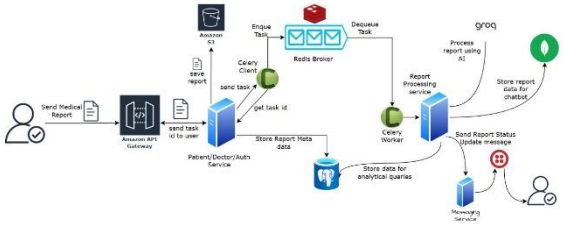

Figure 1: System Architecture of Report Processing

A. Report Ingestion and Task Queue Management
The system begins with a user uploading their medical report through a secure web interface. This upload request is first handled by Amazon API Gateway, which authenticates the request and forwards the document to a backend microservice responsible for managing patients, doctors, and user authentication. The report is then securely stored in Amazon S3, and a task ID is generated and returned to the user for tracking the processing status.

To ensure scalability and efficiency, a Celery client within the backend microservice enqueues the processing task to a Redis broker. Redis acts as a fast, in-memory message queue that temporarily holds the tasks. This asynchronous model ensures that multiple report processing requests can be handled in parallel without overloading the core processing unit, making the system highly scalable and responsive. Task metadata is simultaneously stored in a PostgreSQL database for future reference and analysis.

B. AI-Based Report Analysis and Data Storage
Once a task is dequeued by a Celery worker, the report processing pipeline begins. At this stage, the report is parsed and analyzed using the Llama 3.3 70b model running on a high performance inference backend such as Groq. The AI model extracts key health indicators from the medical report including values like hemoglobin, PCV, RBC, WBC, leukocyte counts, and other critical parameters.

The extracted structured data is stored in PostgreSQL to enable efficient analytical querying and visualization. Meanwhile, the unstructured and conversational data (e.g., summaries, insights) is stored in MongoDB, which serves as a memory bank for the chatbot service. This dual-database design PostgreSQL for relational data and MongoDB for document-based storage ensures optimal performance and flexibility for both structured queries and AI-driven features.

C. Visualization, RAG Chatbot, and User Communication
A key feature of the system is the integration of a personalized AI chatbot. After at least five reports have been processed for a user, a custom chatbot is generated using data from their historical reports. This chatbot leverages the stored health information to enable meaningful, context-aware interactions, allowing users to ask questions like, "How has my hemoglobin level changed over time?" The chatbot responds with insightful, data-backed answers tailored to the individual user.

For real-time communication, the system integrates with Twilio to send status updates via SMS or email. Users are notified when a report is uploaded, when processing is complete, and when new insights are available. Additionally, the platform provides interactive, easy-to-understand visualizations such as line graphs and trend charts to help users and healthcare providers track key health metrics over time. This user-focused design ensures not just efficient data processing, but also engaging and actionable health insights.

IV. RESULTS AND DISCUSSION

A. Results

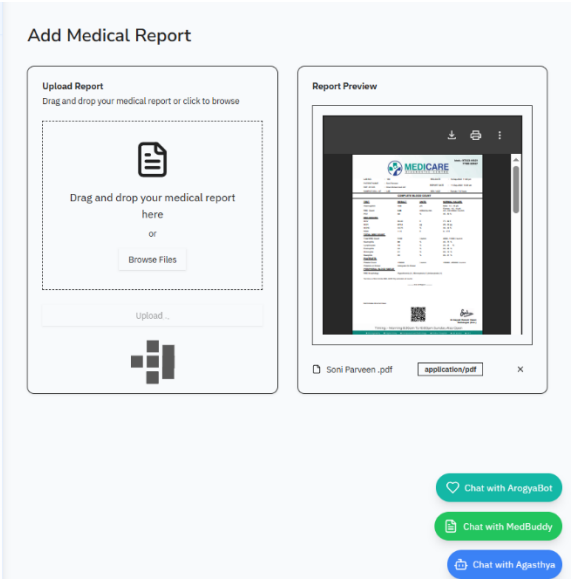

Figure 2: User Interface for uploading the reports

Figure 3: Processed report with extracted data

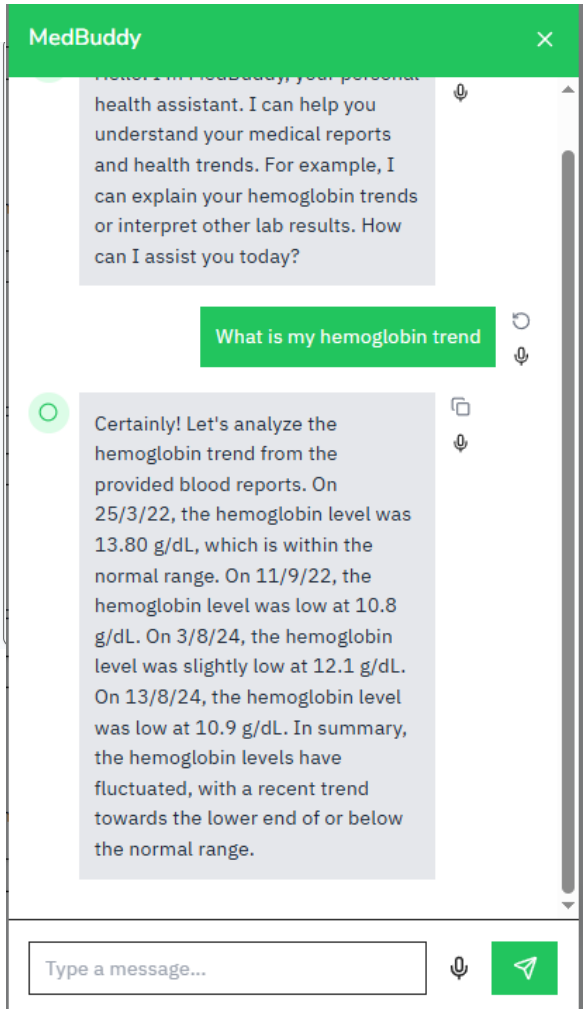

Figure 4: RAG chatbot response and UI

Figure 5: Data visualization and charts of report data

B. Discussion

The proposed system effectively combines AI-powered data extraction and visualization to make medical information more accessible and actionable. By analyzing uploaded reports and displaying results through intuitive cards and charts, users can easily understand health trends. The chatbot, trained on a patient's last five medical reports, enables meaningful, personalized conversations that enhance engagement and clinical insight. While variability in report formats

poses a challenge, the system's use of scalable technologies like Celery, Redis, and AWS shows promise for broader implementation in healthcare environments.

V. CONCLUSION

Bringing together AI-driven data extraction, real-time visualization, and personalized interaction, this system enhances the way patients and healthcare providers engage with medical information. By processing clinical reports asynchronously and presenting insights through dynamic charts and conversational AI, it ensures both accuracy and accessibility. The use of recent patient history for chatbot responses further personalizes the experience, making health monitoring more intuitive. This approach reflects a shift toward smarter, user-centered healthcare tools that simplify complex data while maintaining clinical relevance.

REFERENCES

[1] N. A. Abudiyab and A. T. Alanazi, "Visualization Techniques in Healthcare Applications: A Narrative Review," *Cureus*, vol. 14, no. 11, p. e31355, Nov. 2022, doi: 10.7759/cureus.31355.

[2] R. R. Austin, M. A. Mathiason, and K. A. Monsen, "Using data visualization to detect patterns in whole-person health data," *Res. Nurs. Health*, vol. 45, no. 4, pp. 466–476, Aug. 2022, doi: 10.1002/nur.22248.

[3] A. Menon, A. M. S, A. M. Joykutty, and A. Y. Av, "Data Visualization and Predictive Analysis for Smart Healthcare: Tool for a Hospital," in *Proc. 2021 IEEE Region 10 Symposium (TENSYMP)*, Jeju, South Korea, 2021, pp. 1–8, doi: 10.1109/TENSYMP52854.2021.9550822.

[4] F. Rajabiyazdi, C. Perin, L. Oehlberg, and S. Carpendale, "Communicating Patient Health Data: A Wicked Problem," *IEEE Comput. Graph. Appl.*, vol. 41, no. 6, pp. 179–186, Nov.–Dec. 2021, doi: 10.1109/MCG.2021.3112845.

[5] S. M. Zulkafli, M. M. Ariffin, and A. Zakariya, "Data Analytics and Visualization of Remote Healthcare Monitoring System," in *Proc. 2022 6th Int. Conf. Comput., Commun., Control Automat. (ICCUBEA)*, Pune, India, 2022, pp. 1–6, doi: 10.1109/ICCUBEA54992.2022.10010938.

[6] N. Liu *et al.*, "A New Data Visualization and Digitization Method for Building Electronic Health Record," in *Proc. 2020 IEEE Int. Conf. Bioinformatics Biomed. (BIBM)*, Seoul, South Korea, 2020, pp. 2980–2982, doi: 10.1109/BIBM49941.2020.9313116.

[7] F. Hossain, R. Islam-Maruf, T. Osugi, N. Nakashima, and A. Ahmed, "A Study on Personal Medical History Visualization Tools for Doctors," in *Proc. 2022 IEEE 4th Global Conf. Life Sci. Technol. (LifeTech)*, Osaka, Japan, 2022, pp. 547–551, doi: 10.1109/LifeTech53646.2022.9754925.

[8] L. Meloncon and E. Warner, "Data visualizations: A literature review and opportunities for technical and professional communication," in *Proc. 2017 IEEE Int. Prof. Commun. Conf. (ProComm)*, Madison, WI, USA, 2017, pp. 1–9, doi: 10.1109/IPCC.2017.8013960.

[9] D. P. Rajasagi, "Immersive Health Data Visualization in Virtual Reality," in *Proc. 2023 IEEE Conf. Virtual Reality 3D User Interfaces Abstracts Workshops (VRW)*, Shanghai, China, 2023, pp. 971–972, doi: 10.1109/VRW58643.2023.00328.