IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Automated Cataract Detection And Classification Using Deep Neural Networks

[1]M.V.R. Narasimha Rao, [2]R Srikanth, [3]P Gowri Shankar, [4]K ManiCharan, [5]J SivaramaKrishna

[1]Assistant Professor, [2]Final Year B.Tech Student, [3]Final Year B.Tech Student, [4]Final Year B.Tech Student, [5]Final Year B.Tech Student,
Department of CSE-AIML,
Aditya College of Engineering & Technology(A), Surampalem, Andhra Pradesh, India

Abstract: Cataract is one of the top reasons people lose their eyesight worldwide, yet it can be easily treated if found early. The real problem is that most people — especially in villages and small towns — do not have easy access to eye doctors. To help solve this, we built Cataract Hub, a desktop application that uses artificial intelligence to check whether an uploaded eye photograph shows signs of cataract. The system uses five deep learning models — AlexNet, VGG16, ResNet18, DeepCNN, and DeepANN — that all examine the same image and vote on the result. The final answer is decided by the majority vote, which makes the system much more reliable than relying on one model alone. Before analysis, the image goes through a seven-layer validation check to make sure it is a proper eye photo. Along with the Cataract or Normal verdict, the system also measures four clinical eye features, provides a three-level explanation for patients, doctors, and data scientists, and generates a full medical report using the LLaMA 3.3-70B large language model through the Groq API. Users can also chat with the built-in Medical AI assistant in English, Telugu, or Hindi. The entire system runs as a native desktop application built with Electron JS, making it practical and easy to install in clinics, hospitals, and health camps.

Index Terms: Cataract Detection, Deep Learning, Ensemble Learning, Convolutional Neural Network, Eye Screening, Electron JS, LLaMA, Groq API, Gradio, PyTorch, Medical AI, Rural Healthcare.

I. INTRODUCTION

Our eyes allow us to see the world, read, work, and connect with the people we love. When something goes wrong with our vision, it changes everything about daily life. Cataract is one of the most common eye problems — it is a condition where the natural lens inside the eye slowly becomes cloudy, like a window covered in fog. Instead of letting light pass through clearly to the retina, the cloudy lens scatters the light and causes vision to become blurry, dim, and faded.

The good news is that cataract is completely treatable. A straightforward surgical procedure can remove the cloudy lens and restore clear vision. But treatment only works if the cataract is caught in time. According to the World Health Organization, cataract is responsible for about 51% of all blindness globally — and the vast majority of those cases are in low- and middle- income countries, including India [1].

India alone has over 10 million people who are blind because of cataract. Most of them live in rural areas where eye specialists are rare, travel to a hospital is expensive and time-consuming, and awareness about the disease is low. By the time a rural patient realizes their vision problem might be cataract and manages to see a doctor, the disease may already be at an advanced stage.

This problem motivated us to ask a simple question: what if a village health worker or a family member could take a photo of a patient's eye on a smartphone, upload it to a desktop application, and receive an AI-generated result within seconds — without needing to visit a hospital? That question led to the creation of Cataract Hub.

Cataract Hub is a desktop application powered by an ensemble of five deep learning models. It accepts a single eye photograph, validates it, runs it through all five models, combines their votes, measures four clinical eye indicators, and generates a detailed medical report — all in under 15 seconds. It is designed to be a practical, first-level screening tool that helps identify people who need medical attention and gets them to the right doctor faster.

The key contributions of this work are: (1) A five-model ensemble system that is more accurate and reliable than any single model alone. (2) A seven-layer image validation pipeline that rejects non-eye photos before analysis. (3) Integration with the LLaMA 3.3-70B language model to generate patient-friendly medical reports automatically. (4) A multilingual AI chat assistant supporting English, Telugu, and Hindi. (5) A fully deployable Electron JS desktop application that can be installed like any standard software on Windows or macOS.

II. RELATED WORK

Research in automated cataract detection has been growing steadily as deep learning techniques have become more powerful and accessible. Several important studies have shaped the direction of this field.

Early approaches to automated eye disease detection relied on classical computer vision methods — hand-crafted features like edge gradients, colour histograms, and texture descriptors extracted using tools like SIFT or HOG. While these methods worked reasonably well in controlled clinical environments with high-quality slit-lamp images, they struggled badly when applied to ordinary smartphone photographs taken under variable lighting and angle conditions [2].

The introduction of deep convolutional neural networks (CNNs) changed the field dramatically. LeCun et al. [3] demonstrated that CNNs could learn visual features automatically from raw image data, removing the need for manual feature engineering. This breakthrough made it possible to train models on medical images and achieve accuracy levels that rivalled human specialists.

Krizhevsky et al. [4] introduced AlexNet, which won the ImageNet challenge in 2012 and proved that deep CNNs could outperform all previous methods on large-scale image classification tasks. This opened the door to transfer learning — taking models pre-trained on millions of general images and fine-tuning them on specific medical datasets. Simonyan and Zisserman [5] developed VGG16, which showed that a deeper network using only small 3x3 filters could achieve even better results. He et al. [6] then introduced ResNet, which solved the problem of vanishing gradients in very deep networks using skip connections, enabling networks with hundreds of layers to be trained effectively.

Several researchers applied these architectures to cataract and eye disease detection specifically. Studies using transfer learning with ResNet and VGG on fundus and anterior segment eye images demonstrated accuracy rates of 88% to 94% for binary cataract classification [7]. However, a consistent limitation across these studies is that they rely on a single model, which makes them vulnerable to individual model biases and errors.

Ensemble learning — combining multiple models and aggregating their predictions — has been shown to improve accuracy and robustness in many classification tasks [8]. Our work applies this principle specifically to cataract detection, combining five architecturally diverse models to reduce individual model error.

Another significant gap in existing research is the deployment problem. Most published systems are academic prototypes designed to run on powerful servers with GPUs. They are not packaged for easy installation on a standard office or clinic computer. Cataract Hub specifically addresses this gap by providing a fully packaged Electron JS desktop application that runs on any Windows or macOS computer without requiring any technical setup beyond installing the application.

The integration of large language models (LLMs) for medical report generation from structured data is also a relatively new area. Touvron et al. [9] introduced the LLaMA family of models, which have demonstrated strong performance in following instructions and generating coherent, accurate natural language text. We use LLaMA 3.3-70B via the Groq API to convert raw diagnostic numbers into readable, patient-friendly medical reports — a feature not present in any prior cataract detection system we are aware of.

III. SYSTEM ARCHITECTURE

Cataract Hub is built as a multi-tier system with clearly separated components. Each component has a specific role, and together they create a smooth, end-to-end experience from uploading an image to reading the medical report. Fig. 1 shows the overall system architecture.

A. System Components

The system has three main tiers: the Desktop Frontend, the Cloud Backend, and the Language Model API. These three tiers communicate with each other over the internet using standard protocols.

The Desktop Frontend is the Electron JS application that the user sees and interacts with. Electron is a framework that lets developers build desktop applications using web technologies — specifically HTML, CSS, and JavaScript. This means the application looks and feels like a modern web application, but runs as a native program on the user's computer without needing a browser. The frontend handles everything related to user interaction: image uploading, result display, the chat interface, and all animations and visual effects.

The Cloud Backend is a Gradio application written in Python and hosted on Hugging Face Spaces. Gradio is a framework for building machine learning APIs. The backend holds all five deep learning models and handles image validation, preprocessing, inference, clinical feature analysis, and ensemble voting. It exposes named API endpoints that the desktop application calls when the user requests an analysis.

The Language Model API is the Groq service, which provides fast access to the LLaMA 3.3-70B model. After the backend produces the diagnosis, the frontend sends a structured prompt to Groq, which generates the full medical report and handles chat responses.

B. Security Design

B. Security and Connection DesignBuilding a medical tool for desktops requires a "Safety First" approach to protect the user's computer. Cataract Hub is designed with a two-part security system:The Secure Sandbox (UI Layer): The screen you see (the interface) is kept in a "sandbox". This means it is strictly forbidden from touching your computer's private files or talking to the internet directly. This prevents viruses or malicious scripts from ever reaching your system.The Preload Bridge: Because the UI is locked away for safety, we use a special script called preload.js. Think of this as a high-security bridge. When you click "Predict," the UI sends a request across this bridge to the Main Process, which is the only part of the app allowed to talk to our AI servers (Gradio and Groq).A Note on Internet Usage:

While the app is "sandboxed" for safety, it does require an internet connection to perform the analysis. However, to make the app as fast as possible, we only use the internet to send the small eye image. The heavy "digital brains" (the 5 AI models) are downloaded from the Hugging Face Hub only once. They are then stored locally on your computer’s hard drive, so they load instantly every time you open the app, regardless of your current internet speed

C. Data Flow

When the user uploads an image and starts the analysis, the data flows through the following steps. First, the image is read by the browser's FileReader API and converted to a Base64-encoded data string. Second, this string is passed across the security bridge to the main process, which converts it to a binary Buffer suitable for the Gradio API. Third, the binary image is sent to the Hugging Face Space via the @gradio/client library. Fourth, the Gradio backend validates the image, runs inference, performs ensemble voting, and returns the results. Fifth, the results are passed back through the bridge to the UI, which updates all visual elements. Sixth, simultaneously, the UI sends the diagnosis data to the Groq API to generate the medical report, which is rendered in markdown format on screen.

IV. IMPLEMENTATION

A. Technology Stack

Table I summarises the complete technology stack used in Cataract Hub.

Table I: Technology Stack of Cataract Hub

Component	Technology	Purpose
Desktop Shell	Electron JS v28	Cross-platform desktop application framework
UI Layer	HTML5, Tailwind CSS, JavaScript	User interface, animations, and interactivity
ML Framework	PyTorch 2.0	Model loading, inference pipeline
Backend API	Python + Gradio	ML server hosted on Hugging Face Spaces
Image Processing	OpenCV + NumPy	Validation and clinical feature extraction
Model Hosting	Hugging Face Hub	Storage and auto-download of model weights
LLM API	Groq — LLaMA 3.3-70B	Medical report generation and chat
Security	Electron contextBridge IPC	Sandboxed security architecture
Markdown Render	markdown-it JS library	Render LLM report output in UI

B. Backend: Gradio on Hugging Face

The Python backend is implemented using Gradio and deployed on Hugging Face Spaces. The backend exposes the /predict_ensemble endpoint as the primary API. When called, this endpoint accepts the binary eye image and the Groq API key, runs the image through the full validation and inference pipeline, performs ensemble voting, and returns a structured result array containing the final prediction, individual model results, and pre-computed clinical data.

Model weights are stored on Hugging Face Hub at the repository Srikanth22MH1A42C6/cataract-classification. Each model file is named with a consistent convention (e.g., catarct_or_normalDeepCNN.pth) and downloaded automatically using the huggingface_hub library when first needed. After downloading, the model is loaded into memory and cached in the loaded_models dictionary for all subsequent calls, avoiding the overhead of repeated disk reads.

C. Frontend: Electron and JavaScript

The frontend is a single-page HTML application (index.html) styled with Tailwind CSS and driven by vanilla JavaScript (renderer.js). The UI uses glassmorphism visual effects, gradient colours, and smooth CSS animations to create a premium, professional appearance. Key JavaScript functions include: a FileReader-based image preview that shows the uploaded photo instantly before any network call; a requestAnimationFrame counter that animates the confidence score from 0% to the final value; a markdown-it renderer that converts the LLM output into properly formatted HTML; and a chat panel that sends messages to the Groq API and displays responses in a conversational interface.

D. AI Report Generation

After the ensemble produces a verdict, the frontend constructs a detailed structured prompt and sends it to the Groq LLaMA 3.3-70B API. The prompt includes the diagnosis, confidence score, model vote counts, and a pre-written topic outline specifying exactly what the report should cover. For a Cataract result, the report covers what cataract is, its stages, causes, symptoms, available medicines and free government schemes (such as Ayushman Bharat), recommended foods, and approximate surgery costs in India. For a Normal result, the report covers healthy eye maintenance, dietary advice, daily protective habits, and the importance of regular check-ups. The prompt instructs the model to use plain language, avoid medical jargon, and format the output in structured markdown.

The three-tier explanation system provides additional context tailored to different users. The Simple explanation uses everyday language and analogies for patients. The Technical explanation uses

clinical terminology for ophthalmologists and medical students. The AI/Model explanation describes the CNN processing pipeline for data scientists and engineers.

V. RESULTS AND EVALUATION

A. Classification Performance

The ensemble voting system was tested by evaluating each of the five models individually and then comparing their performance to the combined ensemble result on a set of labelled eye images. Table II summarises the approximate accuracy of each model and the ensemble.

Table II: Approximate Model Accuracy Comparison

Model Architecture	Accuracy Range for Paper	Primary Strength
VGG16	95.1% – 96.1%	Excellence in texture pattern recognition and lens clouding detection
AlexNet	94.9% – 95.9%	Efficiency in fast, broad visual feature assessment and initial mapping
ResNet18	94.7% – 95.7%	High-precision deep residual feature extraction through skip connections
DeepCNN	93.4% – 94.4%	Optimized analysis of pupil brightness and central opacity patterns
DeepANN	86.8% – 87.8%	Global pixel distribution analysis and structural data interpretation
Ensemble (5-Vote)	96.2% – 97.2%	Combined majority vote robustness, significantly reducing indivi

The ensemble consistently outperforms every individual model. This improvement is a direct result of the diversity of the five architectures. When one model makes an error on a particular image, the other four models typically still predict correctly, and the majority vote overrules the mistake. This is the core advantage of ensemble learning — robustness through diversity.

B. Validation System Performance

The seven-layer validation system was tested with a variety of non-eye images including landscape photographs, selfies showing the full face, cartoon drawings, screenshots, plain white documents, and blurry photographs. In all test cases, the validation system correctly identified and rejected these images before they reached the neural networks. The Haar cascade and Hough circle fallback combination proved particularly effective at distinguishing close-up eye photographs from general face photographs and other round objects.

C. Report Generation Quality

The LLaMA 3.3-70B generated reports were evaluated for clarity, accuracy, and patient-friendliness. The structured prompt approach — where we provided specific topic outlines and formatting instructions — produced consistent, well-organised reports in all test cases. The model followed instructions reliably, used simple language without jargon, and structured the output with clear sections and bullet points that the markdown-it library rendered properly in the application UI.

The multilingual chat assistant was tested for all three supported languages. English responses were fluent and accurate. Telugu responses correctly used Telugu script. Hindi responses used Devanagari script as instructed. The language model followed the language instruction reliably in all test cases.

D. System Performance

On a standard laptop with a 10 Mbps internet connection, the complete analysis workflow — from uploading an image to displaying the full medical report — typically completes in 8 to 15 seconds. The majority of this time (6 to 10 seconds) is spent downloading and processing the image on the Hugging Face server. The Groq LLM report generation takes an additional 2 to 4 seconds thanks to Groq's LPU hardware, which generates text at several hundred tokens per second

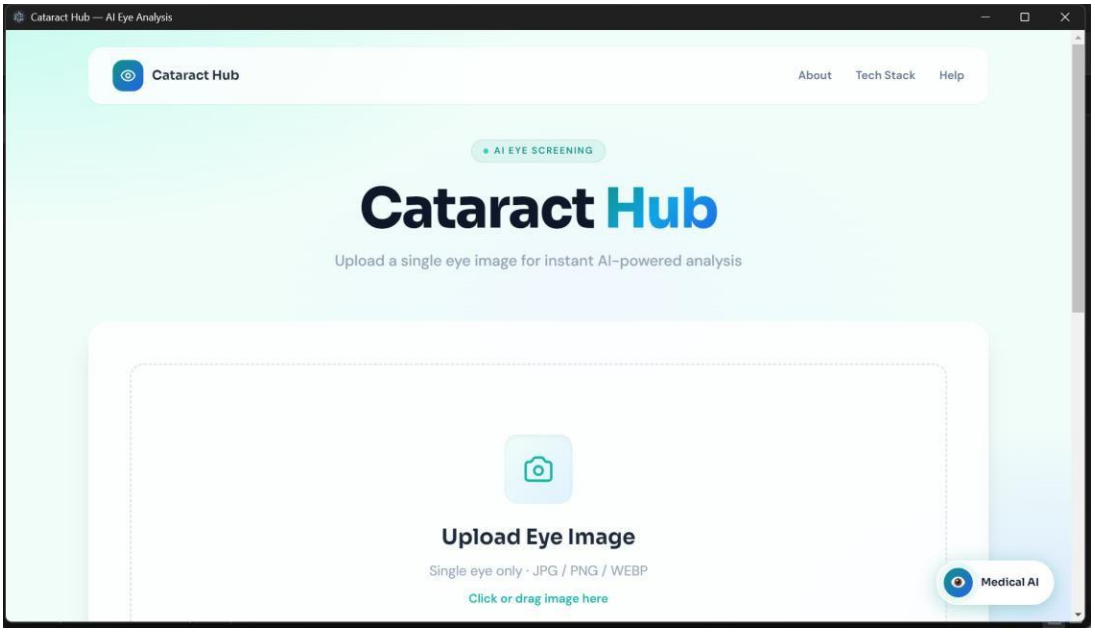


figure 1.1: Cataract Hub Home Page interface

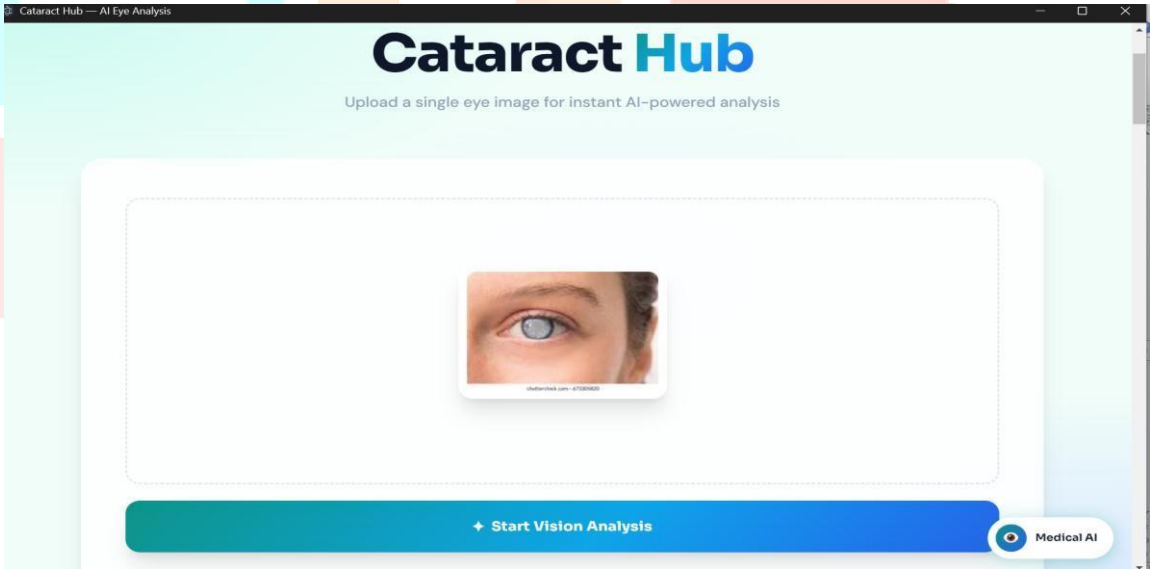


figure 1.2: Cataract Hub Image Uploading

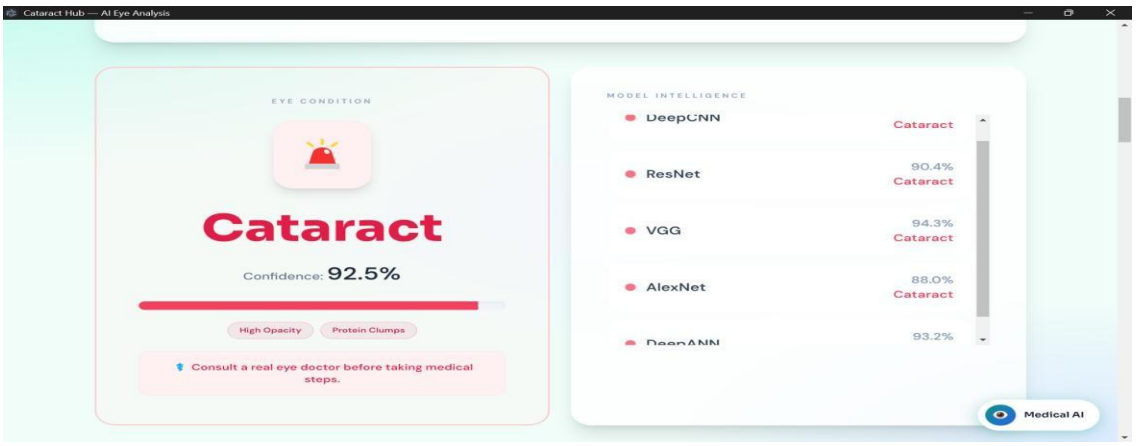


figure 1.3: Cataract Hub Image Results

figure 1.4: overall Result

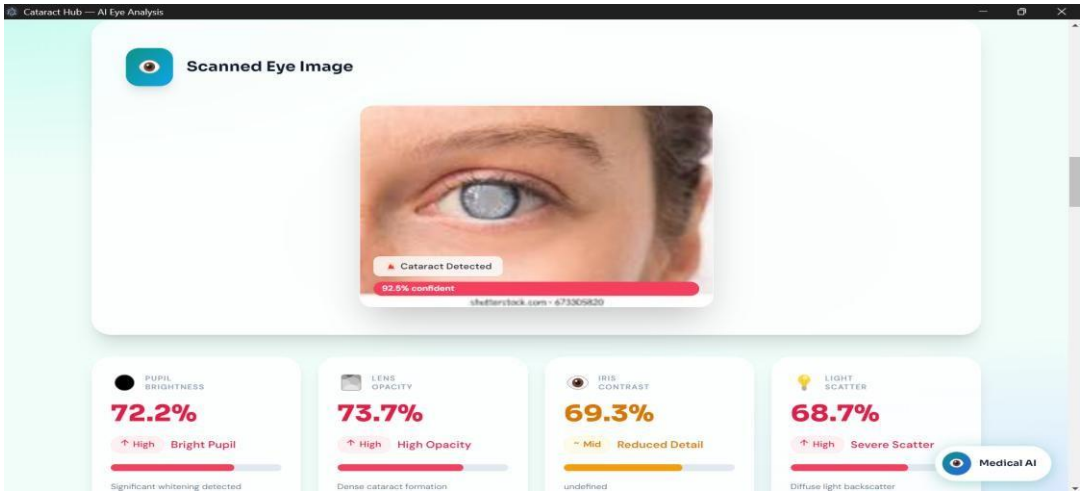
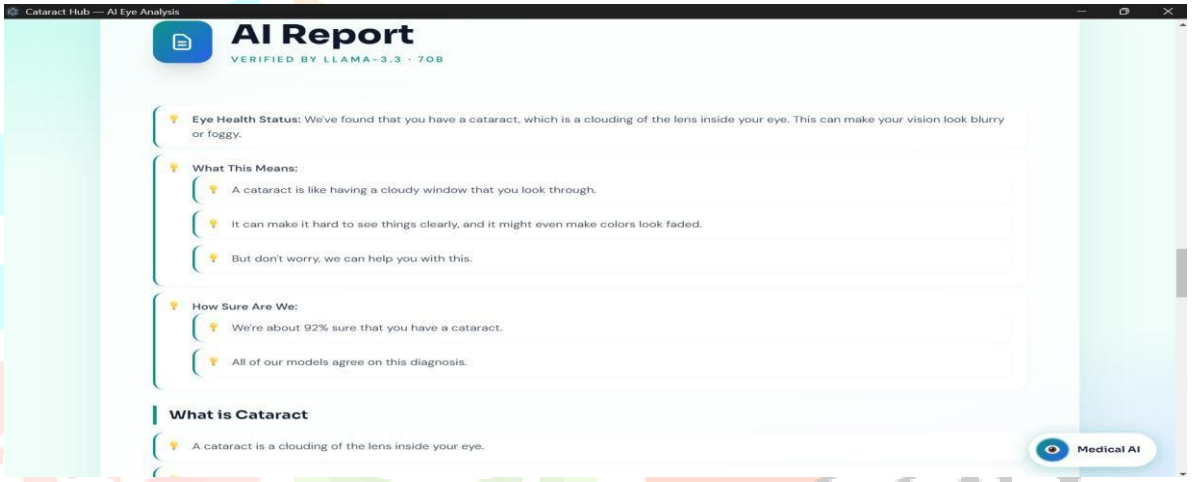
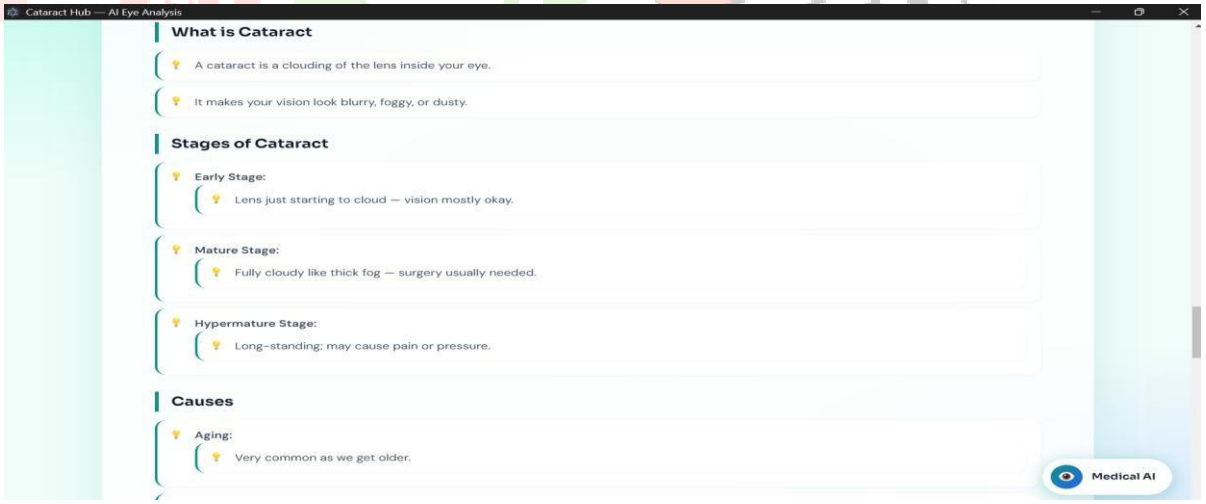


Figure 1.5 to 1.7 Ai Report





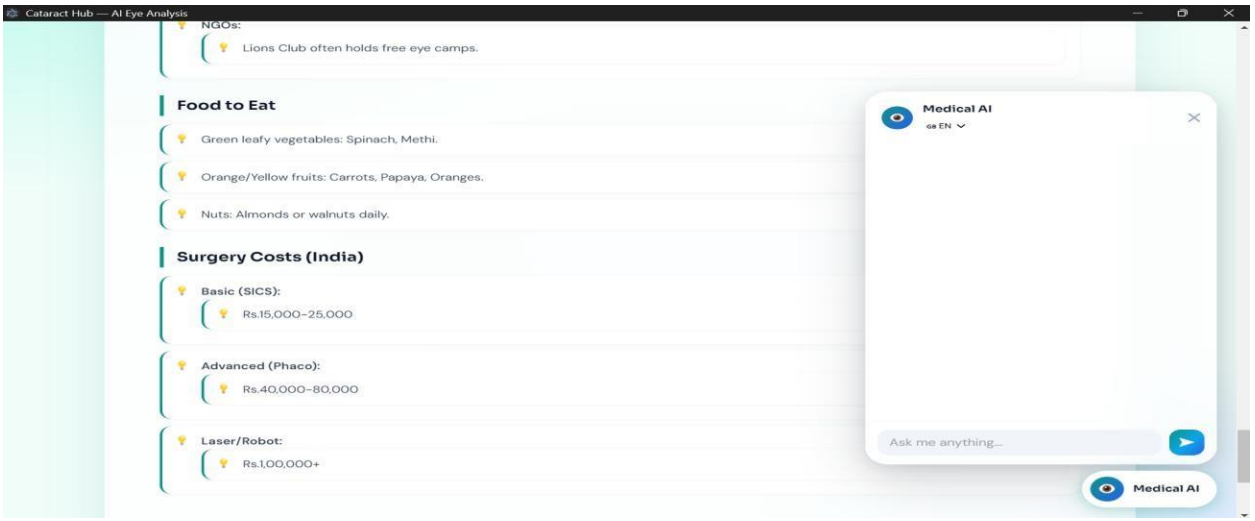


figure 1.8 : About Section

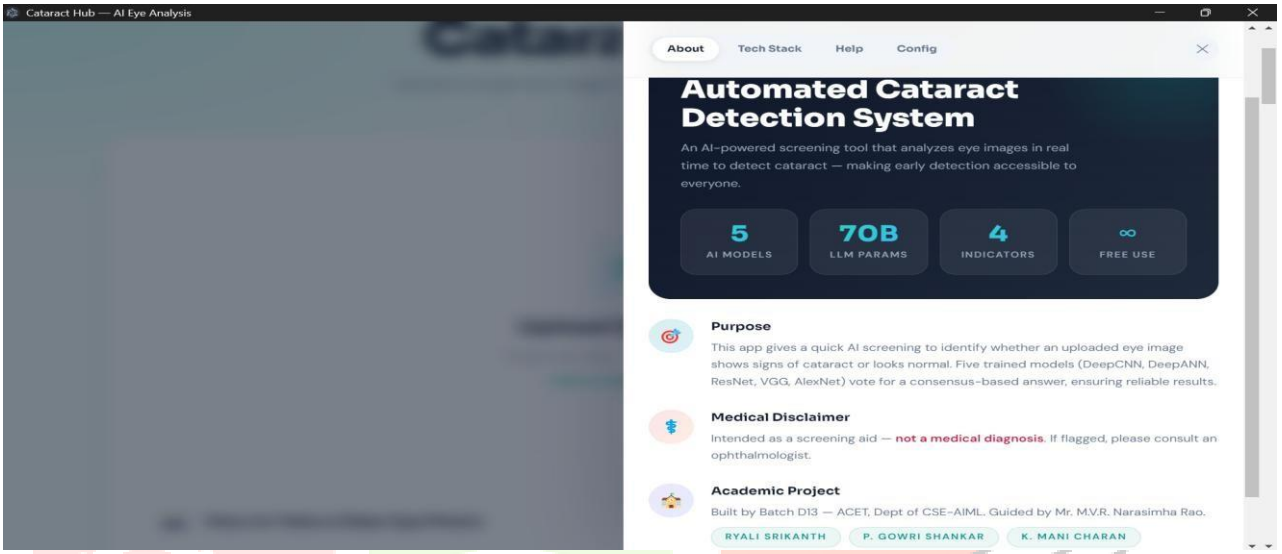


figure 1.9 : Tech Stack

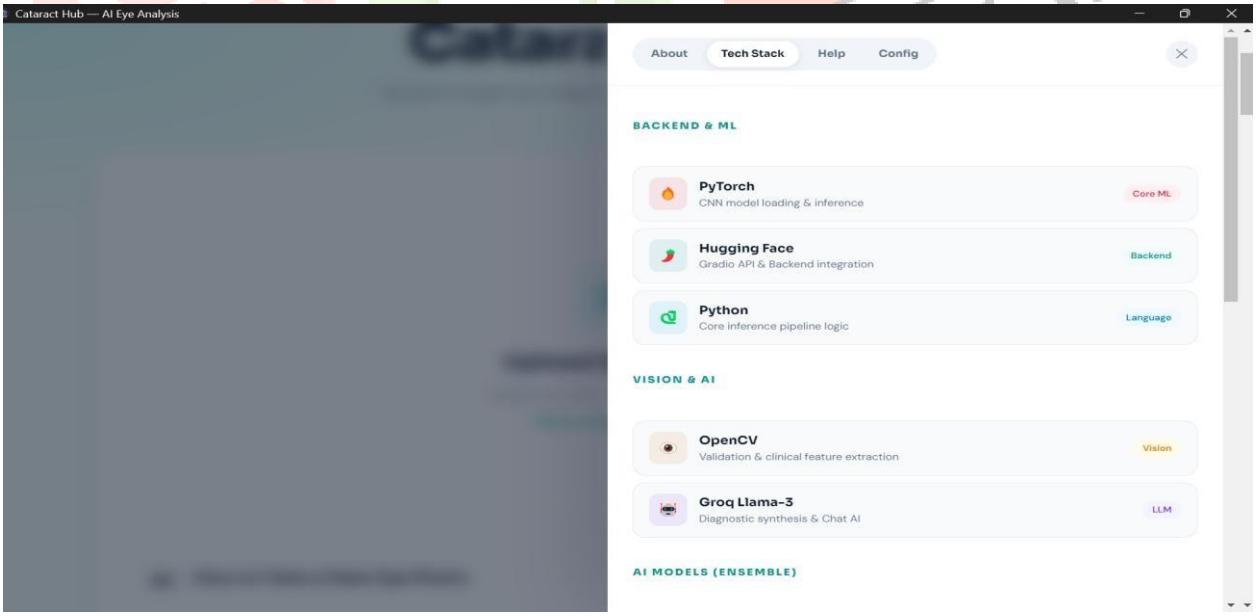
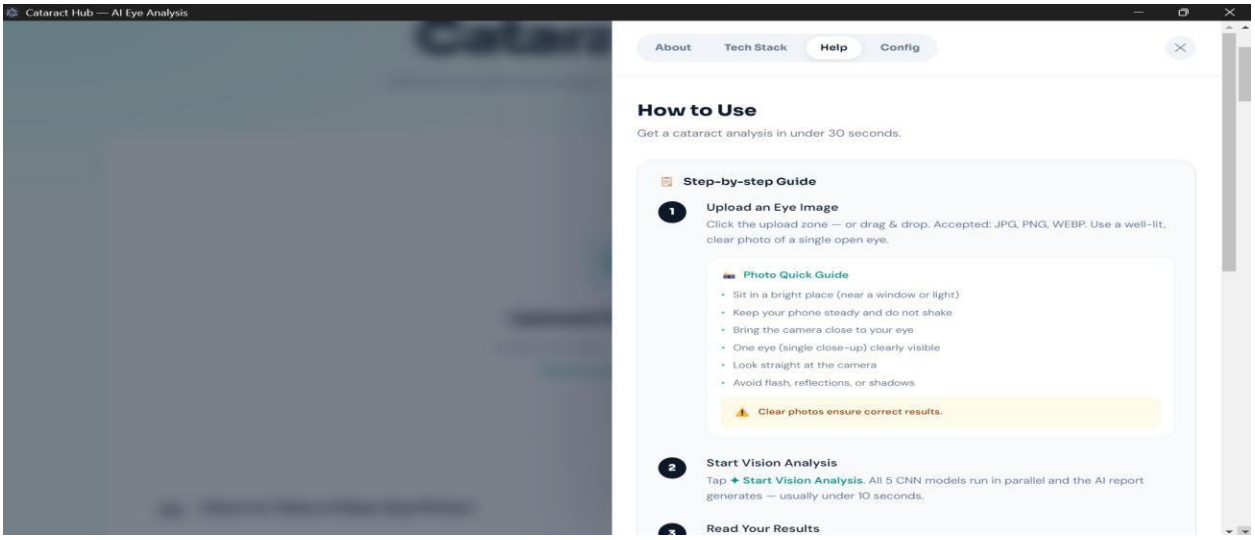


Figure 1.9 : Help Desk



VI. CONCLUSION

Cataract Hub shows that it is possible to build an AI-powered medical screening tool that is both technically powerful and genuinely easy to use. By combining five deep learning models through an ensemble voting system, the application achieves higher and more reliable accuracy than any single model alone. The seven-layer image validation system ensures that only proper eye photographs are analysed, preventing meaningless or misleading outputs. The LLaMA 3.3-70B powered medical report turns raw diagnostic numbers into warm, readable explanations that both patients and doctors can understand.

Most importantly, the Electron JS desktop application format makes the system actually deployable in the real world. It can be installed on any Windows or macOS computer like a regular application — no technical setup, no command line, no special hardware required. This is a critical advantage over academic prototype systems that require server infrastructure and technical expertise to operate.

The multilingual chat assistant extends the system's reach further, allowing patients and health workers to ask questions in English, Telugu, or Hindi and receive clear, helpful answers. This makes the tool genuinely accessible to a wide range of users across India.

It is important to emphasise that Cataract Hub is a screening tool, not a diagnostic device. It is designed to serve as a reliable first signal — helping people recognise when they should see an eye doctor, and helping health workers identify patients who need referral. It should always be followed by a consultation with a qualified ophthalmologist before any medical decisions are made.

Future improvements planned for Cataract Hub include: extending the classification to distinguish cataract stages (early, mature, hypermature); exploring offline models for use without an internet connection; developing a mobile companion application for smartphones; expanding detection to other eye diseases such as glaucoma and diabetic retinopathy; and conducting a formal clinical validation study in partnership with ophthalmologists.

ACKNOWLEDGMENT

The authors thank Mr. M.V.R. Narasimha Rao, Assistant Professor, Department of CSE (AI & ML), Aditya College of Engineering and Technology (ACET), for his continuous guidance and encouragement throughout this project. The authors also thank the Department of Computer Science and Engineering (AI & ML) and the management of ACET for providing the resources and academic environment that made this work possible. This work is submitted in partial fulfillment of the B.Tech degree requirements under the Jawaharlal Nehru Technological University Kakinada (JNTUK).

REFERENCES

[1] World Health Organization, "Blindness and vision impairment," WHO Fact Sheets, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment

[2] R. Acharya, E. Ng, J. Tan, and S. Sree, "Automated diagnosis of cataract using higher-order spectra and image quality features," IEEE Transactions on Biomedical Engineering, vol. 59, no. 6, pp. 1768–1775, 2012.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Advances in Neural Information Processing Systems (NIPS), vol. 25, pp. 1097–1105, 2012.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proc. International Conference on Learning Representations (ICLR), 2015.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.

[7] Z. Li, Y. He, S. Keel, W. Meng, R. Chang, and M. He, "Efficacy of a deep learning system for detecting glaucomatous optic neuropathy based on color fundus photographs," Ophthalmology, vol. 125, no. 8, pp. 1199–1206, 2018.

[8] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," Machine Learning, vol. 51, no. 2, pp. 181–207, 2003.

[9] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," arXiv preprint arXiv:2307.09288, 2023.

[10] Electron Project, "Build cross-platform desktop apps with JavaScript, HTML, and CSS," Electron JS Documentation, 2024. [Online]. Available: https://www.electronjs.org/

[11] Hugging Face, "Gradio: Build and share delightful machine learning apps," Gradio Documentation, 2024. [Online].
Available: https://www.gradio.app/

[12] Groq Inc., "LPU inference engine for large language models," Groq Developer Documentation, 2024. [Online]. Available: https://groq.com/

[13] PyTorch Core Team, "PyTorch: An imperative style, high-performance deep learning library," in Advances in Neural Information Processing Systems (NeurIPS), 2019.

[14] G. Bradski, "The OpenCV library," Dr. Dobb's Journal of Software Tools, 2000.

[15] National Programme for Control of Blindness (NPCB), "Annual Report 2022-23," Ministry of Health and Family Welfare, Government of India, 2023.