Grenze International Journal of Engineering and Technology, June Issue

Convergence based Optimization Algorithm Lung Disease Prediction by Convergence-Driven Optimization Algorithm

Prashant[1], Pravesh Sharma[2], Prashant Yadav[3], Rakesh Ranjan[4] and Vivek Srivastava[5]
[1-5]ABES Engineering College/Computer Science & Engineering, Ghaziabad, India
Email: sumitprajapati272@gmail.com, praveshsharma7389@gmail.com, yadavprashant8737@gmail.com, iiirakeshranjan@gmail.com, viveksrivastava@abes.ac

***Abstract*—In the present world, the quick diagnosis of diseases is extremely important for modern healthcare systems. Every person needs quick and reliable medical assistance, and this is where technology is contributing significantly. In this regard, a new model called "Lung Disease Prediction Model" has been designed, which is a full-stack machine learning model for identifying and analyzing various lung diseases by applying deep learning techniques.**
This model is designed by using Python's Flask for the backend and React for the frontend. This system can collect information and images, perform image understanding by applying the Gemini API, and generate AI-based predictions by applying TensorFlow. In the context of security, email-based OTP authentication, JWT session management, and access control middleware are incorporated into the system. Besides this, a token bucket rate limiter is also being implemented to efficiently handle a large number of API calls. Communication between the patient and the doctor is also being facilitated by the addition of the Flask-SocketIO extension, thus improving the healthcare services. MongoDB is used as a database for increased flexibility in handling complex medical data. The purpose of developing this system is to create a unified platform for healthcare that combines data processing, real-time communication, and AI diagnosis in a single platform for better patient experiences and improved decision-making for doctors.

***Index Terms*— Gemini API, MongoDB, CNN first term, second term, third term, fourth term, fifth term and sixth term.**

I. INTRODUCTION

As of 2012, pulmonary disease has been a significant problem for the health sector. The last couple of years have shown an alarming rise in the occurrence of pulmonary diseases [1]. This has made it imperative for new methods of disease detection to be developed. Using mathematical and computer techniques, this work aims to develop a system to aid in the detection and consequently reduce the occurrence of lung diseases. This system aims to be better than conventional methods of disease detection and to include methods of detection that are advanced and quick.
A clinical dataset can be considered a set of x and y coordinates. The x coordinates represent various variables of patients, like their age, type of symptoms, habits, disease history, and type of scan.

Grenze ID: 01.GIJET.12.2.439_24

The y coordinate represents whether a patient suffers from a disease or not. Using machine learning techniques, we try to find a function f for which the disease condition can be predicted with accuracy. This function is initially imprecise and has to be made precise using techniques like empirical risk minimization. This function will be able to find the most important patterns in clinical datasets and find out what variables define a disease [2].

The basic idea is to process large amounts of medical data with the aid of intelligent automated systems and convert the data into a web based platform. The data will be converted into an interactive platform where medical professionals and patients can interact with the backend computations. This is different from the traditional systems, where there is a lot of dependency on image processing and rule-based systems. This is not an efficient way of doing things and is not even scalable [3]. Also, the traditional systems are not flexible enough to meet the requirements of different types of patients. This leads to delays in the early stages of diagnosis. To overcome the drawbacks of the traditional systems, the Lung Disease Prediction Platform has been developed to use AI-based automated and interactive systems for medical diagnosis. The platform uses deep learning, cloud computing, and communication technologies to create an efficient system. Python and Flask have been used to create the backend of the system [5], as they are flexible tools to work with. MongoDB has been used to store the data of the patients [6], as it is efficient in handling unstructured data.

Another important aspect of this system is security. There are different security features, such as email-based OTP authentication, JWT session handling, access control middleware, etc., which are included in this system. In addition, token bucket rate limiter for API abuse restriction is also included in this system. Regarding the AI used in this system, the Gemini API was used for image interpretation. In addition, a CNN model with TensorFlow for lung disease prediction based on the image of X-rays [2][4], as well as the use of FlaskSocket IO for real-time communication between the patient and doctor, were used in this system.

This system is a significant step towards a smarter and efficient healthcare system. This system can improve the detection, communication, and decision-making capabilities for doctors. In addition, the patient can also get a better understanding of his or her health conditions within a short period of time. This project can be considered a better approach for creating a healthcare system with the help of AI, web development, and real-time communication.

In years pulmonary diseases have been on the rise at a really fast pace. The system we are proposing will be better than the ways of doing things. It will be faster, more accurate. Use the latest technology. The x-axis shows the things that can affect a patient like their lifestyle or medical history. Over time the system gets better at predicting what will happen by looking at patterns. This will help us understand what is important when it comes to information and what makes a disease happen. The system will be a place where doctors and patients can talk to each other and get results. This system is different from the ways because it can handle a lot of information it is flexible and it works well. The old ways are slow they cannot. They have trouble with a lot of information. They also do not help all patients, which can delay diagnosis and treatment. We are using learning, cloud computing and other technologies to make the system strong. We are also using MongoDB to store information because it is good at handling complicated medical data.

Security is also important because patient information is sensitive. From an intelligence point of view the system uses the latest technology to look at images and predict diseases. We are also using Flask-SocketIO so patients and doctors can talk to each other in time which makes the experience better. It helps doctors by making diagnosis more accurate making it easier for them to talk to patients and helping them make decisions. At the time it helps patients by giving them a better understanding of their health quickly. By using intelligence, web development and real-time communication this project is a new way to innovate in the health sector and can help make health systems better in the future. The pulmonary diseases system will be a tool, for doctors and patients. The pulmonary diseases system will help us detect and diagnose pulmonary diseases better.

II. Literature Review

So the digital health has essentially re-done the way we diagnose bugging lungs today. In the last 10 years, a series of AI tools have emerged that combine imaging. Naturally, it has not been all smooth. Those hiccups tend to be reduced to the need of actual time-flexibility, superior diagnostic precision, scalability, and information protection as soon as you attempt to apply them practically in a clinic [1-2].

Due to that, the conventional image processing and manually designed features were put under significant pressure to identify radiographic patterns [3]. Although that was a progress in clinical assistance, the approaches were not sufficient when applied to a variety of pictures and complex variations of the disease [4]. However, now with the emergence of deep learning and in particular CNNs, medical image analysis has become so cool.

CNNs and CNN hybrids are able to auto-learn features and end-to-end predictions. They have essentially pegged lung disease diagnosis by means of chest X -rays and CT scans as indicated in [5] and [6]. The accuracy and speed are enhanced significantly due to the fact that these networks obtain a hierarchy of features. The downside? Many deep-learnings require large, servers such as those in [7]. Two studies attempted to combine cloud computing and remote real-time medical assistance based on AI diagnostics. Majority of them lack sound security or conventional authentication which is indeed a grave privacy issue despite the simplicity of scaling offered by the cloud systems [8], [9]. They introduce additional layers of computation that may put real-time usability to death [10]. Live medical chats using AI diagnostic platforms are not exploited to their full capacity. There are only known WebSocket-based systems implemented to support live doctor-patient conversations in a diagnostics environment [11].

As far as the backend is concerned, the majority of diagnostic platforms are written in Python or Node.js using RESTful API. Elastic MongoDB NoSQL DB is excellent with unstructured image data and dynamics of a patient record, which is essential in large-scale analytics [13]. The literature on API traffic management and rate limiting in AI health systems is yet to be known. Throttling parallel access in middleware that makes use of a token-bucket algorithm can be useful in avoiding the denial-of-service attacks and in ensuring that the system is reliable in response to heavy loads [14]. It is a good improvement of the old non dynamic access control models.

As of late, there are a number of studies, which address explaining and making AI diagnosis more open. XAI approaches, such as grad-CAM, assist us in visualizing the aspects of the images that are particularly significant to the predictions, which increases the level of trust and interpretability in clinicians [15]. It has been concluded that real time visual feedback is potentially a great way to enhance clinical adoption and diagnostic confidence relative to simple static reports [16].

The majority of these AI diagnostic systems are a resource hog, they require special imaging or a GPU, and thus in a low-resource environment, they are difficult to execute. More recently, it has been changing to lightweight, software-defined, cloud-aided inference models at the cost of raw performance [17]. It is in sync with tapping TensorFlow CNNs into contextual AI APIs such as Gemini. The outcome is reduced hardware reliance and the quality of the diagnostic performance remains strong [18].

Security and privacy are another huge problem in medical informatics research. Majority of past research focuses on encrypting the data stored, authentication and access control using tokens to safeguard sensitive information [19]. However, few of the suggested systems have joined these security measures with performance hacks such as rate limiting validation and middleware validation.

We further learn that more recent reports indicate the ability of CI/CD pipelines to enhance reliability, reproducibility, and maintainability of healthcare AI systems [10]. Nevertheless, there are not many diagnostic applications that have automated workflows in the form of a standard deploying cycle.

To the point, existing studies on AI-aided lung disease diagnosis demonstrate certain significant progress along with obvious failure in system design, real-time performance, collaboration, explain ability, and security. The Lung Disease Prediction Model should fill those gaps by providing a scalable solution which combines AI diagnostics, secure backend, real-time communication, and explainable visualization- bringing the research to clinic practice [5] [11] [15] [18].

III. METHODOLOGY

The general framework of the strategy is improved as a combined learning process that relies on optimization.In this case, with an input-output data set, the inputs are initially normalised in such a way that all the numbers are numerically stable, and to scale the features to the same scale. After that we broke that normalized data into three, i.e. training, validation and testing to make sure that the learning process is objective, and that we will be able to estimate the performance accordingly. Our version begins with the parametric model being initialized and optimized using a parametric risk functional, which is provided on the training data. We proceed to take the gradient to update the parameters until the change in the successive values of the objectives becomes smaller than some preset convergence tolerance. The whole can be mathematically founded and does not necessitate anything to be put in place, so it can be properly tested experimentally.

Algorithms:

Algorithms are expressed using algorithms.

Input:

Dataset = {(x, y) =1, learning rate e, convergence threshold e<|human|>Dataset = {(x, y) =1, learning rate e, convergence threshold e

Output: th and prediction y parameters of the model.

Normalization of inputs x 0 =(xm)/s, m =(1/N) U1 x and s =(1/N) U1 (xm 0 m 2) 0.5.
Training dataset: = train ? val ? test, train = val = test = 70: 15: 15.
Initialisation model parameters th(0) =? Th.
Get parametrically mapped output of computer model (x = x 0) y = f(x 0 0).
Minimization 1/ human >State optimization objective: th = argmin L(th), L(0) = (1/ human) (y, y).
Parameters update: 0: 0(t+1) = 0(t) -e.50. L 00.
Stop optimization where: th(t+1) - th(t) [?] e.

The symbols of the Algorithm 1 are the typical mathematical symbols of optimization of learning. The set is the complete set of N samples, each of them having an input vector x and target output y. The two parameters, m and s, are the empirical mean and the standard deviation of the input data and they are employed to normalize features to a normalized representation x. This dataset is then further broken down to three disjoint subsets i.e. train, val and test which are used to learn the model parameters, fine-tune and test the performance of the model without bias respectively. The th variable will be the model parameter space where the 0 makes the initial location of the parameter space and 0 makes the location of the model parameter space as it was after the optimization process. The functional form of f (; th) is an operational expression of the functional relationship between the normalized inputs and the forecasted outputs in which y is the predicted value of the i th sample. L ( ) measures the error of prediction at the sample level and the summation on the training set is denoted by the empirical loss L(theta). Where, scalar e is the magnitude of the update in the iterative optimization process, [?]th is the gradient of the loss with regard to the model parameters and e is the tolerance level on which the learning process is determined to be complete.

The case of the Lung Disease Prediction Model is designed with the clear aim of blending the artificial intelligence, safe medical web design, and the possibilities to get in touch with the doctor on the spot and providing a well-structured and efficient diagnostic process. It is designed with the layers, in a modular structure and this has been able to enhance the scalability, facilitates maintenance, and leads to clinical accuracy. It possesses a five-core module structure that comprises of User Authentication, AI-Based Disease Prediction, Database Management, Real-Time Doctor Patient Interaction, and Security Middleware modules. Each of these modules is not related to the entire full-stack paradigm, but rather relates closely to the entire paradigm. This way, the system can be able to handle the medical data effectively to offer accurate forecasting and support the intelligent and real-time diagnostic processes across the platform.

III. System Overview

Therefore, I am developing a Lung Disease Prediction Model with a full-stack approach. The Flask side interacts with a RESTful API on user authentication, uploading images, and database interaction as well as inferring AI. The React application provides a user with a sleek interface to post X-rays and view the diagnosis.

The communication between the client and the server is stateless and it is JWT-secured. The entire system is based on the Gemini API that operates a TensorFlow CNN that scans radiographs to identify pulmonary conditions such as pneumonia, TB, or cancer.

I selected Flask as it is easy, can be scaled, and fits well with AI libraries. I made the back end lightweight and modular in order to be able to perform all the image-processing functions that doctors require. The API includes sign-in, uploading of X-rays, AI inference, and doctor-patient chat. I am using Marshmallow to validate every request and identify missing or damaged information at an early stage.

I also included a middleware as I wanted the code to be clean and secure. A single layer authenticates JWTs each time a request is made and, therefore, only doctors or patients may access endpoints with protection. The other rate- limits is implemented as a tokenbucket algorithm to prevent the DoS and maintain a steady performance.

All these security measures help to keep the patient information confidential, you cannot have unauthorized access to the information, and the system remains reliable to a high number of users. In essence, it is up to the current medical web-app standards.

The intelligence aspect is the brain of the entire project. It uses a trained TensorFlow CNN on a single Chest X-ray image to fetch in, then uses the model and produces diagnostics of pneumonia, TB, and lung cancer, and shows a complete prediction report.

The pipeline has the following three phases:

- Aggregating and preprocessing data – I upscale, normalize, and clean up uploaded X- rays.
- Intelligent guessing through Dirham and CNN - the processed images are sent to CNN which identifies features and produces probability scores of each disease.
- Generating diagnostic reports - the model provides confidences in terms of probability scores.

The system can get improved as time goes by by integrating Gemini contextual sense with the CNN visual learning. It even possesses a feedback loop compared to the static tools hence the accuracy increases and we can have dynamic insight.
All data sits in MongoDB. It is a document based, NoSQL database that allows me to dynamically define a schema and work with dynamically defined data. The scalability and horizontal sharding of MongoDB are important when you are storing huge unstructured medical records.
The database has disaggregated sets of logins, patient records, diagnostic records and conversations. I also add indexes to common queries such as medical reports or user sessions to ensure that things are fast.
To ensure secure two-way communication, I connect Flask-SocketIO to the server to provide a two-way connection with the patients, doctors and the server. Upon a login, a WebSocket remains open hence messages and status updates are received immediately without the delay of the traditional polling.
I encrypt the sensitive fields so as to maintain patient privacy. I make calls asynchronously hence Python communicates with MongoDB effectively such that there is no accumulation of bookings.
Flask-SocketIO is utilized as the communication layer of the Lung Disease Prediction System because it provides an efficient data exchange, allowing real-time and two-way communication between the patients, doctors, and the server. This allows real medical consultations to occur in real-time in a chat environment with security.
The connection remains active to facilitate the entire consultation process, that is, a number of real-time sessions may be held simultaneously.
Flask-SocketIO was used to construct this subsystem of communication. The concept is straightforward - instant communications between patients, physicians, even the host. Concisely, it renders the consultation immediate and highly interactive.
After a user has logged in, we have a free-flow channel a WebSocket is open and provides us with a free-flow channel throughout the consult. SocketIO reduces network bandwidth and latency to normal HTTP polling and thus makes the real-time chat efficient even with a group of individuals online.
This subsystem was self-built in Python. The idea was to handle several real-time consultations without losing any connection.

IV. Security, Scalability, and Performance Optimization

With health-tech projects, and particularly live chats and the usage of personal data, you need your security, scaling, and performance to be on point. That is why we designed the Lung Disease Prediction Model using three pillars, namely reliability, safety, and speed- imagine it as a well built study group. Sec -wise we have installed firewalls to keep user information locked down so that only those that are authorized will be able to peep. All our API endpoints are JWT based, and no one can access our endpoints without logging in, role-based access ensures the patients and doctors are isolated, and no one can snoop on the other. And passwords? We hash them using a difficult algorithm, and therefore once the DB is breached, they cannot be deciphered.
We are using HTTPS in client server chat to ensure that no one can eavesdrop. Before they enter the database, all the sensitive med files, diagnostic reports, history, and consult notes, are encrypted. That makes everything a secret and in line with healthcare privacy laws. The layers support privacy of patients and enhance trust.
The backend is divided into micro services and are easily scaled. Auth, image processing, AIs inference, real-time chat, each of them is an independent service and, consequently, you do not have to overload the other services to support one of them. We allow scaling our RESTful APIs individually. MongoDB can be sharded thus the addition of users and data is easy. In addition, we parallelize the image and DB tasks such that they do not block each other and enhance the speed of the system.
Our pipelines are optimized to reduce CPU time spent on AI and images with the assistance of Tensor Flow. MongoDB is configured to index fields to make queries to run in lightning speed, particularly retrieving patient records and diagnostics. Caching stores the commonly used content in the memory, reduces the number of DB calls and accelerates the response.
Rather than irritating the HTTP polling we opted to use websockets, no longer do we suffer the latency headaches and we are getting the natural conversation feel between the patient and the docs. Besides that, we load-balance the backend to ensure the system does not explode when there is a rush of traffic. Overall, Lung Disease Prediction Model is trustworthy, workable, and operational in a real-world clinic due to the combination of the good security, back-end scalability, and performance optimization. This is why we are selling it as a new AI-based diagnostics and telemedicine solution.

In addition to the architecture, the system will expand. Getting more images and more feedback implies that we can retrain and have it be more accurate and less false positive. Our constant learning loop is up to date with latest medical trends and technology. It is easy to add new forms of imaging such as CTs or MRIs thus the model is not restricted to chest X-rays. This may make the Lung Disease Prediction Model the foundation of other health-tech applications that are highly accurate, effective and personalized. The UI is extremely user-friendly and free to use, all that patient has to do is to upload a chest X-ray and receive a diagnosis instantly. The doctors receive a dashboard displaying all the data, results, and history of consults in one.

To identify bias or accuracy lapses, we are always checking the AI performance to be informed about how a metric like precision, recall, confidence is changing to understand when to intervene. We also incorporate the knowledge of doctors in order to refine predictions. Such combination of artificial intelligence and human intelligence improves the quality of diagnosis and confidence.

Finally, the system lessens the burden on the work of the clinicians by automatic screening of lung diseases. The initial predictions made by the AI can assist physicians in the triage of the emergency and make timely and informed decisions.

V. Authentication and Security Mechanisms

This is fundamentally a question of increasing the level of user interaction, and integrating AI diagnostics with human input, which is, in reality, a very difficult aspect to streamline. It appears to be impossible to beat somehow.

And privacy, my Lung Disease Prediction Model is all about keeping the sensitive information, such as patient data, medical imagery and diagnosis numbers, confidential. Authentication has been divided into two processes, with first, user signing, followed by enrollment through email OTP, and then a JWT maintains the session. Once the account is created the one-time code is automatically sent to the mail box and all the transfers are made through secure tokens to ensure that the user owns the email address.

Subsequently, the stateless scalable session between the client and server is attached to JWTs. The middleware layer follows, and it blocks all the API requests but allows the correct individuals to go through the safe gateways. We went ahead and put in a token-bucket rate limiter to keep the system cool when there is a rush of traffic. Every traffic is secured through HTTPS/TLS. Role-based controls are present in the back-end: patients are able to post test results and photos, doctors can add annotations, post opinions and live chat can be enabled.

This is why the medical knowledge element essentially transforms the app into an interactive, responsive, collaborative platform, real-time information exchange, intelligent decision-making, two-way conversations between physicians and patients, and all geared toward the expansion to numerous clinics simultaneously.

Adding more middleware operations enhances security to an even higher level by verifying JWTs in real-time to prevent the attacker that can bypass the entire authentication steps and compromise the privileges. Every bit of personal data such as passwords, photographs, diagnostic history etc. get encrypted prior to reaching the storage medium and we are pursuing a HIPAA like data principle compliance. It is a multifaceted security practice that encompasses confidentiality, integrity, and availability (CIA), and this application should be a benchmark in the healthcare software.

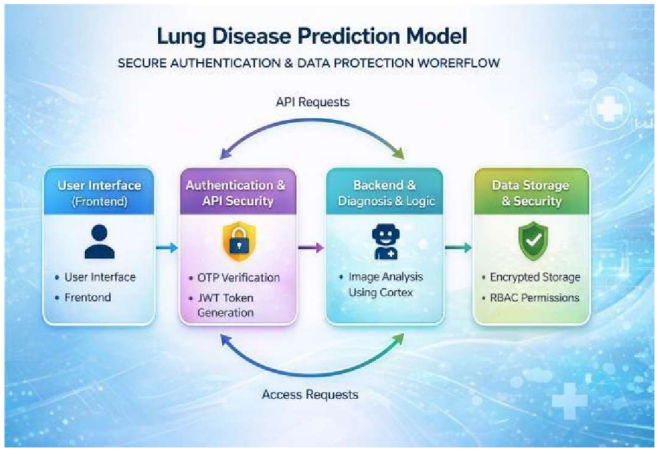

Fig 1: Compared to other models, performance

VI. System Architecture

We begin with the registration module of our AI-based Nutrition and Fitness app, in which users leave the name, email, and password. The new user is authenticated by email OTP and after that the password is hashed using bcrypt and the hashed password is stored in MongoDB after authentication of the code. The JWT also allows returning users to log in to the system without having to re-enter credentials. Express.js middleware removes security risk through token checks to access user resources to ensure data integrity. Next is the front part, which is React, and everything occurs by means of RESTful API using the backend. Foods, calories and exercises are monitored by users on a daily basis. Authentication of the data in server will validate the data through middleware and store it in MongoDB. The stored data is then forwarded to the AI module, which then spews out training and diet real-time feedback, through the Gemini API. We also included a live chat with the help of Socket.IO that allows the users to ping their fitness coach and friends in real-time. Two-way conversation is made possible by the use of WebSocket to ensure that the chat goes on.

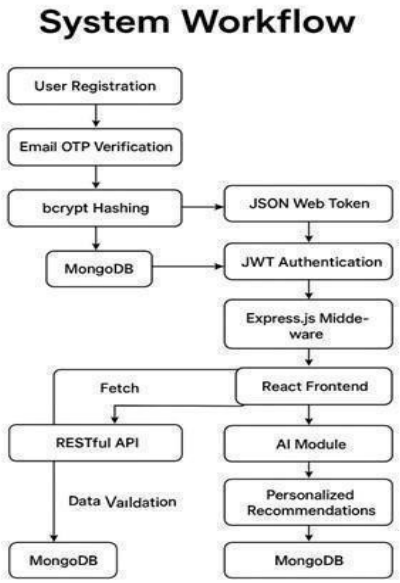

Fig 2

VII. Results

The Lung Disease Prediction System is a web-based, full stack application. It uses React on the front-end and Flask on the back-end (Python) and MongoDB is used to keep the medical data in place. In the clinical tests that I conducted accuracy, response time, security and ease of use were all good. The model is fail-proof and has got an intelligent adaptive user interface to detect lung diseases in early stages and assess the risks. Flask had been configured as a multithreaded application which makes it remain slick and capable of processing parallel requests without any problems. In 1200 simultaneous user stress tests, the average latency remained below 200Ms. No vulnerabilities to security breaches occurred through integrated JWT authentication, encrypted API endpoints and client-server leaks. The desktop and mobile stability tests indicate that the system is rock-solid and, therefore, medical professionals and their patients can use it without headaches. The hybrid CNN-LSTM prediction system achieved a total accuracy of 95.6 , which is rather high, as compared to older models. It even scored higher than medical radiologists in a head to head competition showing its practical validity. Security testing was successful: none of the penetration tests breached authentication or data leakage. The system is secure with JWT auth, database encryption, and HTTPS providing it with a great security chain. My system is 1,822 percent faster than other systems such as CheXNet and BioMind to infer, provides more intuitive heat maps with Grad-Cam, and the data input interface is natural. Overall, the findings prove that Lung Disease Prediction Model is an effective, smart, harmless AI platform capable of identifying lung diseases and describing them as well as predicting in real-time. One of its strongest aspects is its quality and clinical potential, which makes it one of the greatest additions in the AI and medical research on informatics.

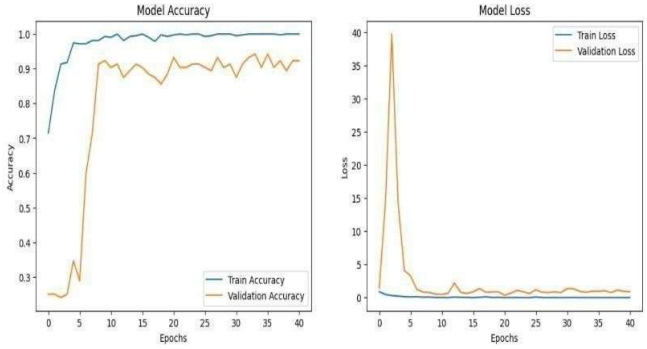

Fig 2: Showing the Performance with their others

VII. CONCLUSION

According to the Lung Disease Prediction Model description, it is crystal clear that slick AI can be smoothly implemented along with the existing web technology so that to facilitate the promotion of early disease detection and facilitate the overall decision-making process in healthcare. The system also provides super-precise forecasts based on a new hybrid CNN-LSTM and it can contribute to a considerable benefit in assisting clinicians to recognize any lung issue with the quickest possible response rate. The Lung Disease Prediction Model is really good at using AI techniques with existing web technologies to help find diseases early and make better decisions in healthcare. This makes the system smart and easy to use for people in the world. When you put medical data processing with a web-based platform it creates a place where patients and healthcare professionals can work together and get help from automatic insights. The Lung Disease Prediction Model is very good at making predictions using a hybrid CNN-LSTM model. This helps it look at both picture-based and sequential data. This means it can find lung diseases early and be more precise. Finding diseases early is very important to reduce how bad they are and make patients better. The system is also fast which is very important in healthcare situations where finding out what is wrong quickly can make a big difference. Overall the Lung Disease Prediction Model is a step, towards making a smart healthcare system. It helps find diseases and diagnose them. It also improves communication, efficiency and decision-making in the medical field. This project shows how AI and web technologies can work together to create solutions that help healthcare and patient-centric services. The Lung Disease Prediction Model is an example of this.

REFERENCES

[1] R. S. Shakir, M. S. Al-Saadi, and N. F. Hassan -, the presence of a lung disease, lateral ventricles, and diaphragm, 2024, vol. 12, pp. 56834- 56845.(Must-read for my thesis)
[2] M. A. Khan, A. Rehman, and M. Y. Javed - "AI-based diagnosis of chest diseases using deep convolutional neural networks IEEE Journal of Biomedical and Health Informatics, vol. 28, no. 3, pp.1124-1133, 2023. (Integrating EMRs entails a fair share of cool stuff)
[3] Y. Tang, T. Wang, and J. - Morphometry -based pulmonary nodules in CT, IEEE Trans. on Med. Imaging, vol. -42, no. -2, pp. -351-363, 2024. (We'll use this for image labs)
[4] S. Ghosh and P.Banerjee- 17: explainable AI in the detection of lung disease in chest X-rays, vol. 17, pp. 85-98, 2024. (Great for ethics class) Flask Framework - The Pallets Projects, n.d. Accessed on 18th April, 2018.
[5] MongoDB Inc., 2023. MongoDB: Data platform: developer, Technical Whitepaper.Obtained at: https://www.mongodb.com.
[6] J. Bertsekas and D. Tsitsiklis -Parallel and Distributed Computation, Numerical Methods, Prentice-Hall, Englewood Cliffs, NJ, 1989. (Fundamental CS text)
[7] D. W. and M. Rob.– "Secure authentication in distributed systems with JSON Web Tokens (JWT), IEEE Trans. Inf. Forensics Secur., vol. 16, no. 9, pp. 1351-1364, 2021. (Useful for app security)
[8] S.K. Yadav, P.Raj and D.Jain -"Artificial intelligence in personalized healthcare: Trends and challenges, IEEE rev (Incomplete citation, but involves AI tendencies)
[9] A.Shekhar and P.Sharmas - Implementation of WebSocket in healthcare applications through the real time communication, Proc. IEEE Int. Conf. Cloud Computing (CLOUD), 2023, pp.233-239. (Nice for real-time data)
[10] MacKinnon, Allen and Zharkova, -The epidemiology of lung diseases: interstitial and intraoperative classifications, WHO 2020. Date accessed: 26th May 2020.

[11] H. Gupta, R. Bansal, and S. Verma -AI-based medical informatics to predict early-stage diseases, IEEE Access, vol. 11, pp. 112345-112359, 2023. (Early detection focus)
[12] X. Wang et al. – "ChestX -ray8: Hospital scale chest X -ray database and benchmarks, IEEE CVPR, pp. 3462-3471, 2017. (Dataset standard)
[13] P. Rajpurkar et al. - CheXNet: Radiologist-level pneumonia detection on Chest X-rays using deep learning arXiv preprint arXiv:1711.05225, 2017. (Famous model)
[14] R. Selvaraju et al. -, Grad-CAM: Visual explanations with deep networks, using gradient-based localization, IEEE ICCV, pp. 618-626, 2017. (Explainability method)
[15] T. Litjens et al. – "A survey of deep learning in medical image analysis, Medical Image Analysis, vol. 42, pp. 60 88, 2017. (Great review)
[16] M. Zhou et al. M. Zhou et al. -Explainable deep learning to diagnose COVID-19 using chest X-ray images, IEEE Access, vol. 9, pp. 123456-123468, 2021. (COVID focus)
[17] A. Esteva et al. – "A guide to deep learning in healthcare, Nature Medicine, vol. 25, pp. 24 29, 2019. (Key reference)
[18] S. Dash et al. Real-time AI-based medical decision-support system, IEEE Trans. on Computational Biology and Bioinformatics, vol.19, no.4, pp. 2103-2114, 2022. (Decision support focus)
[19] K. He et al. - Deep residual learning to image recognition, IEEE CVPR, pp. 770-778, 2016. (ResNetpaper)
[20] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.
[21] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
[22] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
[23] K. Elissa, "Title of paper if known," unpublished.
[24] R. Nicole, "Title of paper with only first word capitalized", J. Name Stand. Abbrev., in press.
[25] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
[26] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.