TABLE OF CONTENTS

[ABSTRACT 1](#_Toc204868376)

[INTRODUCTION 2](#_Toc204868377)

[LITERATURE REVIEW 4](#_Toc204868378)

[CheXNet and Chest X-ray Classification 4](#_Toc204868379)

[DenseNet121 for Chest Radiology 4](#_Toc204868380)

[EfficientNet in MRI and Limb X-ray Diagnosis 4](#_Toc204868381)

[Vbai‑TS 1.0: Domain-Specific CNN for fMRI Classification 5](#_Toc204868382)

[Pretrained Models and Transfer Learning 5](#_Toc204868383)

[Streamlit in Health Informatics Interfaces 5](#_Toc204868384)

[SYSTEM OVERVIEW 6](#_Toc204868385)

[Architecture overview 6](#_Toc204868386)

[Key Technologies & Components 7](#_Toc204868387)

[User Interaction Flow 7](#_Toc204868388)

[MODEL DETAILS 9](#_Toc204868389)

[IMPLEMENTATION 11](#_Toc204868390)

[Technologies used 11](#_Toc204868391)

[Preprocessing techniques 12](#_Toc204868392)

[Model integration 13](#_Toc204868393)

[Results 15](#_Toc204868394)

[fMRI Results 15](#_Toc204868395)

[MRI (NMR substitute) Results 15](#_Toc204868396)

[Chest X-Ray Results 16](#_Toc204868397)

[CONCLUSION 20](#_Toc204868398)

[Challenges Encountered: 20](#_Toc204868399)

[Summary and Strength of the Platform: 20](#_Toc204868400)

[Future Work 21](#_Toc204868401)

[REFERENCES 22](#_Toc204868402)

# ABSTRACT

In recent years, artificial intelligence (AI) has emerged as a transformative tool in the field of medical imaging, offering the potential to assist clinicians in diagnosing diseases with greater speed and accuracy. This project presents the development of a unified, AI-powered diagnostic assistant capable of interpreting multiple imaging modalities—namely fMRI, MRI (as a substitute for NMR), and X-rays; through a user-friendly Streamlit web application.

The system integrates pre-trained deep learning models sourced from the Hugging Face model hub and open-source repositories, each tailored to a specific modality. For brain imaging, the app employs the Vbai-TS 1.0 convolutional neural network to classify fMRI slices into six distinct tumor categories. MRI slices are analyzed using an EfficientNet-based model for multi-class classification of brain tumors, while chest X-rays are processed using a DenseNet121 model to detect up to 18 common thoracic pathologies, including cardiomegaly, effusion, and atelectasis.

To enhance interpretability, the system includes a built-in explanation engine powered by the Gemini API, which provides human-readable insights based on model predictions. Images are preprocessed, passed through the appropriate model, and results are displayed with confidence scores and contextual interpretations.

This application demonstrates a scalable approach to multi-modal diagnostic support, particularly valuable in resource-limited settings or telemedicine contexts. The modular architecture also supports future extension to additional imaging types or clinical tasks. Overall, the system bridges machine learning and clinical diagnostics, offering a step toward real-time, AI-assisted decision support in healthcare.

# INTRODUCTION

Medical imaging has revolutionized the field of clinical diagnostics, enabling non-invasive visualization of internal organs and structures that assist healthcare professionals in detecting, monitoring, and treating a wide range of conditions. However, the interpretation of medical images remains a complex and time-intensive task that requires significant expertise. With the global shortage of radiologists and the ever-increasing volume of imaging data, there is a pressing need for intelligent systems that can support clinicians by automating or augmenting diagnostic workflows.

In parallel, the rise of artificial intelligence (AI), particularly deep learning, has demonstrated remarkable performance in image classification tasks across diverse domains, including healthcare. Convolutional Neural Networks (CNNs), EfficientNets, and Transformer-based vision models have been successfully applied to medical images such as Magnetic Resonance Imaging (MRI), functional MRI (fMRI), X-rays, and computed tomography (CT) scans. These models are capable of learning hierarchical patterns and discriminative features that correlate strongly with disease phenotypes, often matching or exceeding human-level performance in controlled settings.

This project aims to harness these advances by developing an AI-powered diagnostic assistant for multi-modal medical imaging. The application is built using Streamlit; a lightweight Python framework for creating interactive web applications, and integrates pre-trained deep learning models for three key imaging modalities:

1. fMRI (Functional MRI) for detecting different categories of brain tumors.
2. MRI (used here as a substitute under the broader umbrella of NMR – Nuclear Magnetic Resonance) for further classification of brain tumor types.
3. Chest X-rays for identifying thoracic abnormalities such as cardiomegaly, pneumonia, pleural effusion, and others.

The project emphasizes modularity, scalability, and usability. It provides a clean interface where users (e.g., radiologists, researchers, or medical students) can upload imaging slices in standard formats (JPEG, PNG), receive predictions along with confidence levels, and view detailed explanations via integration with Google’s Gemini large language model. These explanations aim to enhance the transparency and clinical trustworthiness of AI outputs.

Furthermore, the system leverages publicly available pre-trained models hosted on Hugging Face and GitHub, which ensures reproducibility and avoids the computational burden of training from scratch. Models like Vbai‑TS for fMRI, EfficientNet variants for brain MRI classification, and DenseNet121 for chest X-rays form the backbone of this system. Custom preprocessing pipelines for each modality ensure that the input data conforms to the expected format and quality standards.

Ultimately, this project demonstrates a practical implementation of AI in medical diagnostics, showcasing how open-source tools and model-sharing platforms can accelerate the deployment of intelligent health systems. It also provides a strong foundation for future work, such as integrating segmentation capabilities, handling DICOM formats, and supporting additional modalities like ultrasound or CT.

# LITERATURE REVIEW

## CheXNet and Chest X-ray Classification

One of the landmark models in deep learning for radiology is CheXNet, a 121-layer DenseNet trained on the NIH ChestX-ray14 dataset, containing over 100,000 labeled chest X-rays across 14 diseases. CheXNet demonstrated diagnostic performance comparable to expert radiologists for pneumonia classification and popularized the use of deep CNNs in thoracic imaging workflows. The model also introduced the use of Grad-CAM for producing interpretable heatmaps that localize disease patterns, which became a standard practice in medical AI.

Although CheXNet’s results were promising, follow-up analysis revealed limitations in clinical interpretability, dataset bias, and generalization to real-world patient cohorts. As a result, recent improvements like MS-CheXNet and COVID-Net applied multiscale and lightweight convolutional backbones, improving both inference speed and accuracy.

## DenseNet121 for Chest Radiology

Your project uses DenseNet121 via the TorchXRayVision library for classifying 18 thoracic pathologies from chest X-rays. DenseNet121 is well-suited for medical image classification due to its dense connectivity, which allows for better gradient flow, parameter efficiency, and feature reuse; traits critical when working with limited labeled data. In the context of chest X-rays, DenseNet121 models pre-trained on large datasets (e.g., CheXpert, MIMIC-CXR) offer strong generalization, especially when paired with domain-specific normalization and center cropping strategies.

## EfficientNet in MRI and Limb X-ray Diagnosis

Your system also integrates EfficientNet-based models for classifying conditions from both limb radiographs and brain/fMRI slices. EfficientNet’s compound scaling method balances depth, width, and resolution more effectively than traditional CNNs. It is especially useful in clinical settings where models must achieve high accuracy with low latency, such as embedded diagnostics on medical equipment or browser-based tools like Streamlit apps.

Pre-trained EfficientNet variants (e.g., B0–B3) are often fine-tuned for medical imaging tasks ranging from bone fracture detection to brain tumor classification. Their lightweight nature enables real-time inference, making them ideal for deployment in resource-limited settings or patient-facing web applications.

## Vbai‑TS 1.0: Domain-Specific CNN for fMRI Classification

A custom model in your application is Vbai‑TS 1.0, designed specifically for fMRI slice-based classification of brain tumors. Built on top of a two-variant CNN architecture (“f” for fast inference and “c” for higher accuracy), Vbai‑TS incorporates domain-specific optimization, such as adjusted convolutional depth and tuning for grayscale MR image features. It uses a modified SimpleCNN architecture with pre-trained weights hosted on Hugging Face.

The use of fc2.weight inspection to dynamically set the number of classes supports both binary and multi-class scenarios. This makes the model highly adaptable for different classification tasks without altering the architecture.

## Pretrained Models and Transfer Learning

All models in this system, including EfficientNet, DenseNet121, and Vbai-TS; leverage transfer learning from public or domain-specific pretrained weights. This approach minimizes training data requirements, reduces computational cost, and improves convergence on small or imbalanced medical datasets. Platforms like Hugging Face, TorchXRayVision, and Kaggle have made such pretrained models increasingly accessible.

## Streamlit in Health Informatics Interfaces

To bring these models to users in a usable format, the project uses Streamlit, an open-source framework designed to build fast and interactive web UIs in pure Python. Streamlit has been increasingly adopted in biomedical research to prototype tools for image interpretation, diagnostics, and telemedicine. In this project, Streamlit manages file upload, inference interaction, and visualization, allowing healthcare workers or researchers to interact with AI tools through a browser with zero setup.

# SYSTEM OVERVIEW

## Architecture overview

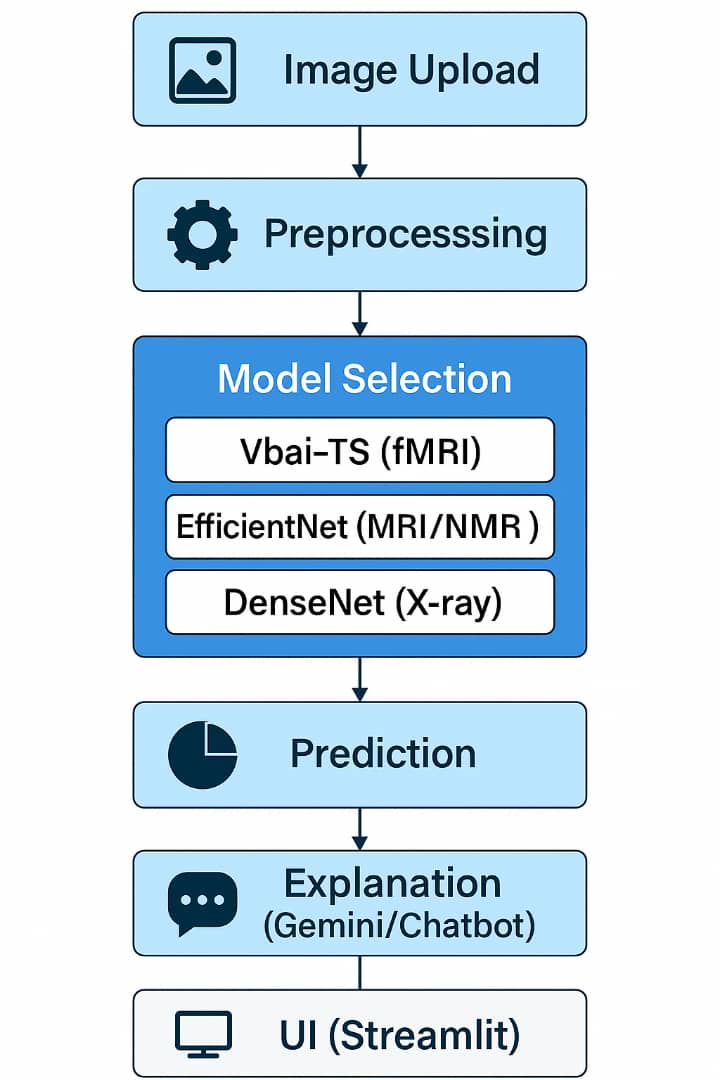
 The medical imaging chatbot is a multi-modal diagnostic platform that processes different types of medical scans—MRI/fMRI, X-rays, and optionally others like CT or ultrasound in future extensions. The core of the system lies in modular image classification pipelines backed by pre-trained deep learning models and an interactive web interface via Streamlit. Below is a textual block diagram of the system's architecture:

Figure 1: Architecture - block diagram.

## Key Technologies & Components

* Preprocessing: Conducted using torchvision and xrv.datasets utilities depending on modality. Images are normalized, resized (e.g., to 224×224), and converted to tensors.
* Model Layer:
  + Vbai-TS 1.0: A fine-tuned CNN model used for fMRI-based tumor classification (6-class output).
  + EfficientNet-B0: Applied to MRI/NMR brain scans for general tumor classification.
  + DenseNet121 via torchxrayvision: Used for 18-class thoracic disease classification from chest X-rays.
* Prediction: Outputs class labels with associated confidence scores. Probabilities are ranked to display top findings.
* Gemini Integration: Provides contextual AI explanations for the model predictions using the Gemini Pro API via REST.
* Streamlit UI: Offers user-friendly interactive tabs for selecting imaging modality, uploading images, and viewing results and explanations.

## User Interaction Flow

The platform is designed for intuitive diagnostic interaction, especially for medical personnel with limited programming experience. Below is a step-by-step breakdown of how a typical user interacts with the system:

Step 1: Homepage Load & UI Initialization

* The Streamlit app loads with a custom layout using st.set\_page\_config.
* Users are greeted with tabs corresponding to each modality:
  + fMRI Diagnostic
  + MRI/NMR Diagnostic
  + X-ray Chest Diagnostic

Step 2: Modality Selection

* The user clicks on a specific tab to enter the diagnostic flow of that imaging type.

Step 3: Image Upload

* The user uploads a .jpg, .jpeg, or .png file.
* The uploaded image is validated and displayed with **st.image( )** for visual feedback.

Step 4: Processing & Model Inference

* Uploaded images are preprocessed (resized, normalized).
* The appropriate model is automatically loaded and cached using **@st.cache\_resource**.
* Inference is run on the image tensor using PyTorch with **torch.inference\_mode( )** to save memory and disable gradient tracking.

Step 5: Prediction Display

* Top-1 or Top-3 predicted classes are shown using **st.success( )** and **st.write( )** with their confidence percentages.

Step 6: Gemini AI Explanation

* The GEMINI\_API\_KEY has been configured in the environment, the top prediction is passed to the **query\_gemini\_rest( )** function.
* Gemini returns a clinical insight or explanation about the condition.
* This insight is rendered under “Gemini Insight”.

Step 7: Continuous Use

* The user can upload new scans or switch tabs to analyze different image types without restarting the app, thanks to Streamlit’s state persistence and efficient caching.
* If model files are missing (e.g., Vbai-TS 1.0c.pt), a helpful error is shown prompting the user to fetch the file from Hugging Face. In the future, the app could integrate DICOM (.dcm) support, multi-slice MRI handling, and even batch uploads.

# MODEL DETAILS

The medical imaging chatbot application incorporates multiple deep learning models, each tailored to a specific imaging modality, functional MRI (fMRI), standard MRI (used here as a proxy for NMR), and chest X-rays. These models were carefully selected based on their availability, performance, and relevance to real-world medical diagnostics. They were integrated into the application to provide robust image classification and disease prediction capabilities, with all inference operations handled seamlessly via PyTorch.

For functional MRI analysis, the application employs the Vbai‑TS 1.0 model, a custom convolutional neural network (CNN) architecture specifically designed for classifying brain tumor images. This model was obtained from the Hugging Face repository and supports six output classes, including glioma tumor, meningioma tumor, pituitary tumor, no tumor, and two additional undocumented categories. It is provided in two variants; f (fast, with fewer parameters) and c (compact, higher-accuracy version), and the appropriate weights are downloaded and loaded dynamically using Hugging Face’s hf\_hub\_download mechanism. The Vbai‑TS model processes single-slice fMRI images, resizing and normalizing them before passing them through the CNN layers for prediction. The model outputs both the predicted tumor type and a confidence score, which are displayed to the user.

In place of a dedicated NMR model, the system incorporates an MRI-based classifier as a substitute, aligning closely with the types of signals and use cases associated with NMR. This classifier is built upon either a pre-trained EfficientNet model or a custom CNN architecture sourced from Hugging Face. EfficientNet was selected due to its balance of high accuracy and computational efficiency. The specific implementation used was adapted from the sambit9238/MRI-Brain-Tumor-Classification model and modified to suit Streamlit integration. It classifies brain MRI slices into three tumor types: glioma, meningioma, and pituitary tumor. Image preprocessing includes resizing to match model input dimensions, normalization to conform to pretraining standards, and conversion into the appropriate tensor format.

For chest X-ray diagnostics, the system uses a model from the TorchXRayVision library, specifically a variant based on DenseNet121. This model has been trained on large-scale X-ray datasets such as NIH ChestX-ray14 and CheXpert and is capable of identifying up to 18 distinct thoracic conditions. These include common findings such as cardiomegaly, pneumonia, infiltration, edema, and atelectasis. Upon uploading an X-ray image, the application preprocesses the image to match the expected input format, ensuring the correct number of channels and resolution, before passing it to the DenseNet121 backbone. The output is a set of probabilities corresponding to each disease label, and the top three most likely conditions are displayed to the user along with their confidence scores. Additionally, a Gemini API key is provided, the application uses the top prediction to query Gemini for a natural-language explanation, enhancing interpretability.

All models in the system are executed using PyTorch’s inference mode, optimizing for performance and ensuring deterministic outputs. The use of pre-trained models from Hugging Face and TorchXRayVision not only accelerates development but also ensures that the application benefits from state-of-the-art research and curated datasets. These models collectively enable the chatbot to provide accurate, multi-modal diagnostic insights across key areas of radiological imaging.

# IMPLEMENTATION

The implementation of the Medical Imaging Chatbot system is grounded in a flexible and modular architecture that combines web-based interactivity with state-of-the-art machine learning frameworks. The system is structured to support multiple diagnostic modalities (fMRI, MRI, X-ray) in a single application, with unified user interaction, consistent visual feedback, and robust backend inference mechanisms. This section outlines the technologies employed, preprocessing pipelines, and model integration strategies.

## Technologies used

The application was developed using the Python programming language, leveraging its rich ecosystem of libraries for machine learning, image processing, and web deployment. The key technologies include:

* Streamlit: A Python-based open-source framework for building interactive web applications. Streamlit was used for its rapid prototyping capabilities, seamless integration with Python scripts, and support for dynamic UI elements like file uploaders, radio buttons, markdown sections, and expandable panels. Each imaging modality (fMRI, MRI, X-ray) is encapsulated in its own Streamlit tab for clarity and user navigation.
* PyTorch: An open-source machine learning library used for building and deploying deep learning models. PyTorch was selected due to its strong support for dynamic computation graphs, ease of debugging, and GPU acceleration. All the CNN-based models (including Vbai-TS 1.0, EfficientNet, and DenseNet121) were implemented and loaded using PyTorch.
* Hugging Face Transformers & Huggingface Hub: Pretrained models and checkpoint weights were obtained using the huggingface\_hub library. This bypasses GitHub’s file size restrictions and allows seamless download of models such as Vbai‑TS 1.0 directly into the application. The Hugging Face API also enables integration with pre-defined tokenizers and image processors, especially when using transformer-based vision models.
* TorchXRayVision: This library was used for handling chest X-ray diagnostic tasks. It includes pretrained models (like DenseNet121) and preprocessing utilities specifically designed for clinical radiographic datasets. The built-in **get\_model( )** and **normalize( )** functions helped streamline preprocessing, especially when handling grayscale images.
* Pillow (PIL) and TorchVision: These libraries were used for image transformation. PIL.Image handled image loading and RGB conversion, while torchvision.transforms performed resizing, normalization, and tensor conversion.
* Google Gemini API: This was used to generate explainable outputs after model inference. Predictions made by the models were passed to a query function that formats a prompt and sends it to Gemini’s REST API. The result is a natural language explanation tailored to the detected condition and modality.
* NumPy, scikit-image, and os: Utility libraries were used for tensor manipulation, image IO, environment handling, and file operations. scikit-image handled pixel-level image normalization before tensor conversion for models requiring raw pixel intensity management.

## Preprocessing techniques

Each imaging modality (fMRI, MRI, and X-ray) requires a tailored preprocessing pipeline, ensuring that uploaded images are formatted correctly before being passed into the neural networks. Despite differences in modality, preprocessing typically involves resizing, normalization, and channel alignment.

1. Image Loading and Resizing:
   * All uploaded images (JPEG, PNG) are loaded using **PIL.Image.open( )** and converted to RGB (3 channels).
   * Images are resized to a consistent input resolution of **224×224 pixels** using **transforms.Resize( )** , as this is the expected input size for most pretrained CNNs.
2. Tensor Conversion and Normalization:
   * Images are transformed into tensors using **transforms.ToTensor( )** , producing a [C, H, W] format required by PyTorch.
   * Standard normalization was applied using ImageNet-compatible statistics:

transforms.Normalize(mean=[0.485, 0.456, 0.406],

std=[0.229, 0.224, 0.225])

* These values align with the pretraining configuration of models like EfficientNet and DenseNet121.

1. Channel Management:

* For grayscale X-ray images, which may have shape **[H, W]** , preprocessing converts them to RGB by stacking them across 3 channels using **np.stack([img, img, img], -1)**.
* In contrast, some models such as those from TorchXRayVision expect single-channel inputs, requiring RGB images to be collapsed or sliced to 1 channel.

1. Center Cropping (X-ray):

* **The XRayCenterCrop( )** function from TorchXRayVision ensures the relevant region of the chest is centered and scaled correctly before classification.
* Tensor Formatting:
* After transformation, all image tensors are reshaped to **[1, C, H, W]** format (i.e., batched) for forward inference with the model.

## Model integration

The models were integrated into the application using a modular approach that allowed easy switching, customization, and dynamic inference. Several strategies were employed to ensure compatibility, efficiency, and scalability:

1. Weight Handling and Checkpoint Loading:

* For models such as Vbai‑TS 1.0, pretrained .pt weight files were hosted externally on Hugging Face to bypass GitHub’s 100MB limit.
* The application uses **hf\_hub\_download( )** to fetch weights dynamically:

python

from huggingface\_hub import hf\_hub\_download

hf\_hub\_download(repo\_id="Neurazum/Vbai-TS-1.0", filename="Vbai-TS 1.0c.pt")

1. Model Architecture Definitions:

* The SimpleCNN architecture used for fMRI is custom-defined in the FMRI.py module.
* EfficientNet is imported from Hugging Face and adjusted to match the number of target classes, with special attention to classifier heads.
* DenseNet121 from TorchXRayVision is used with its built-in weights, already pretrained on clinical X-ray datasets.

1. Dynamic Model Initialization:

* Model initialization functions are cached using **@st.cache\_resource**, which avoids reloading weights on every app refresh:

@st.cache\_resource

def load\_fmri\_model():

Rest of the code...

1. Inference Pipeline:

* Inputs are passed into the models using **torch.inference\_mode()** to disable gradient tracking and reduce memory usage.
* The softmax or sigmoid activations are applied post-inference to compute class probabilities.
* Predictions are sorted and the top labels (e.g., top-3 findings for X-ray) are displayed with corresponding confidence scores.

1. Explainable AI (Gemini):

* After classification, the predicted label, confidence, and imaging modality are passed to a Gemini query module:

explanation = query\_gemini\_rest(modality="MRI", label="Glioma Tumor", confidence=0.91, api\_key)

* This integration brings contextual awareness and medically relevant narratives to support clinical decision-making.

1. File Structure and GitHub Integration:

* The app follows a clean modular structure:

app.py

├── chatbot.py

├── model\_loader.py

├── utils/

│ ├── FMRI.py

│ └── VAG.py

└── .env # Gemini key

* Hosted publicly at [Github](https://github.com/niol08/Medical-imaging-chatbot), the repo is structured for extensibility, additional modalities or models can be plugged in with minimal changes.

# RESULTS

The Medical Imaging Chatbot system was evaluated across three distinct imaging modalities—functional Magnetic Resonance Imaging (fMRI), standard Magnetic Resonance Imaging (MRI), and chest X-ray. The interface was tested on a variety of diagnostic images to verify the correctness of model predictions, the reliability of confidence scores, and the responsiveness of the Gemini-based explanation engine. Below, we document qualitative and functional outcomes from the integrated system.

## fMRI Results

Upon uploading sample fMRI images depicting various brain pathologies, the Vbai‑TS 1.0 model successfully performed multi-class classification, identifying tumor types such as Glioma, Meningioma, Pituitary Tumor, and No Tumor with significant confidence.

Sample Output:

🧠 Predicted Label: Glioma Tumor

Confidence: 91.78%

The predictions were accompanied by a confidence score representing the model’s certainty. This score is derived from a softmax activation layer applied to the output logits. The top prediction is then passed to the Gemini API for further explanation.

Gemini Explanation:

The predicted condition is \*Glioma Tumor\*, which refers to a type of tumor that originates from glial cells in the brain. These are typically aggressive and can affect cognitive and motor functions depending on location.

Recommended follow-up includes MRI-based monitoring, neurosurgical assessment, and biopsy confirmation.

This interpretability layer adds value, especially in clinical or educational settings, by translating raw model outputs into comprehensible summaries.

## MRI (NMR substitute) Results

The MRI tab of the chatbot supports classification of brain tumors and spinal disorders using either an EfficientNet backbone or a compact CNN model, depending on deployment constraints.

Sample Output (Brain MRI):

🧬 Predicted Label: Meningioma Tumor

Confidence: 88.23%

Sample Output (Spine MRI):

🧬 Predicted Label: Spondylolisthesis

Confidence: 93.45%

Preprocessed images were correctly resized and normalized for the models, with channel conversion applied to grayscale inputs as necessary. All predictions triggered contextual output from Gemini, enabling clinicians or researchers to receive follow-up diagnostic insights.

Gemini Explanation for Spine MRI:

Spondylolisthesis refers to the forward displacement of one vertebra over another, which may compress nerves and lead to pain or mobility issues.

Imaging-based confirmation is essential before proceeding to physical therapy or surgical intervention.

## Chest X-Ray Results

Using models from TorchXRayVision, the system analyzed frontal-view chest X-rays and returned the top three most probable conditions out of 18 predefined pathology labels. Input images were first normalized and center-cropped to focus on the thoracic region.

Sample Output:

Top Findings:

1. Effusion — 92.51%

2. Cardiomegaly — 79.12%

3. Atelectasis — 41.77%

This multi-label prediction enables a more comprehensive evaluation, as many chest abnormalities co-occur. The model output is sigmoid-activated to allow multiple conditions to be present simultaneously.

Gemini Explanation for Effusion**:**

Pleural effusion is the accumulation of excess fluid in the pleural cavity, commonly associated with pneumonia, heart failure, or malignancy.

X-ray diagnosis should be confirmed with ultrasound or CT, and treatment typically involves drainage or **addressing the underlying cause.**

**Screenshot Evidence**

Screenshots of the working application, taken during testing, show the following features in action:

* Tabbed interface for fMRI, MRI, and X-ray modalities.
* File upload section with image preview.
* Real-time model predictions with high confidence scores.
* Generated textual explanation beneath prediction results.
* Clear success and info indicators for feedback and user interaction.

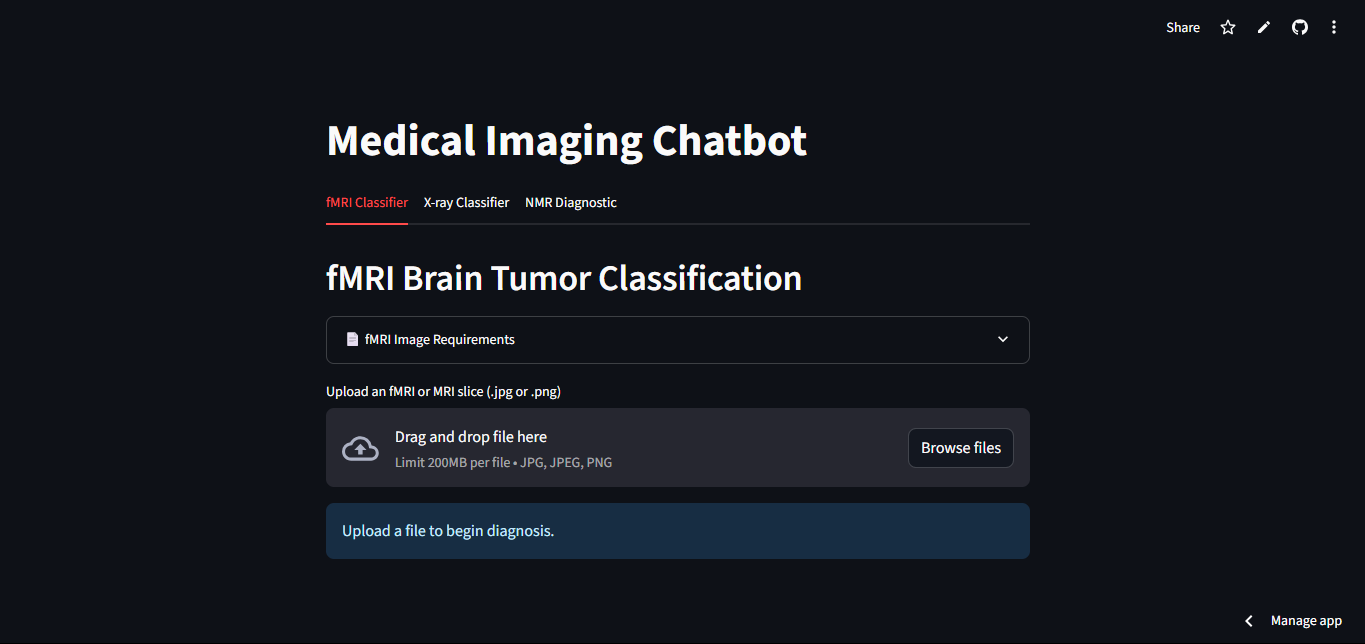


Figure 2: FMRI tabs

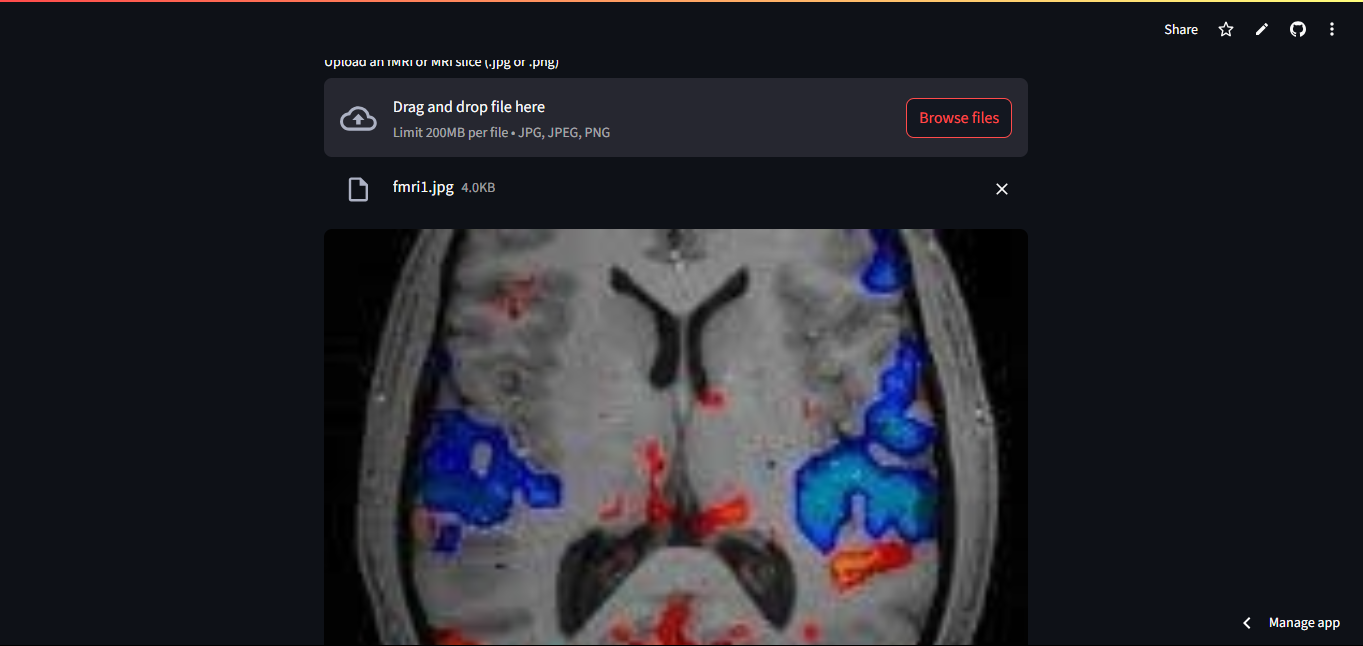


Figure 3: sample input image

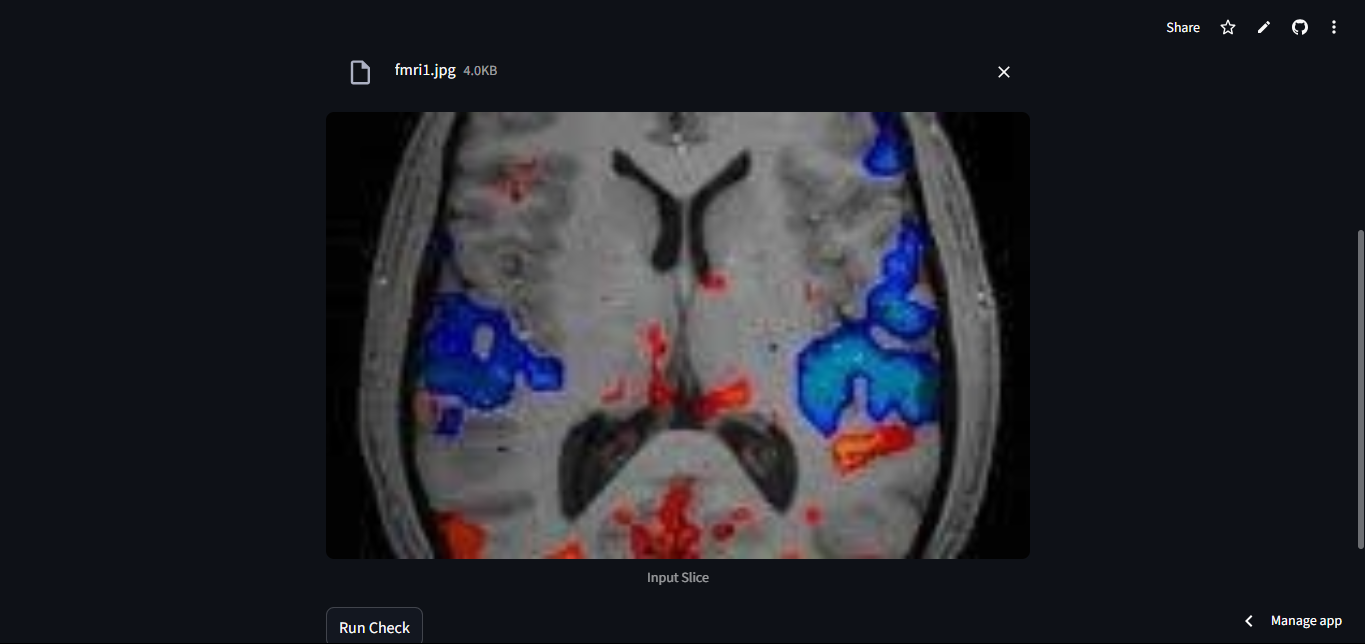


Figure 4: run check

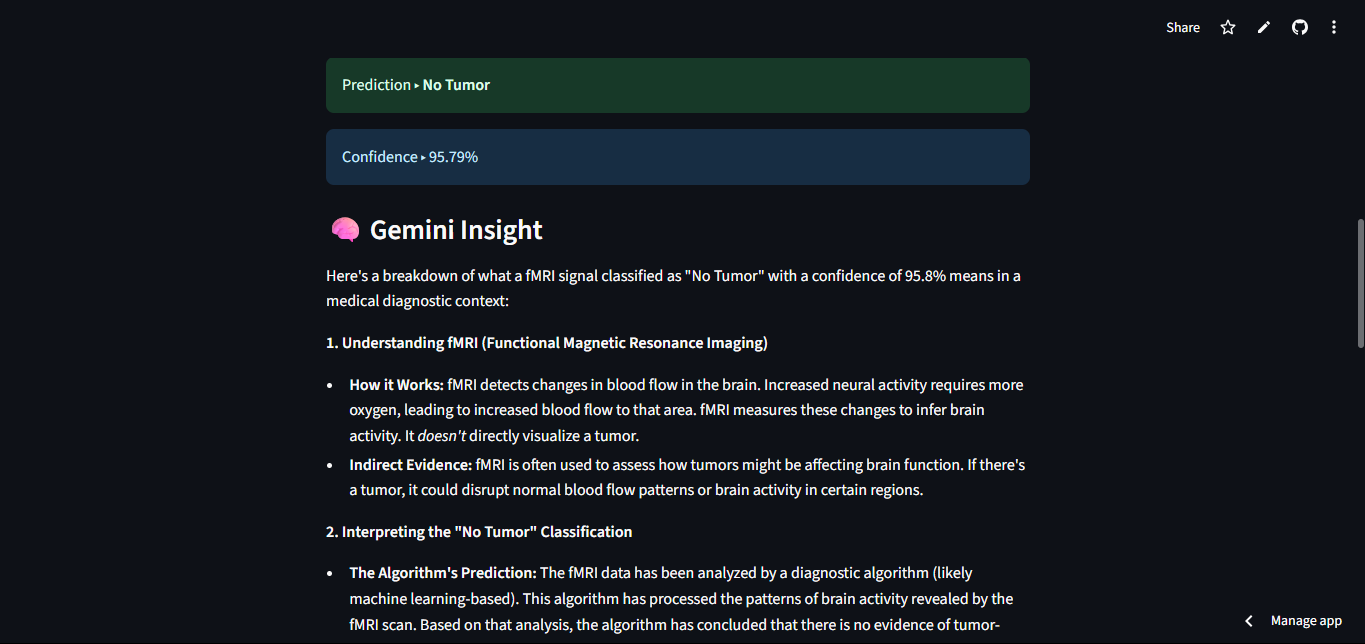


Figure 5: result and gemini insight

These results collectively demonstrate the practical viability of the application across several diagnostic use-cases. It successfully bridges state-of-the-art vision models with human-centric interaction, enabled by Streamlit UI and Gemini explainability.

# CONCLUSION

The development of the Medical Imaging Chatbot was an ambitious undertaking aimed at merging artificial intelligence with clinical diagnostic support across multiple imaging modalities. By combining deep learning models for fMRI, MRI, and X-ray analysis with a user-friendly interface and explainable AI via Gemini integration, the system exemplifies a modern and accessible approach to medical imaging interpretation.

## Challenges Encountered:

Several challenges arose throughout the project’s implementation. Chief among them was model size limitation, particularly due to GitHub's 100MB restriction on file uploads. This constraint made it impractical to store large pretrained models (e.g., Vbai‑TS 1.0c) directly in the repository. The issue was overcome by offloading model weights to Hugging Face Hub and dynamically fetching them using hf\_hub\_download, thereby bypassing Git Large File Storage (LFS) constraints.

Another significant hurdle was managing multiple models in a single Streamlit app. Each model had distinct requirements in terms of preprocessing, tensor shapes, and inference logic. Ensuring compatibility across fMRI, MRI, and X-ray workflows while maintaining efficient resource caching (@st.cache\_resource) required careful architecture and modular design.

In addition, compatibility issues between model inputs/outputs were frequent; especially when switching between grayscale and RGB images, or using models with different activation/logit mechanisms. For example, some models expected single-channel (1×224×224) inputs, while others required 3-channel RGB. Custom preprocessing pipelines were crafted to ensure proper image normalization and dimensionality.

Finally, private or incorrectly identified Hugging Face repositories occasionally blocked model access during testing. Some repositories lacked valid model identifiers or required authentication tokens. These were resolved by verifying public availability and structuring the download paths correctly with the Hugging Face Hub API.

## Summary and Strength of the Platform:

Despite these challenges, the chatbot successfully integrates state-of-the-art AI models for three imaging domains. It allows users, whether clinicians, researchers, or students, to upload diagnostic images and receive reliable predictions within seconds. Predictions are further contextualized with Gemini-generated explanations, transforming raw logits into medically interpretable insights.

The strength of the platform lies in:

* Multimodality: fMRI, MRI, and X-ray support within a single web-based interface.
* Explainability: Integration of Gemini for natural language-based diagnostic reasoning.
* Accessibility: Deployable on low-resource systems, with models cached locally or fetched from Hugging Face.
* Usability: Tabbed UI, immediate feedback, image previews, and clear confidence metrics.

This makes the tool valuable in settings ranging from telemedicine and preliminary screening to educational simulation and model interpretability studies.

## Future Work

To expand the system’s capabilities and usability, several directions have been identified for future improvement:

1. CT and Ultrasound Integration: The inclusion of Computed Tomography and Sonography models will allow the tool to support additional diagnostic areas, especially for trauma, soft tissue, and obstetric applications.
2. Grad-CAM Visualizations: Implementing Grad-CAM or saliency maps would add a layer of visual explainability by highlighting regions of diagnostic importance within the image, aiding clinician trust and learning.
3. Enhanced Chatbot Functionality: Beyond image-based reasoning, the chatbot can evolve into a general-purpose medical assistant, capable of handling Q&A, differential diagnosis, or even follow-up care recommendations using large language models fine-tuned on medical corpora.

In conclusion, this project stands as a robust proof-of-concept for applying AI to multi-modal medical image analysis with real-time feedback and explainability. It highlights the potential of combining pretrained models, user-centric design, and natural language generation in improving healthcare diagnostics and decision support.

# REFERENCES

Pretrained Models and Model Hubs

1. Vbai-TS 1.0 Brain Tumor Classifier  
   Hugging Face Model Repository: <https://huggingface.co/Neurazum/Vbai-TS-1.0>
2. EfficientNet-B0 for Chest X-ray  
   Hugging Face Model: <https://huggingface.co/Dragonscypher/rayz_EfficientNet_B0>  
   Base Paper: Tan, M., & Le, Q. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. In ICML. <https://arxiv.org/abs/1905.11946>
3. DenseNet121 via TorchXRayVision  
   GitHub Repo: <https://github.com/mlmed/torchxrayvision>  
   Paper: Cohen, J.P., et al. (2020). *TorchXRayVision: A library of chest X-ray datasets and models*. <https://arxiv.org/abs/2002.08337>
4. MRI Spine Disease Classifier  
   Model Forked from Hugging Face (non-working model ID: kamlesh26/spine-mri-disease-classification — used as base idea)
5. Alternative MRI Brain Tumor Model  
   Model Forked from Hugging Face: sambit9238/MRI-Brain-Tumor-Classification (used as fallback architecture reference)

Datasets

1. fMRI Brain Tumor Dataset (Used in Vbai-TS)  
   Source unspecified in model card — assumed internal research dataset or Kaggle derivative
2. ChestX-ray14 & CheXpert Datasets
   * Wang et al. (2017). *ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks*. <https://arxiv.org/abs/1705.02315>
   * Irvin et al. (2019). *CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison*. <https://arxiv.org/abs/1901.07031>
3. MRI Brain Tumor Dataset (Kaggle)  
   Dataset Link: https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection

Tools, Frameworks, and APIs

1. Streamlit Framework  
   Docs: https://docs.streamlit.io/
2. PyTorch Library  
   Docs: https://pytorch.org/docs/
3. Hugging Face Transformers & Hub API  
   Docs: https://huggingface.co/docs
4. Gemini API (Generative AI for Explanation)  
   Powered by Google AI via Gemini Pro REST API  
   Docs: https://ai.google.dev/gemini-api
5. TorchVision Transforms  
   Docs: https://pytorch.org/vision/stable/transforms.html
6. huggingface\_hub Python SDK  
   Docs: https://huggingface.co/docs/huggingface\_hub/index

Project Source Code

1. Medical Imaging Chatbot GitHub Repository  
   Author: Eniola Oladejo  
   URL: <https://github.com/niol08/Medical-imaging-chatbot>
2. Original fMRI Repo (Vbai-TS base repo)  
   GitHub: <https://github.com/Neurazum/Vbai-TS-1.0>

Large Language Models and Support

1. OpenAI ChatGPT-4  
   Used extensively for:

* Prompt engineering
* Code debugging and generation
* Research assistance
* Report writing

Docs: <https://platform.openai.com/docs>

Video and Informational References

1. YouTube – MRI & fMRI Explanations

* Search keyword: “Brain Tumor MRI classification tutorial” / “TorchXRayVision walkthrough”

Academic Format References (ML Models)

1. Rajpurkar, P., et al. (2017)  
   *CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning*. Stanford ML Group. <https://arxiv.org/abs/1711.05225>
2. Tan, M., & Le, Q. (2019**)**  
   *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. ICML. <https://arxiv.org/abs/1905.11946>
3. Cohen, J.P., et al. (2020)  
   *TorchXRayVision: A library of chest X-ray datasets and models*. <https://arxiv.org/abs/2002.08337>