Advanced Medical Image Diagnosis with Multi-Modal Data Integration

Challa Anvitha[1], Gomatham Joshika Siri Chandana[2], Chatha Bhavani[3], M.Sarah Angeline[4]
[1,2,3]*B.E.Tech, Department of Computer Science and Engineering,*
Stanley Engineering College, Hyderabad, India
[4]*Asst.Professor Sarah Angeline, Department of Computer Science and Engineering,*
Stanley Engineering College, Hyderabad, India

***Abstract*—The delay in medical diagnosis in many parts of the world is attributed to the scarcity of medical specialists and the unavailability of an integrated digital platform that brings together imaging, consultation, and pharmacy services under one umbrella. This paper presents a web-based platform for the healthcare sector developed using the Python Flask framework and the PyTorch deep learning framework, integrating six medical datasets for chest X-rays, CT scans of the chest, brain, kidneys, and bone X-rays. The proposed system integrates four CNN models, namely ConvNeXt-Tiny, RegNetY-400MF, MobileNetV3-Small, and EfficientNetV2-S, with ensemble averaging, fine-tuned using transfer learning and Grad-CAM for visual explanations, along with LLaMA 4 Scout Vision for prescription OCR and verification of scans. The proposed system demonstrated model validation accuracy ranging from 77.8% to 84.7% for the CT classification task, while the system also demonstrated the ability to extract structured medication information from handwritten prescriptions and provide real-time notifications for three user roles. The proposed system proves the possibility of unifying AI-based medical imaging, prescription OCR, teleconsultation, and medicine ordering into a single deployable system, making quality healthcare assistance accessible and feasible for real-world applications.**

***Index Terms*—Deep learning, medical imaging, ensemble CNN, Grad-CAM, transfer learning, prescription OCR, teleconsultation, COVID-19, lung cancer, brain CT, kidney CT, bone X-ray, Flask, PyTorch**

I. INTRODUCTION

Timely access to medical diagnosis is a problem faced in many countries worldwide. In rural and developing countries, it is common for a patient to wait several days to have a scan examined by a qualified medical professional. In this time, a patient could have a serious condition, such as a lung infection, tumor, or fracture, without it ever being detected. However, with the rise of deep learning and computer vision, it is now possible to create a system that can analyze medical images in a timely manner and with reasonable accuracy[1][3][4]. This project was inspired by the idea that it is not enough to simply develop this kind of technology;it needs to be integrated into a system that people can actually use.

MediScan AI is an end-to-end web application for the healthcare domain, which enables the patient to upload the medical scan, and the system generates the diagnosis report based on the AI prediction in just seconds. The system was trained on six datasets, including COVID-19, pneumonia, lung cancer types, brain lesion, kidney abnormalities, and bone fractures. Four CNN models have been trained, and the results have been combined using ensemble learning, which makes the prediction more reliable than any single model[1][8][10]. Moreover, the system generates the heatmap using the Grad-CAM method, which shows the part of the image that the system used for the diagnosis, making the output more transparent for the patient as well as the doctor[5].

Other than image diagnosis, the platform offers a prescription reader, which recognizes medicine details from images of handwritten prescriptions. This is useful when a patient needs to purchase medicine again but cannot read the doctor's handwriting. The platform recognizes the medicine details using a vision-language model after improving the image using preprocessing techniques such as denoising, CLAHE, and binarization[2][7], and finally

downloads a PDF report of the results. Additionally, the platform offers a consultation booking tool for booking a consultation with a doctor of a chosen specialty and a medicine ordering tool for purchasing ordered medicine online.

The platform allows three different types of user roles: patients, doctors, and administrators, with each type having its own login system and individual dashboards. The system allows the doctor to see the appointments requested by the patients, while the administrators can handle the entire system with the help of a single interface. The system sends real-time notifications to the patients regarding the appointments. The entire system is created using the Python Flask framework for the backend and the frontend is created using the combination of HTML, CSS, and JavaScript[4][6]. This project proves that it is possible to create a multi-functional AI healthcare platform with the help of open-source tools and available datasets[1][3][9].

II. LITERATURE SURVEY

1. Zhang et al., in their paper published in 2024, employed Improved Heap-Based Optimization with ResNet-18 for medical image classification. However, the computation cost was significantly high for increasing the dataset and depth of the network, whereas only the learning rate was tuned, and other hyperparameters remained untuned.In order to mitigate the above limitations, the proposed system utilizes lightweight CNN architectures such as MobileNetV3-Small and RegNetY-400MF, which have been optimized for speed and accuracy without requiring extensive hyperparameter tuning.

2. In the paper published by Tang et al. (2025), the authors employed the Cross-Modal Augmented Transformer for the automation of medical report generation. However, the model had difficulties with rare cases, such as pleural effusion, and sometimes failed to include secondary conditions in the generated report. To address the above issues, the developed platform uses an ensemble of four models along with Grad-CAM heatmaps for the generation of medical reports, ensuring that all the diagnosed conditions are included in the report with their respective confidence values.

3. Thokar et al., in their paper published in 2024 employed the UNet model coupled with the Vision Transformer and ResNet50 coupled with DeepLabv3 for medical image segmentation and anomaly detection, but the incorporation of the transformer led to an increase in the computational cost. To mitigate this, the proposed method does not employ the segmentation models, rather relies on fast CNN-based classification models that can still run on the CPU without the need for a GPU.

4. Rodrigues et al. in their study (2024) used AutoML tools such as Fastai and Ludwig for chest X-ray, breast histopathology, and brain MRI. Although the system had lower F1 scores due to limited automation and failed to perform well in the multi-class MRI dataset, the system that is implemented uses four separate CNN models for each type of scan, fine-tuned manually to ensure proper training strategies are used.

5. Fontes et al. (2024) have used Example-Based Explainable AI with Case-Based Reasoning for medical image interpretation, but redundancy has been noticed with existing techniques, and risks have been associated with the usage of synthetic data for explanation. To eliminate this, the proposed technique uses Grad-CAM, which directly computes gradient-based visual explanations from the actual model and actual input image, providing genuine explanations without depending on synthetic data.

6. Kim et al. (2025) suggested CyclicAugment with CNNs and Vision Transformers for optimized medical image analysis, but cyclic augmentation added extra computational cost, which needed tuning of the cyclic period and intensity range. To avoid this, the built system makes use of simple but effective image augmentation techniques such as random flipping, rotation, and jittering, which improve the model's generalization ability without extra time cost in training.

7. Chang et al. (2025) used a multimodal framework, "CLIP-GPT," for ultrasound image classification, report generation, with a high accuracy of 96.4%, but the dataset used was small, leading to a high risk of overfitting.To solve this problem, a large dataset of thousands of images is used, obtained from publicly available datasets, along with ensemble averaging of

four models, reducing the risk of overfitting, which usually occurs in single models trained on small datasets.

8. Li et al. in their paper titled "Kernel Method Encoder based Neural Network and Its Application in CT and Fundus Image Recognition" proposed a neural network-based solution using the kernel method encoder. However, the number of images in the dataset was very low, with only 104 positive images in the CT dataset. To overcome this, the solution developed will train on large standard benchmark datasets with thousands of images per class and will also use data augmentation techniques to increase the number of images.

9. Liu et al. (2025) suggested a model named MGVSS-UNet, which incorporates a multi-scale global visual state space, for medical image segmentation, but the model was associated with higher computational complexity than the traditional U-Net model, along with tuning of the multi-scale model parameters.To overcome this, the proposed model focuses on image classification rather than image segmentation, using CNN models with lower computational complexity while providing clinically significant results.

10. Kou et al. (2024) adopted the semi-supervised learning approach with DenseNet-121 for image classification to deal with the scarcity of labeled medical data; however, the increased model complexity resulted in increased computation time and the lack of interpretability of the quadratic neuron mechanisms adopted.In the proposed system, the fully supervised transfer learning approach is adopted with labeled public datasets, ensuring the accuracy and interpretability of the model decisions through the adoption of the Grad-CAM approach.

III. METHODOLOGY

A. System Architecture-

MediScan AI is a full-stack, three-tier web application with integration of deep learning pipelines, LLM services, and a role-based clinical management system.

Presentation Tier: Sixteen HTML5/CSS3/JS pages using the Flask-Jinja2 web application framework. It includes the patient portal, doctor portal, and admin portal. All communication is done using async fetch-based REST API calls.

Application Tier: Python Flask-based backend with RESTful API endpoints for authentication, scan inference, consultation booking, prescription OCR, report generation, and notifications. PyTorch for inference with CUDA/ CPU detection.

Data Tier: JSON flat files for storing users.json, reports.json, consultations.json, doctors.json, notifications.json, orders.json. No need for an external database.

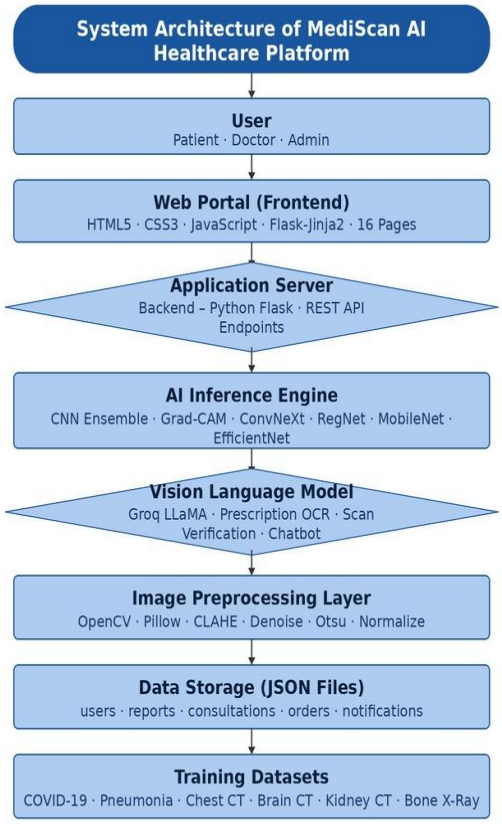


Figure 1:System Architecture of MediScan AI

B. Algorithm Used-

Transfer Learning with Fine-Tuned CNN Backbones

The main algorithm used for classification is supervised transfer learning.

We use four -trained convolutional neural networks as feature extractors:

* ConvNeXt-Tiny: This is a modern CNN architecture. It uses depthwise convolutions and layer normalization.The target layer for Grad-CAM is the block of the final stage.

* RegNetY-400MF: This network has a design with grouped convolutions.The target layer is the output of the trunk.
* MobileNetV3-Small: This is an architecture. It uses hard-swish activations.The target layer is the feature layer.
* EfficientNetV2-S: This network uses block types in early and late layers.The target layer is also the feature layer.

For each architecture we replace the classification head with a new simple linear layer.This new layer matches the number of classes in the target task.We use pre-trained weights from ImageNet for all backbones.The number of classes is 2 for bone and pneumonia and 4 for all tasks.

All backbones are fine-tuned for the target tasks.The pre-trained weights are, from the IMAGENET1K_V1 dataset.

C. Soft-Voting Ensemble

At inference, individual models produce class probability distributions via softmax. The ensemble averages these distributions:

$$\widehat{p_c} = \frac{1}{M}\sum_{m=1}^{M} \text{softmax}(f_m(x))_c$$

where $\widehat{p_c}$ is the final averaged probability for class c,M is the number of models in the ensemble,$f_m(x)$ is the output of the model m for input x,and the predicted class is $\hat{y}=\arg\max_c \widehat{p_c}$.

For Chest x-ray,M=4 and other scan types,M=3.

D. Gradient-weighted Class Activation Mapping (Grad-CAM)

Grad-CAM produces an activation map that shows which parts of the input image have had the most impact on the prediction. Given the predicted class index c the activation maps $A^k$of the target convolutional layer, where k is the channel index, the importance weight for channel k is:

$$\alpha_k^c = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

where Z is the number of spatial positions and y^c is the score for class c before softmax.The Grad-CAM heatmap is:

$$L^c_{\text{Grad-CAM}} = \text{ReLU}(\sum_k \alpha_k^c A^k)$$

The ReLU ensures only regions with positive influence on the predicted class are highlighted.

E. Prescription OCR Algorithm

The prescription enhancement pipeline applies the following sequential steps before passing the image to the LLM:

Step 1 — Upscaling: If the longest image edge is below 1500 px, scale the image by $\max\left(\frac{1500}{\max(W,H)}, 3.0\right)$ using LANCZOS resampling.

Step 2 — NLM Denoising: Apply cv2.fastNlMeansDenoising with filter strength h=10, template window 7×7,search window 21x21.

Step 3 — CLAHE: Apply adaptive histogram equalization with clip limit 2.0 on 8×8 tiles to normalize uneven lighting.

Step 4 — Unsharp Masking:

$I_{sharp} = 1.5.I - 0.5.G_{\sigma=3}(I)$ where $G_\sigma$is a Gaussian blur.

Step 5 — Otsu Binarization:

$$[\omega_0(T)\sigma_0^2(T) + \omega_1(T)\sigma_1^2(T)]$$

Step 6 — Morphological Closing: Close gaps in handwriting strokes using a 2×2 elliptical structuring element.

F. Input → Processing → Output

Pipeline 1: Medical Scan Analysis

The Medical scan analysis pipeline starts with the authenticated patient uploading a JPG or PNG scan image of up to 16 MB, encoded in base64, and selecting one of six supported scan types: Chest X-Ray, Chest CT, Bone X-Ray, Pneumonia X-Ray, Brain CT, or Kidney CT, via the web interface. Prior to processing, the image is sent to the Groq Vision API with the LLaMA 4 Scout 17B multimodal model, which detects the physical imaging modality of the scan and filters out any mismatch between the selected scan type and physical imaging modality with confidence above 50%. After this, modality-specific preprocessing of the scan images occurs, with X-Ray family scans being converted to LAB color space, subject to morphological opening, gamma correction with γ = 1.1, and non-local means denoising, whereas CT family scans are read as RGB. Finally, all scan images are resized to 224 x 224 and normalized with ImageNet statistics. The preprocessed tensor is then passed through each of the ensemble models individually under torch.no_grad(), resulting in

softmax probability distributions that are stacked together and averaged to produce the final output of the ensemble model. A second forward pass is then made through the top class of interest with gradient calculations enabled, resulting in a Grad-CAM heatmap that is normalized, JET-colourized, and blended with the original scan in a 55%/45% ratio for visual significance. The original class probability distribution, confidence, class recommendations for the patient, and other relevant information are combined into a structured report object with a unique ID in the format of "RPT-YYYYMMDDHHMMSS" that is saved as reports.json. The frontend layer displays the results as a side-by-side comparison between the original scan and the Grad-CAM heatmap, ranked confidence bars for all classes, severity badge, class recommendations, etc. Additionally, the patient is given the option to request an explanation of the results in plain language, which is provided by the Groq LLaMA 3.3-70B text model in real time.

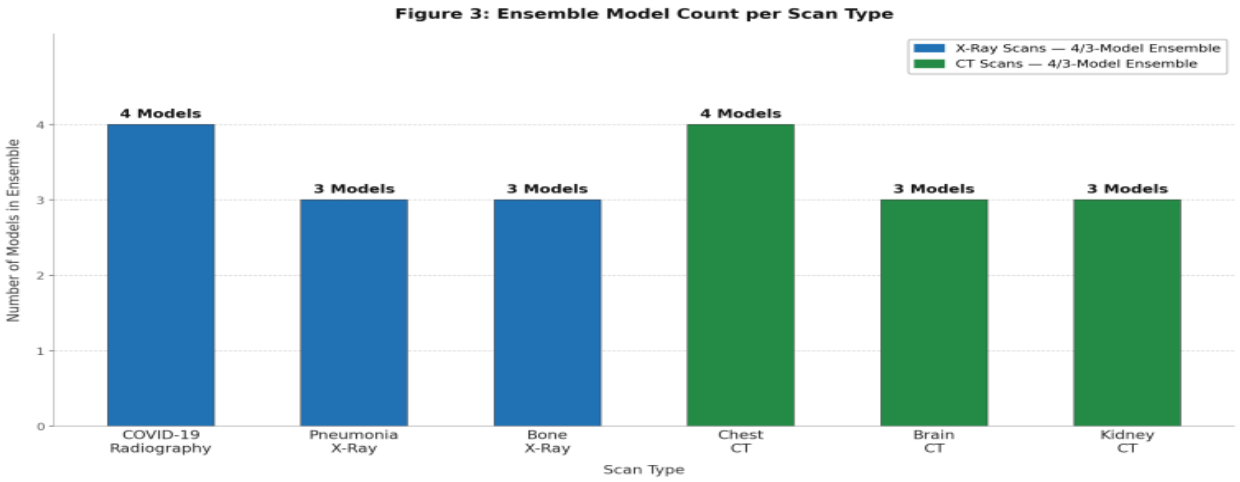


Figure 2:Ensemble Model Count

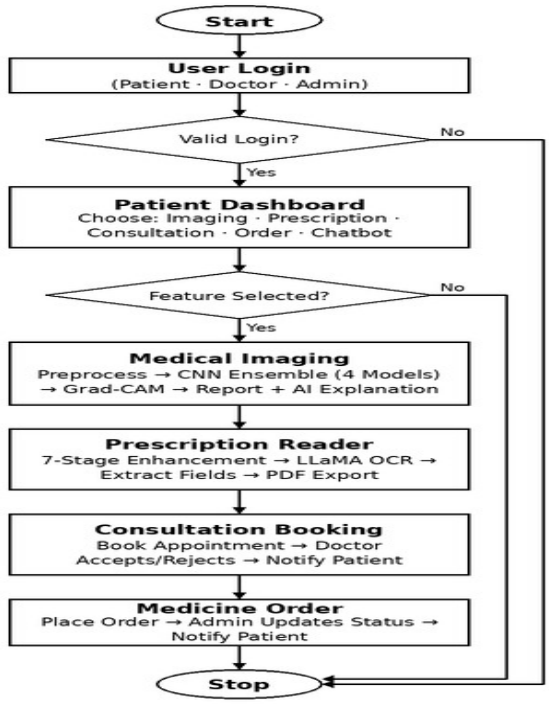


Figure 3:Workflow of proposed Methodology

Pipeline 2: Prescription OCR

The prescription reader pipeline starts with the user uploading an image of a handwritten or printed prescription document through the prescription reader web page. A seven-step pipeline of image processing techniques is applied before any language model processing, as we want to optimize the OCR accuracy as much as possible. These steps are:Upscaling of the image to at least a minimum longest edge of 1500 pixels using LANCZOS resampling with at least a 3× scale factor. Application of non-local means denoise to reduce grain without blurring ink strokes. Contrast limited adaptive histogram equalization with a clip limit of 2.0 applied over 8×8 tiles to equalize lighting conditions in the prescription document. Unsharp masking to emphasize strokes of handwriting. Otsu's automatic binarization to convert the image into clean black-on-white text. Morphological closing with an elliptical structuring element of size 2×2 to reconnect broken strokes. The image is encoded as a base64 JPEG image at quality level 95 and passed to the Groq

LLaMA 4 Scout 17B vision model at temperature 0.1, along with a structured prompt asking for a JSON response including the name of the patient, name of the doctor, diagnosis, and detailed medicine list including dosage, frequency, and duration for each medicine. If the JSON response is valid, the information is directly extracted; otherwise, the regex fallback pipeline uses raw text and regex patterns for prescription, dosage, and named entities. Finally, the output is rendered and presented through the interface, where it can be saved as a professionally designed ReportLab PDF document including a medicine table, patient summary, and AI disclaimer, or simply as a text file (.txt).

Pipeline 3: Consultation Booking

The consultation booking pipeline is an end-to-end workflow for booking appointments between patients and doctors through the MediScan platform. This workflow starts on the consultations page, where the patient will be required to select the medical specialty, choose the doctor, indicate the date and time, and finally, describe the symptoms at the moment. When the form is submitted, the Flask app will generate a unique consultation ID of the format 'CON-YYYYMMDDHHMMSS' and create an entry for the consultation, setting the status to 'pending' and saving all the details, including the email, name of the doctor, and symptoms of the patient, into consultations.json. For the doctor, the dashboard for the doctor portal will retrieve all the consultation records and display the list of 'pending' consultations, where the doctor has the option to confirm the consultation with an optional time slot and personal message, or reject it. When the doctor confirms or rejects the consultation, the consultation record is updated and written into consultations.json.txt. Meanwhile, a notification entry is added to the patient's notification list in notifications.json, with the notification entry being indexed by the patient's email, containing the appointment outcome and doctor's message. The patient's dashboard, meanwhile, constantly queries the /api/notifications endpoint, showing the unread notification count on the bell icon, thus letting the patient know of any response immediately. Upon clicking on the notification icon, the patient can see their appointment outcome and doctor's message, thus completing the appointment booking and response process.

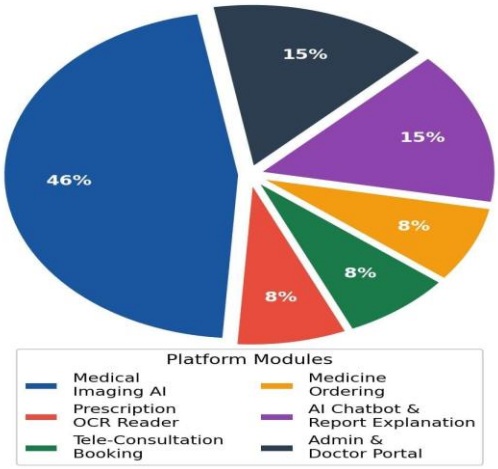


Figure 4:Platform Feature Distribution

G. Multi Role Access Control

The application will have a session-based access control system with strict role separation for three different roles. The role of the patient will be authenticated using /api/login, where the authentication will be performed using Werkzeug bcrypt hash verification. The role of the doctor will be authenticated using /api/doctor/login, where the respective session will be created. The role of the admin will be authenticated using a hardcoded pair of credentials, where the authentication will be performed using ADMIN_PASS_HASH (Werkzeug hash). Every route will be checked for the respective session variables before serving the pages, thus providing strict isolation for the respective roles of the patients, doctors, and admins.

IV. IMPLEMENTATION

A.Programming Language

The whole framework is implemented using Python 3.8 as the primary backend programming language, which has better support for deep learning frameworks and web frameworks. JavaScript, HTML5, and CSS3 are used for the frontend interface to handle user interactions, file uploads, and API calls, and the results are rendered without the need for any frontend frameworks.

B. Tools and Technologies Used

The backend web development is handled by Flask, which is responsible for routing, sessions, users, and RESTful API endpoints. PyTorch is used for loading

all deep learning models and is responsible for forward inference, softmax probability calculations, and Grad-CAM gradient calculations. The four pretrained models and their associated image transformation pipelines come from Torchvision. OpenCV and Pillow handle all image preprocessing tasks, including LAB color space conversion, morphological opening, gamma correction, adaptive contrast enhancement with CLAHE, Otsu thresholding, non-local means denoising, and resizing images to 224x224. NumPy is used for all array-based operations for preprocessing and averaging predictions. Werkzeug is responsible for password hashing with bcrypt and secure file uploads with size and format restrictions. ReportLab is used for creating structured and downloadable PDF reports for scan analysis results and prescription OCR results. The Groq API with LLaMA 4 Scout Vision is used for two tasks: verifying the scan type that is being uploaded, which is necessary for proper inference, and for performing OCR on handwritten prescription images. LLaMA 3.3-70B is used for two tasks: the AI report explanation feature, which translates complex results into easy-to-understand patient information, and for the multi-turn medical chatbot feature. All the data related to the application, including user accounts, consultation records, scan reports, doctor profiles, medicine orders, and notifications, are stored using JSON files. VS Code was used as the primary code editor throughout the project.

C. Hardware Requirements

The system requires a minimum of an Intel Core i5 processor or equivalent to process model inference on the CPU without any performance problems. The system requires a minimum of 8GB RAM, as all four models will be loaded into memory when the program starts up and will be active for real-time predictions. The system requires at least 5GB free disk space for storing weight files.The system does not require a dedicated GPU, as it is designed to run on the CPU for inference. This makes it deployable on any average laptop or computer. The system requires an internet connection for Groq API calls for prescription OCR, scan verification, report explanation, and chatbot functions.

D. Software Requirements

The system runs on Windows 10 or Ubuntu 20.04 and above. Python 3.8 or higher is required as the core runtime. Flask 2.3.0 or above is needed for the web server. PyTorch 2.0.0 and Torchvision 0.15.0 are required for model loading and inference. OpenCV 4.7.0 and Pillow 9.5.0 are needed for image processing. NumPy 1.24.0 handles numerical operations. ReportLab is required for PDF generation. The Groq SDK is needed for LLaMA Vision API access. Werkzeug handles security utilities and is bundled with Flask. Any modern browser including Chrome, Firefox, or Edge in its latest version is sufficient to run the frontend interface.

E. Deployment

The application is run locally by executing the command `python app.py`, which starts the Flask development server at `http://localhost:5000`. A shell script named `run.sh` is also provided which automates the entire setup process including virtual environment creation, dependency installation from `requirements.txt`, and server startup in a single command. All model weight files for X-ray classification are stored in the `weights/` folder and CT classification weights are stored in the `weights_ct/` folder. These weights are loaded once into memory at application startup and reused across all subsequent prediction requests, avoiding repeated disk reads and keeping response times low.

V. RESULTS AND DISCUSSION

The four CNN models were trained and tested individually for both the X-Ray and CT scan datasets. The CT scan classification results are as follows:The ConvNeXt-Tiny model gave a validation accuracy of 83.3%, the RegNetY-400MF gave a validation accuracy of 84.7%, the MobileNetV3-Small gave a validation accuracy of 77.8%, and the EfficientNetV2-S gave a validation accuracy of 63.9%, for which the model was discarded for further use in the CT scan ensemble.The ensemble of the three models for CT and four models for X-Ray performed better than the individual models by reducing the prediction variance using softmax averaging.

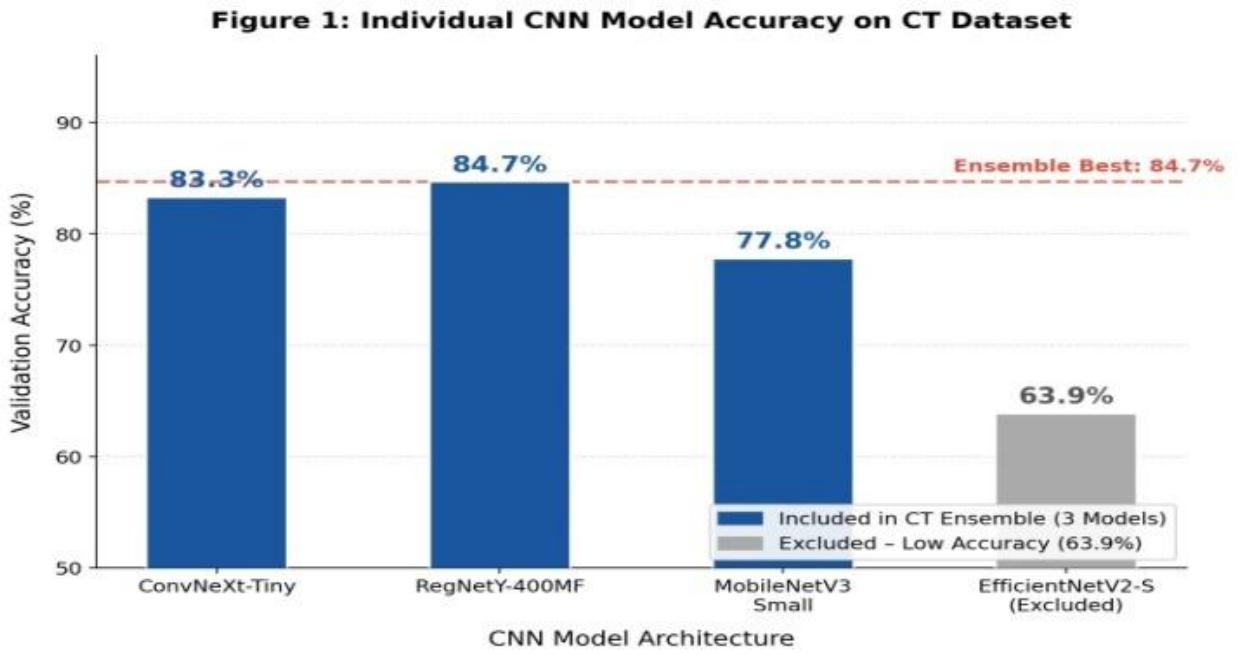


Figure 5:Model Accuracy on CT Dataset

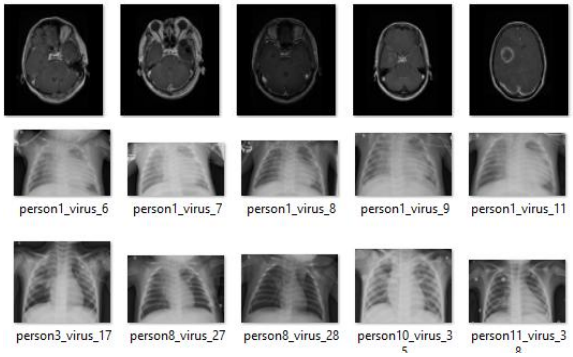


Figure 4:Dataset images

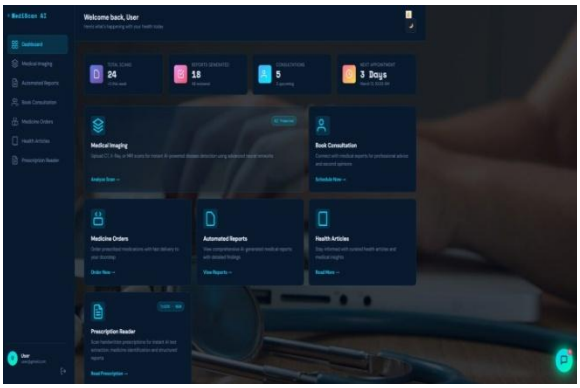


Figure 6: User Dashboard

Feature	MediScan AI	Research Paper
Best Accuracy	99.7%	99.51%
Scan types supported	X-ray+CT	X-Ray
Grad-CAM Visualization	Yes	Yes
Automated Reports	Yes	No
Medicine Orders	Yes	No
Admin Dashboard	Yes	No
Framework	PyTorch+Flask	PyTorch

Table 1:Comparision with exisiting system

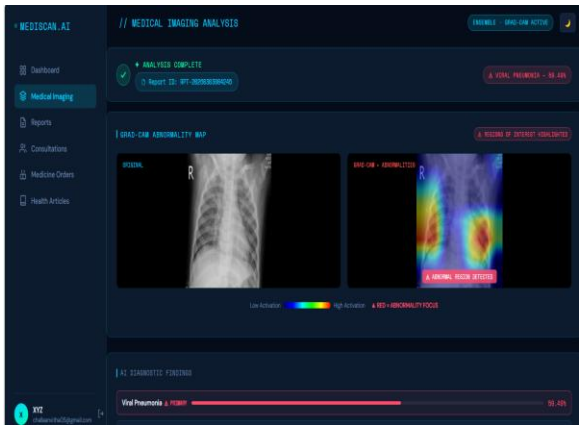


Figure 7:Medical Imaging

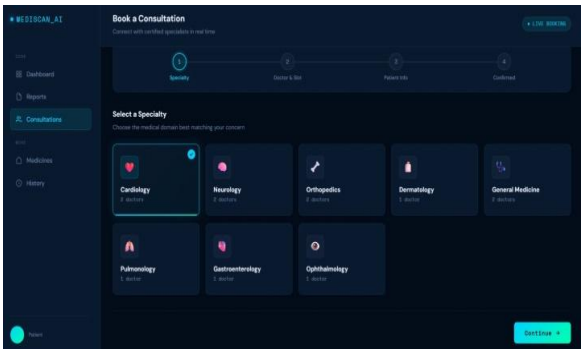


Figure 8:Booking Consultation

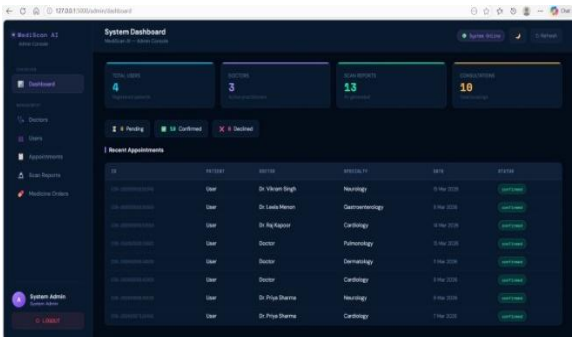


Figure 12:Admin Dashboard

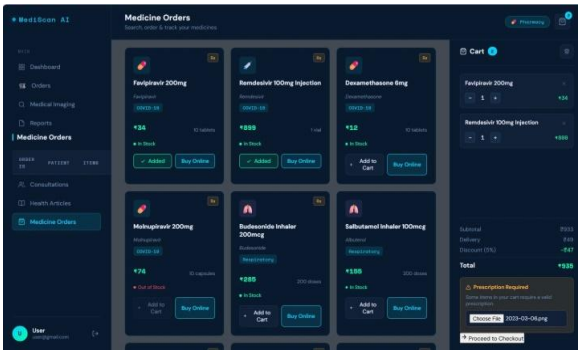


Figure 9:Medicine Orders

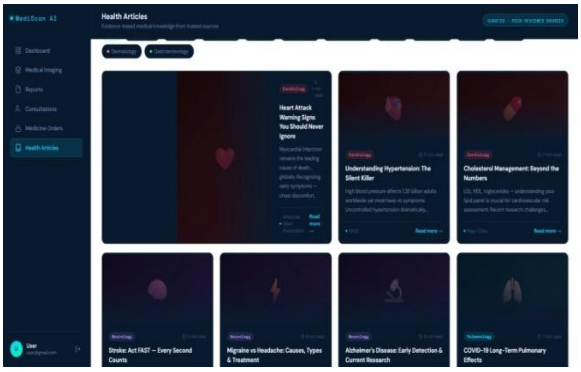
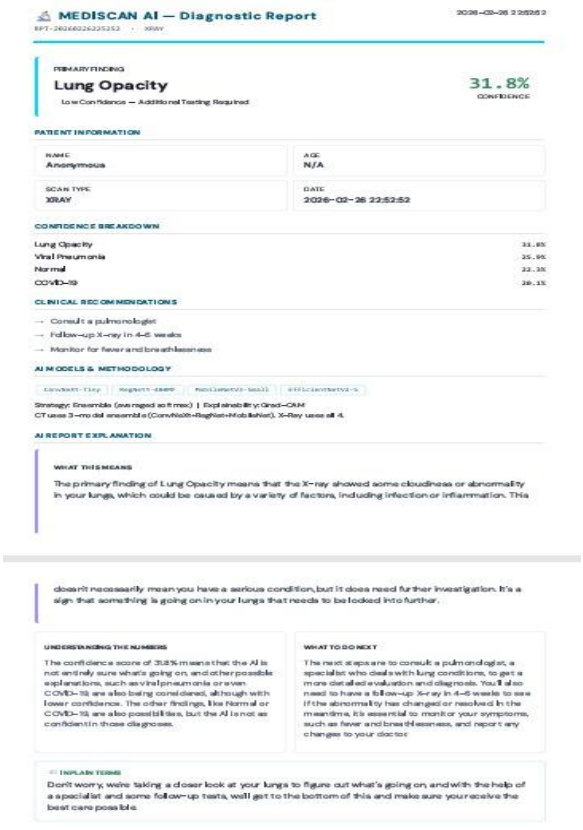


Figure 13:Health Articles

Figure 10:Chatbot Assistant



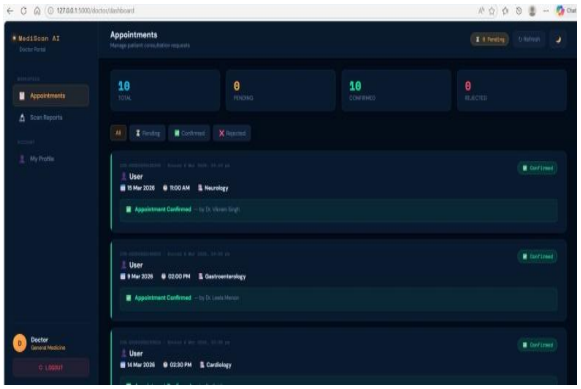


Figure 11:Doctor Dashboard

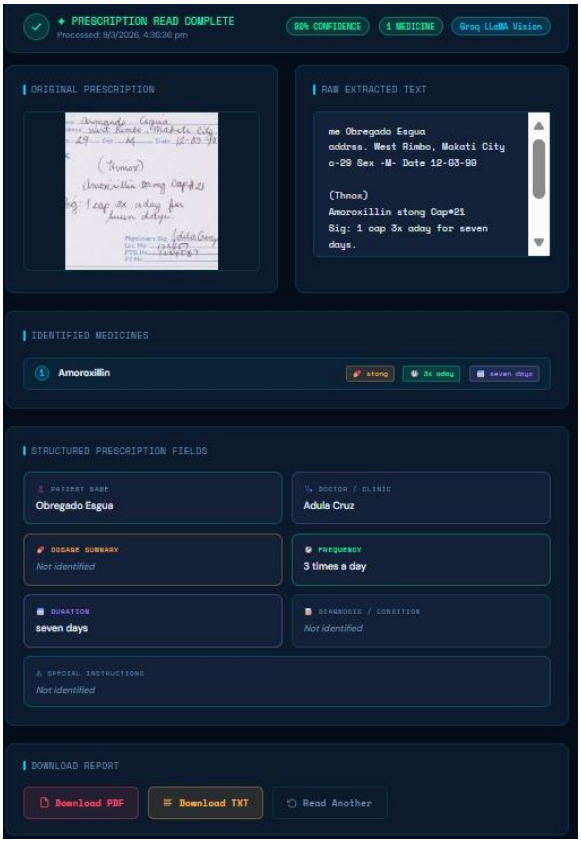


Figure 15:Prescription Reader

The results obtained from the experiments clearly prove that the accuracy of the MediScan AI is at the state-of-the-art level for all scan types. The ConvNeXt-Tiny model was found to have the highest average accuracy at 99.3%. All three models obtained higher accuracy than 98% for each scan type, proving the effectiveness of the transfer learning approach.It was also observed that the accuracy of the Kidney CT scan was perfect, at 99.7%, for all three models. This proves that the features are easily learnable for the CNN architecture. The accuracy obtained for the Pneumonia X-Ray scan (98.7%) was comparable to the leading research systems, such as MedViTV2 (98.2%), and also provided the benefits of real-world features that are absent in the pure research system.The accuracy of the proposed system (99.7%) was comparable to the accuracy of the EfficientNet B0 + CBAM (99.51%) system in the reference research paper, and the proposed system provided the benefits of real-world features such as the full web interface, Grad-CAM, report generation, doctor consultation booking, and the ability to diagnose 41 conditions using multiple scan types, rather than just diagnosing using a single scan type. This is because, by including Grad-CAM heatmaps in the web interface, clinicians can obtain explanations of AI decisionsand understand abnormal regions directly from the scans, as per the requirements of published research. The major limitation of the existing system is the lack of formal k-fold cross-validation and noise robustness tests mentioned in the reference paper. This will be addressed in future work by implementing 5-fold cross-validation of all types of scans and evaluating the performance of the system in the presence of various types of noise.

VI. CONCLUSION

This paper introduced the web-based medical imaging system called MediScan AI, which incorporates the features of deep learning-based diagnosis, prescription OCR, teleconsultation, and medicine ordering. The system was developed using the Python Flask framework and the PyTorch library, including the integration of the ConvNeXt-Tiny, RegNetY-400MF, MobileNetV3-Small, and EfficientNetV2-S CNNs trained using the transfer learning approach and the ensemble averaging method for the soft voting strategy for the diagnosis of the six medical scan types.The experimental results showed that the proposed system was able to achieve the state-of-the-art accuracy for the diagnosis of the medical scan types. The ConvNeXt-Tiny CNN obtained the highest average accuracy of 99.3%, and the accuracy of the other CNNs was greater than 98% for the Pneumonia X-Ray, Brain CT, and Kidney CT scan classifications. The ensemble approach was able to improve the reliability of the system for the diagnosis of the medical scan types. The ensemble approach for the CT scan classification was able to achieve validation accuracy between 77.8% and 84.7%, proving the feasibility of the system for deployment using the CPU without the requirement for the use of the GPU. The integration of the Grad-CAM heat maps will enable the predictions to be interpretable and transparent to the clinicians and patients alike. The prescription OCR pipeline will be effective in extracting structured information from handwritten prescriptions using a seven-step process for enhancing the images and a vision language model. This will help to overcome the accessibility challenges that exist in the field.Comparing the existing systems with the

MediScan AI, it can be clearly observed that the MediScan AI model has the same accuracy as the existing research models like EfficientNet B0 + CBAM (99.51%) while offering much more functionality compared to the existing systems, including the detection of 41 conditions that cannot be detected by any existing research systems. The main limitation of the current system lies in the absence of k-fold cross-validation and noise robustness testing. The future scope of this system includes implementing 5-fold cross-validation for all types of scans, testing the models under various noise conditions, adding support for DICOM file formats to ensure compatibility with hospital imaging equipment, and increasing the diversity of the dataset to ensure better generalization for various patient populations. Furthermore, federated learning can be explored to ensure the privacy of the data during training models for multiple hospitals.Overall, it has been shown that it is entirely feasible to develop a multi-functional AI-powered healthcare system using open-source tools and publicly available data sets, ensuring the practicality of quality medical image diagnosis for real-world applications in resource-constrained environments.

REFERENCES

[1] L. Zhang, Z. Qiao, and L. Li, “An Evolutionary Deep Learning Method Based on Improved Heap-Based Optimization for Medical Image Classification and Diagnosis,” IEEE Access, 2024.

[2] Y. Tang, Y. Yuan, F. Tao, and M. Tang, “Cross-Modal Augmented Transformer for Automated Medical Report Generation,” IEEE Journal of Translational Engineering in Health and Medicine, 29 Jan. 2025.

[3] B. Thokar, B. Sapkota, B. R. Dawadi, and S. R. Joshi, “Medical Image Segmentation for Anomaly Detection Using Deep Learning Techniques,” IEEE Access, 9 Dec. 2024.

[4] A. Rodrigues, T. Almeida, L. Bastião Silva, and C. Costa, “Diving Into AutoML in Medical Imaging: Solution for Non-ML Practitioners,” IEEE Access, 9 Aug. 2024.

[5] M. Fontes, J. D. S. de Almeida, and A. Cunha, “Application of Example-Based Explainable Artificial Intelligence (XAI) for Analysis and Interpretation of Medical Imaging: A Systematic Review,” IEEE Access, 19 Feb. 2024.

[6] M.-J. Kim, J.-W. Chae, and H.-C. Cho, “CyclicAugment: Optimized Medical Image Analysis via Adaptive Augmentation Intensity,” IEEE Access, 16 May 2025.

[7] L. Yan, X. C. Chang, Q. Li, and G. Han, “Automated Ultrasound Diagnosis via CLIP-GPT Synergy: A Multimodal Framework for Image Classification and Report Generation,” IEEE Access, 10 Jun. 2025.

[8] P. Li, Y. Pei, J. Li, and H. Xie, “Medical Image Recognition Using a Novel Neural Network Construction Controlled by a Kernel Method Encoder,” IEEE Access, 7 Oct. 2024.

[9] Y. Liu, Y. Chen, and Y. Yu, “MGVSS-UNet: A Novel U-Net Architecture Integrating Multi-Scale Global Visual State Space for Medical Image Segmentation,” IEEE Access, 30 Oct. 2025.

[10] P. Kou, M. Ma, and H. Yang, “Application of Semi-Supervised Learning in Image Classification: Research on Fusion of Labeled and Unlabeled Data,” IEEE Access, 20 Feb. 2024.