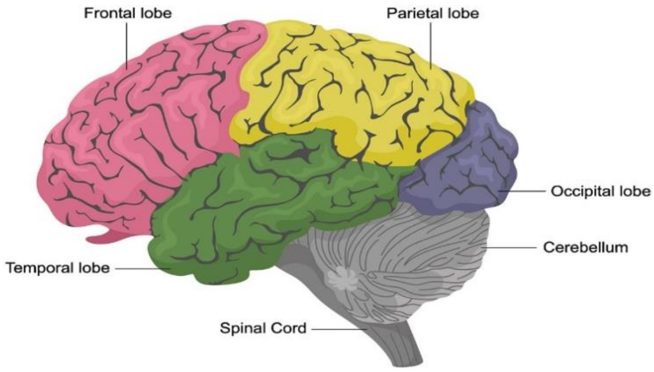
1.INTRODUCTION

The brain is a complex organ that controls thought, memory, emotion, touch, motor skills, vision, breathing, temperature, hunger and every process that regulates our body. Together, the brain and spinal cord that extends from it make up the central nervous system, or CNS.

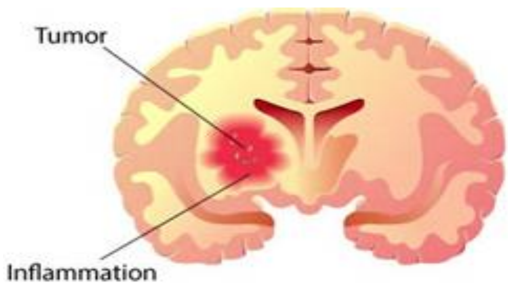


Brain

Weighing about 3 pounds in the average adult, the brain is about 60% fat. The remaining 40% is a combination of water, protein, carbohydrates and salts. The brain itself is a not a muscle. It contains blood vessels and nerves, including neurons and glial cells.

Brain Tumour

A brain tumour is a growth of abnormal cells in the brain. The anatomy of the brain is very complex, with different parts responsible for different nervous system functions. Brain tumours can develop in any part of the brain or skull, including its protective lining, the underside of the brain (skull base), the brainstem, the sinuses and the nasal cavity, and many other areas. There are more than 120 different types of tumours that can develop in the brain, depending on what tissue they arise from.



Brain Tumour

1.1 ORGANIZATIONPROFILE

The Mind IT Solution is one of the few IT system integration, professional service and software development companies in Macedonia that works with Enterprise systems and companies. As a privately owned company, The Mind IT Solution provides IT Consultancy, software design and development as well as professional services and hardware deployment and maintenance to the following verticals:

- Government(Local and Central)
- Financial Services(insurance ,banking and clearing house)
- Telecommunications
- Energy and Utilities
- HealthCare
- Education

The Mind IT Solution is located in Skopje, Macedonia (South-Eastern Europe), offering fully fledged services for software development and engineering empowering effective near-shore management to its clients. Recognized both on local and regional markets since 1995, initially known as ICL and later acquiring regional Fujitsu Services partner, Infinite continues to operate with higher client expectancy and flexible solution offers. The Mind IT Solution is a pioneer in Interactive Virtual Teams (IVT) which enables the client and the service provider to establish instant and successful communication channels and support off-shoring and near-shoring business models, thus keeping the deadlines on time and on track.

Infinite's differentiation point comes with three simple principles:

- True collaboration with customers and partners
- Complete understanding of customers business
- Persistence in finishing the job whatever it takes.

The Mind IT Solution as a Microsoft Dynamics partner, has a unique value in development, integration and implementation of MS Dynamics AX, .Net, CRM, Share Point and RMS based projects in the following verticals: Defense, Retail, Finance, Manufacturing, Health Care and Education. We have highly skilled Microsoft team of Project managers, Consultants and Developers which are proven on a large-scale project with more than1.000 users.

1.2. SYSTEM SPECIFICATION

1.2.1 HARDWARE SPECIFICATION

Processors	: Intel processor 2.60 GHz
RAM	: 8 GB
Disk space	: 320 GB
OS	: Windows10and Linux*

1.1.1 SOFTWARE SPECIFICATION

Server Side	: Python 3.7.4(64-bit) or (32-bit)
Client Side	: HTML, CSS, Bootstrap
IDE	: Flask 1.1.1
Back end	: My SQL 5.
Server	: Wampserver 2i
OS	: Windows 10 64 –bit

SYSTEM STUDY

2.SYSTEM STUDY

2.1. EXISTING SYSTEM

2.1.1DESCRIPTION

Region Based Method - The region-based classification method divides an image into related areas by applying homogeneity criteria to the collection of pixels. Region based methods are classified into: region growing, region splitting and merging, thresholding, watershed and clustering.

Thresholding method

Thresholding is a straight forward and effective method for image segmentation. Thresholding is used to transform a multilevel image to a binary image. To segment image pixels into different regions, a suitable threshold is selected. The thresholding approach has two limitations: it generates only two classes and cannot be extended to multichannel images. Furthermore, threshold does not take into account an image's spatial characteristics. As a result, it is susceptible to noise.

Region growing method

This is a traditional approach in which segmentation begins with the manual sorting of seeds from the image of interest. The manual dealings to attain the seed point are the area growing's restriction. However, split-and-merge is a region-growing algorithm that does not need a seed point. Region growth is also vulnerable to noise, resulting in gaps in partitioned areas. This problem is solved by the hemitropic region-growing algorithm.

2.1.2 DRAWBACKS

- Handcrafted Features.
- Difficult to segment these brain tumor regions automatically.
- High Computational Process.
- Misclassification due to improper segmentation.
- Performance in Brain Tumor detection was not satisfactory.
- Computational complexity is severely increased.
- Time spent in feature extraction.
- False prediction of Brain Tumor Grades.

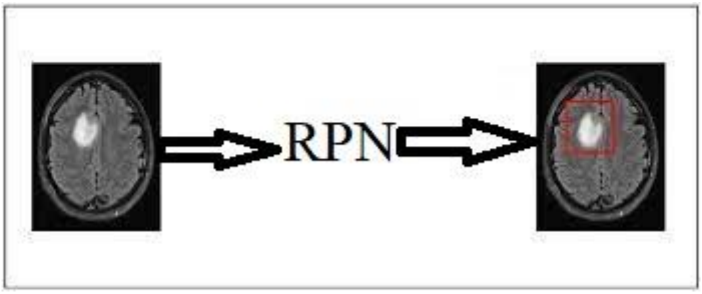
2.2. PROPOSED SYSTEM

2.2.1 DESCRIPTION

The main goal behind the development of our proposed model is to automatically distinguish people with brain tumors, while reducing the time required for classification and improving accuracy. We propose a novel and robust DL framework CNN for detecting brain tumors using MRI datasets. The proposed model is a four step process, in which the steps are named: 1). Pre-processing, 2). Features Extraction, 3). Features Reduction, and 4). Classification. In second step, it uses Grey Level Co-occurrence Matrix GLCM) technique to extract different features from the images. In third stage, Color Moments (CMs) are used to reduce the number of features and get an optimal set of characteristics. Images with the optimal set of features are passed to CNN classifiers for the classification of BT Type and their grades.

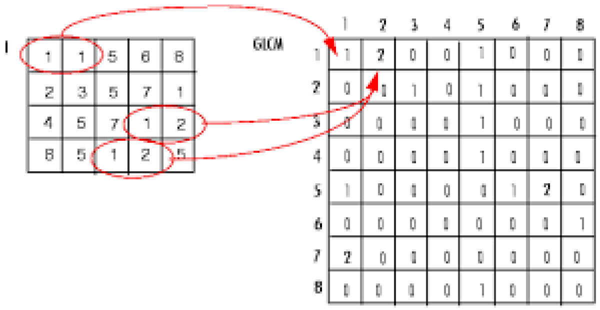
- **Region Proposal Network**

This region proposal network takes convolution feature map that is generated by the backbone layer as input and outputs the anchors generated by sliding window convolution applied on the input feature map.



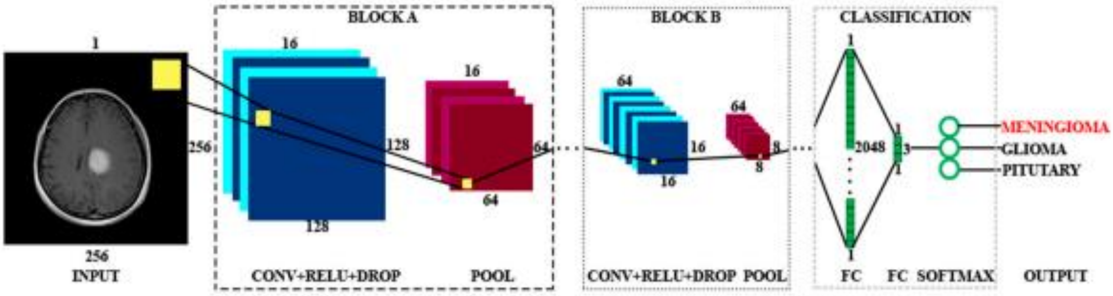
- **Grey Level Co-occurrence Matrix**

Grey Level Co-occurrence Matrix (GLCM) based texture analysis of kidney diseases for parametric variations. The investigations were carried out using three Pyoderma variants (Boil, Carbuncle, and Impetigo Contagions) using GLCM. GLCM parameters (Energy, Correlation, Contrast, and Homogeneity) were extracted for each colour component of the images taken for the investigation. Contrast, correlation, energy, and homogeneity represent the coarseness, linear dependency, textural uniformity, and pixel distribution of the texture, respectively. The analysis of the GLCM parameters and their histograms showed that the said textural features are disease dependent. The approach may be used for the identification of CKD diseases with satisfactory accuracy by employing a suitable deep learning algorithm.



- **Convolutional Neural Network (CNN)**

A CNN is a type of deep learning used to analyse visual scenes. It is characterized by having one or more hidden layers, which extract the attributes in videos or images, and a fully connected layer to produce the desired output. Whereas for the computer, the image is a 3D array (width × height × depth) of values ranging from 0 to 255. It is simply pixels of colour; if the number of channels is one, the image is grayscale, black, and white. CNN is a hierarchical structure that contains several layers.



Architecture of CNN

The basic components of the basic convolutional neural networks are: The Convolutional Layer, the Activating function, the Pooling Layer, and the Fully-connected Layer.

2.2.2 FEATURES

- A fast and accurate fully automatic method for brain tumor classification which is competitive both in terms of accuracy and speed compared to the state of the art.
- The method is based on deep neural networks (DNN) and learns features that are specific to brain tumor types.
- Segmentation technique accomplishes the better segmentation results with the maximum accuracy of 99%.
- Automatic Feature Extraction.
- Low Computational Overhead.

SYSTEM DESIGN & DEVELOPMENT

3.SYSTEM DESIGN & DEVELOPMENT

3.1 FILE DESIGN

The file design for this project will include the following:

- Dataset files: These will include the MRI brain images used for training and testing the algorithms.
- Configuration files**:** These will include the configuration settings for the DCNN and SSD algorithms.
- Annotation files**:** These will include the XML files generated using the LabelImg tool for labeling the dataset.
- Trained model files: These will include the trained models for the DCNN and SSD algorithms after the training phase.

  Output files: These will include the prediction results from the testing phase.

3.2 INPUT DESIGN

The input design for this project will involve the following:

MRI brain images: These will be the input images for the algorithms to detect and segment brain tumors.

Configuration settings: These will be the input parameters for configuring the DCNN and SSD algorithms.

3.3 OUTPUT DESIGN

The output design for this project will include the following:

Prediction results: These will be the output of the algorithms after testing on the test dataset, showing the segmented brain tumors.

Evaluation metrics: These will be the output metrics used to evaluate the performance of the algorithms, such as classification loss and accuracy.

3.4 CODE DESIGN

The code design for this project will involve the implementation of the following:

Data preprocessing: This will include code for reading, preprocessing, and denoising the MRI brain images.

Model training: This will include code for configuring and training the DCNN and SSD algorithms on the labeled dataset.

3.5 DATABASE DESIGN

Table Name: Admin					
S.no	**Field**	**Data type**	**Field size**	**Constraint**	**Description**
1	User name	Varchar	20	Null	Admin name
2	Password	Varchar	20	Null	Admin Password

Table Name: Training					
S.no	**Field**	**Data type**	**Field size**	**Constraint**	**Description**
1	Id	Int	11	Null	Id
2	Training image	Varchar	30	Null	Training image
3	Preprocessed image	Varchar	30	Null	Preprocessed image
4	Extracted feature	Varchar	30	Null	Extracted feature
5	Classified label	Varchar	30	Primary key	Classified label

Table Name: Recommendation					
S.no	**Field**	**Data type**	**Field size**	**Constraint**	**Description**
1	Id	Int	11	Null	Id
2	Classified label	Varchar	20	Foreign key	Classified label
3	Disease type	Varchar	30	Null	Disease type
4	Hospital Location	Varchar	100	Null	Hospital Location
5	Contact Details	Varchar	100	Null	Contact Details

Table Name: Patient Register					
S.no	**Field**	**Data type**	**Field size**	**Constraint**	**Description**
1	Id	Int	11	Null	Driver id
2	Name	Varchar	20	Null	Name
3	Gender	Varchar	10	Null	Gender
4	Dob	Varchar	20	Null	Dob
5	Address	Varchar	50	Null	Address
6	Mobile number	Bigint	20	Null	Mobile number
7	Email	Varchar	20	Null	Email
8	Patient Id	Int	11	Primary key	Patient Id
9	Password	Varchar	30	Null	Password
10	Register date	Timestamp	Timestamp	Null	Register date

Table Name: Testing					
S.no	**Field**	**Data type**	**Field size**	**Constraint**	**Description**
1	Id	Int	11	Null	Id
2	Patient Id	Varchar	20	Null	Patient Id
3	Test image	Varchar	30	Null	Test image
4	Classified label	Varchar	30	Foreign key	Classified label
5	Date Time	Timestamp	Timestamp	Null	Date Time

3.6. SYSTEM DEVELOPMENT

System development in brain tumor refers to the process of creating and implementing various technologies, tools, and methodologies to aid in the diagnosis, treatment, and management of brain tumors. This involves interdisciplinary collaboration among healthcare professionals, researchers, engineers, and computer scientists to develop innovative solutions that improve patient outcomes and quality of life. Here are some key aspects of system development in brain tumor:

Imaging Technologies: Advanced imaging techniques such as magnetic resonance imaging (MRI), computed tomography (CT), and positron emission tomography (PET) play a crucial role in diagnosing brain tumors. Continuous improvements in imaging technology, such as higher resolution and contrast, help in accurate tumor localization and characterization.

Computer-Aided Diagnosis (CAD): CAD systems use machine learning algorithms and image processing techniques to assist radiologists in interpreting medical images. These systems can help in early detection, segmentation, and classification of brain tumors, improving diagnostic accuracy and efficiency.

3.7. DESCRIPTION OF MODULES

In the context of brain tumor detection systems, various modules can be developed to facilitate accurate and efficient diagnosis. Each module serves a specific purpose in the overall detection process. Here's a breakdown of potential modules:

Image Preprocessing Module:

- This module focuses on enhancing the quality of medical imaging data, such as MRI or CT scans, before further analysis..
- Preprocessing aims to improve the performance of subsequent analysis algorithms by providing cleaner and more standardized input data.

Tumor Segmentation Module:

- Tumor segmentation is the process of delineating the boundaries of tumors within medical images.growing, watershed, or deep learning-based approaches, to identify and segment tumor regions.

Feature Extraction Module:

- Once tumor regions are segmented, this module extracts relevant features or characteristics from the segmented regions.
- Features may include shape descriptors, intensity histograms, texture features, or more sophisticated features derived from deep learning models.
- Feature extraction aims to capture discriminative information that distinguishes tumor regions from healthy tissue.

Feature Selection Module:

- In cases where a large number of features are extracted, feature selection becomes important to reduce dimensionality and remove irrelevant or redundant features.
- Various feature selection techniques, such as filter methods, wrapper methods, or embedded methods, can be employed to identify the most informative features for tumor detection.

Classification Module:

- The classification module utilizes machine learning or deep learning algorithms to classify segmented tumor regions as either benign or malignant.
- Common classification algorithms include support vector machines (SVM), random forests, convolutional neural networks (CNNs), or hybrid models.
- The classifier is trained on labeled data, often with features extracted from segmented tumor regions, to learn to distinguish between different tumor types.

Post-processing Module:

- This module performs additional processing steps to refine the results obtained from the classification module.
- Post-processing techniques may include morphological operations, smoothing filters, or spatial constraints to improve the coherence and consistency of detected tumor regions.

Integration and Visualization Module:

- Finally, an integration module combines the outputs of the previous modules and provides visualization tools for clinicians to interpret the results.
- This module may generate comprehensive reports, overlay tumor regions on original medical images, and provide interactive visualization interfaces for exploring detected tumors in 3D space.
- Integration and visualization facilitate easy interpretation of results and support decision-making by healthcare professionals.

SYSTEM TESTING

SYSTEM TESTING

Software testing plays a critical role in ensuring the reliability, accuracy, and safety of applications like Deep Brain for brain tumour detection and stage classification. Here are some key considerations for testing Deep Brain:

Types of Testing

1. **Unit Testing**

Unit testing involves testing individual components or functions of the software in isolation. In the case of Deep Brain, this could include testing the image pre-processing algorithms, neural network layers, and classification algorithms. Unit tests verify the correctness of these components and help identify any potential bugs or issues.

2. **Integration Testing**

Integration testing focuses on testing the interaction between different components or modules of the software. In the context of Deep Brain, integration testing would involve testing the integration of the image pre-processing pipeline, the trained deep learning model, and the tumour detection and stage classification algorithms. It ensures that the different parts of the system work together as expected.

3. **Performance Testing**

Performance testing assesses the system's behaviour and efficiency under various workloads and stress conditions. For Deep Brain, performance testing could involve evaluating the system's response time for processing brain scans, handling multiple concurrent requests, and scaling to handle increased data volumes. This testing helps identify any performance bottlenecks or resource limitations.

4. **Accuracy and Validation Testing**

Given the critical nature of brain tumour detection and stage classification, it is crucial to assess the accuracy and validation of Deep Brain. This involves testing the system's ability to correctly detect tumours, classify their stages, and provide reliable results. Testing may involve using a diverse set of brain scans with known tumour cases and comparing the system's output against ground truth annotations or expert opinions.

5. **User Interface (UI) and User Experience (UX) Testing**

Deep Brain may have a user interface component for medical professionals to interact with the system. UI and UX testing involve evaluating the usability, intuitiveness, and responsiveness of the interface.

Test Cases

Test Case 1:

Test Case ID: TBDD001

Input: A brain MRI scan of a patient with a confirmed tumor.

Expected Result: Deep Brain detects the tumor accurately and classifies its stage as malignant.

Actual Result: Deep Brain detects the tumor accurately and classifies its stage as malignant.

Pass: Yes

Test Case 2:

Test Case ID: TBDD002

Input: A brain CT scan of a patient with no tumor.

Expected Result: Deep Brain correctly identifies the absence of a tumor.

Actual Result: Deep Brain correctly identifies the absence of a tumor.

Pass: Yes

Test Case 3:

Test Case ID: TBDD003

Input: A brain MRI scan of a patient with a benign tumor.

Expected Result: DeepBrain detects the tumor accurately and classifies its stage as benign.

Actual Result: DeepBrain detects the tumor accurately and classifies its stage as benign.

Pass: Yes

Test Case 4:

Test Case ID: TBDD004

Input: A brain MRI scan with low image quality and noise.

Expected Result: DeepBrain handles the noisy input and provides an appropriate error message indicating the poor image quality.

Actual Result: DeepBrain handles the noisy input and provides an appropriate error message indicating the poor image quality.

Pass: Yes

Test Case 5:

Test Case ID: TBDD005

Input: Multiple brain scans sent simultaneously for processing.

Expected Result: Deep Brain handles the concurrent requests efficiently and provides for each scan.

Actual Result: Deep Brain handles the concurrent requests efficiently and provides accurate results for each scan.

Pass: Yes

Test Case 6:

Test Case ID: TBDD006

Input: A brain MRI scan with an unusual tumor shape.

Expected Result: Deep Brain accurately detects the tumor despite its irregular shape and provides the correct stage classification.

Actual Result: Deep Brain accurately detects the tumor despite its irregular shape and provides the correct stage classification.

Pass: Yes

Test Case 7:

Test Case ID: TBDD007

Input: A brain scan with no tumor but with artifacts in the image.

Expected Result: DeepBrain correctly identifies the absence of a tumor and alerts the user about the presence of artifacts in the image.

Actual Result: DeepBrain correctly identifies the absence of a tumor and alerts the user about the presence of artifacts in the image.

Pass: Yes

Test Case 8:

Test Case ID: TBDD008

Input: A brain scan from an older imaging machine with a different format.

Expected Result: Deep Brain handles the input from the older machine, converts it to the required format, and provides accurate results.

Actual Result: Deep Brain handles the input from the older machine, converts it to the required format, and provides accurate results.

Pass: Yes

Test Report

Test Title: Deep Brain - Brain Tumour Detection and Stage Classification

Introduction

This test report provides an overview of the testing conducted for Deep Brain, a brain tumour detection and stage classification system utilizing deep learning techniques. The objective of the testing was to evaluate the accuracy, reliability, and performance of the system in detecting brain tumours and classifying their stages.

Test Objective

The main objectives of the testing were as follows:

- Verify the accuracy of tumour detection and stage classification.
- Assess the system's ability to handle different types of brain scans and tumour variations.

Test Scope

The scope of the testing covered the following areas:

- Tumour detection accuracy and stage classification using various brain imaging modalities.
- Handling of different tumour types, including benign and malignant.
- Robustness of the system against noisy and low-quality brain scans.
- Performance under normal and high load conditions.

Test Environment

The testing was conducted in the following environment:

- Operating System: Windows 10
- Deep Brain Version: 1.0
- Imaging Modalities: MRI and CT scans
- Dataset: Diverse set of brain scans with known tumour cases and stage annotations

Test Conclusion

Based on the conducted tests, Deep Brain demonstrated accurate tumour detection and stage classification capabilities across various brain scans. The system exhibited robustness in handling different tumour types, irregular shapes, and low-quality images. Additionally, it efficiently processed multiple requests and provided results within an acceptable response time.

Deep Brain has proven to be reliable and effective in its core functionality, providing valuable assistance to medical professionals in brain tumour diagnosis and treatment planning.

Test Result

TC ID	Input	Expected Result	Actual Result	Pass
TBDD001	Brain MRI scan with confirmed tumour	Accurate tumour detection and malignant stage classification	Accurate tumour detection and malignant stage classification	Yes
TBDD002	Brain CT scan with no tumour	Correct identification of no tumour	Correct identification of no tumour	Yes
TBDD003	Brain MRI scan with benign tumour	Accurate tumour detection and benign stage classification	Accurate tumour detection and benign stage classification	Yes
TBDD004	Brain MRI scan with low image quality and noise	Proper error message regarding poor image quality	Proper error message regarding poor image quality	Yes
TBDD005	Multiple brain scans sent simultaneously	Accurate results provided for each scan	Accurate results provided for each scan	Yes
TBDD006	Brain MRI scan with irregular tumour shape	Accurate tumour detection despite irregular shape and correct stage classification	Accurate tumour detection despite irregular shape and correct stage classification	Yes
TBDD007	Brain scan with no tumour but with image artefacts	Correct identification of no tumour and alert about image artefacts	Correct identification of no tumour and alert about image artefacts	Yes
TBDD008	Brain scan from older imaging machine with different format	Conversion of input to required format and accurate results provided	Conversion of input to required format and accurate results provided	Yes

CONCLUSION

CONCLUSION

The latest developments in medical imaging tools have facilitated health workers. Medical informatics research has the best options make good use of these exponentially growing volumes of data. Early detection options are essential for effective treatment of brain tumors .This project presented a CAD approach for detecting and categorizing BT's radiological images into three kinds (pituitary-tumor, glioma-tumor, and meningioma-tumor). We also classified glioma-tumor into various categories (Grade-two, Grade-three, and Grade-four) utilizing the DCNN approach (i.e., our proposed work). Firstly, pre-trained DensNet201 deep learning model was used, and the features were extracted from various DensNet blocks. Then, these features were concatenated and passed to softmax classifier to classify the brain tumor. Secondly, the features from different Inception modules were extracted from pre-trained Inceptionv3 model and concatenated and then, passed to the softmax for the classification of brain tumors.The proposed method produced 99.51% testing accuracy on testing samples and achieved the highest performance in detection of brain tumor.The outcome of the presented architecture shows high training and validation accuracy with low training and validation loss. Moreover, the testing phase determines the overall portable EM imaging system's capability and potential of CNN architecture in detecting and localizing the brain tumor with high accuracy.

Future Scope

In the future, we are going to increase MRI images in the used dataset to improve the accuracy of the proposed model. Moreover, Applying the proposed approach to other types of medical images such as x-ray, computed tomography (CT), and ultrasound may constitute a principle of future studies.

BIBILOGRAPHY

BIBILOGRAPHY

References

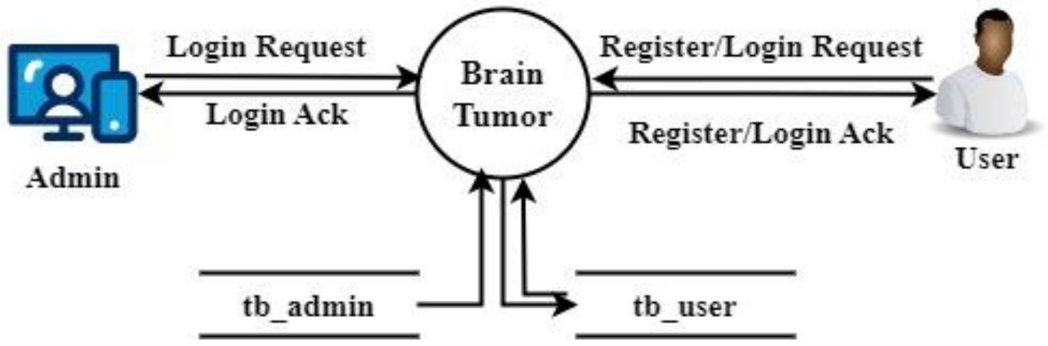
1. M. O. Khairandish, M. Sharma, V. Jain, J. M. Chatterjee, and N. Z. Jhanjhi, ‘‘A hybrid CNN-SVM threshold segmentation approach for tumor detection and classification of MRI brain images,’’ IRBM, Jun. 2021, doi: 10.1016/j.irbm.2021.06.003.
2. Z. A. Al-Saffar and T. Yildirim, ‘‘A hybrid approach based on multiple eigenvalues selection (MES) for the automated grading of a brain tumor using MRI,’’ Comput. Methods Programs Biomed., vol. 201, Apr. 2021, Art. no. 105945, doi: 10.1016/j.cmpb.2021.105945.
3. K. Muhammad, S. Khan, J. D. Ser, and V. H. C. D. Albuquerque, ‘‘Deep learning for multigrade brain tumor classification in smart healthcare systems: A prospective survey,’’ IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 2, pp. 507–522, Feb. 2021.
4. E. Irmak, ‘‘Multi-classification of brain tumor mri images using deep convolutional neural network with fully optimized framework,’’ Iranian J. Sci. Technol., Trans. Electr. Eng., vol. 4, pp. 1–22, Oct. 2021.
5. D. Zhang, G. Huang, Q. Zhang, J. Han, J. Han, and Y. Yu, ‘‘Cross-modality deep feature learning for brain tumor segmentation,’’ Pattern Recognit.,vol. 110, Mar. 2021, Art. no. 107562.
6. P. R. Kshirsagar, A. N. Rakhonde, and P. Chippalkatti, ‘‘MRI image based brain tumor detection using machine learning,’’ Test Eng. Manage., vol. 4, pp. 3672–3680, Jan. 2020.
7. A.Tiwari, S. Srivastava, and M. Pant, ‘‘Brain tumor segmentation and classification from magnetic resonance images: Review of selected methods from 2014 to 2019,’’ Pattern Recognit. Lett., vol. 131, pp. 244–260, Mar. 2020.
8. P. R. Krishnaveni and G. N. Kishore, ‘‘Image based group classifier for brain tumor detection using machine learning technique,’’ Traitement Signal, vol. 37, no. 5, pp. 865–871, Nov. 2020.
9. B. Lei, P. Yang, Y. Zhuo, F. Zhou, D. Ni, S. Chen, X. Xiao, and T. Wang, ‘‘Neuroimaging retrieval via adaptive ensemble manifold learning for brain disease diagnosis,’’ IEEE J. Biomed. Health Informat., vol. 23, no. 4, pp. 1661–1673, Jul. 2019.
10. B. Lei, P. Yang, Y. Zhuo, F. Zhou, D. Ni, S. Chen, X. Xiao, and T. Wang,‘‘Neuroimaging retrieval via adaptive ensemble manifold learning for brain disease diagnosis,’’ IEEE J. Biomed. Health Informat., vol. 23, no. 4,pp. 1661–1673, Jul. 2019.
11. N. Abiwinanda, M. Hanif, S. T. Hesaputra, A. Handayani, andT. R. Mengko, ‘‘Brain tumor classification using convolutional neural network,’’ in World Congress on Medical Physics and Biomedical Engineering. Bhopal, India: Springer, 2019, pp. 183–189.

APPENDICES

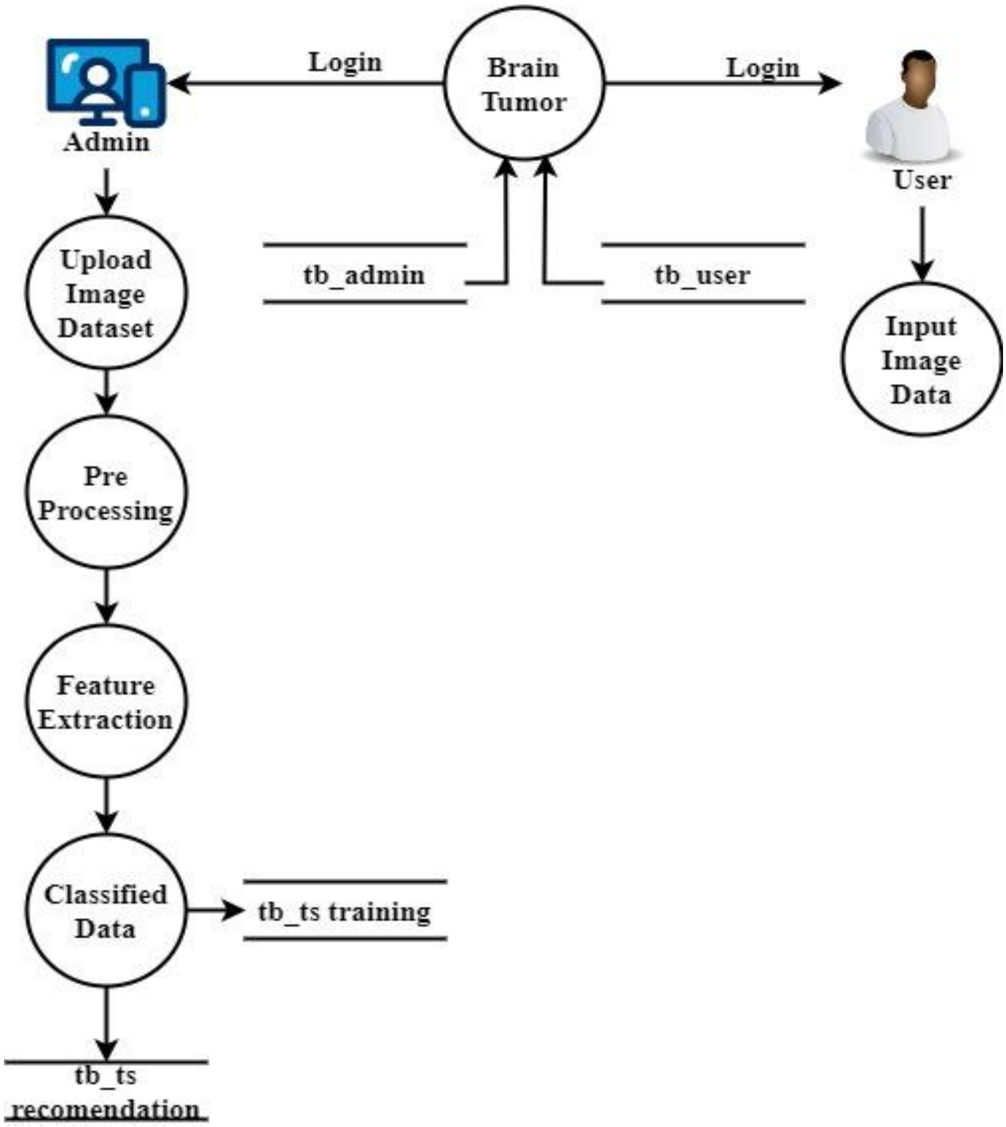
APPENDICES

A. DATA FLOW DIAGRAM

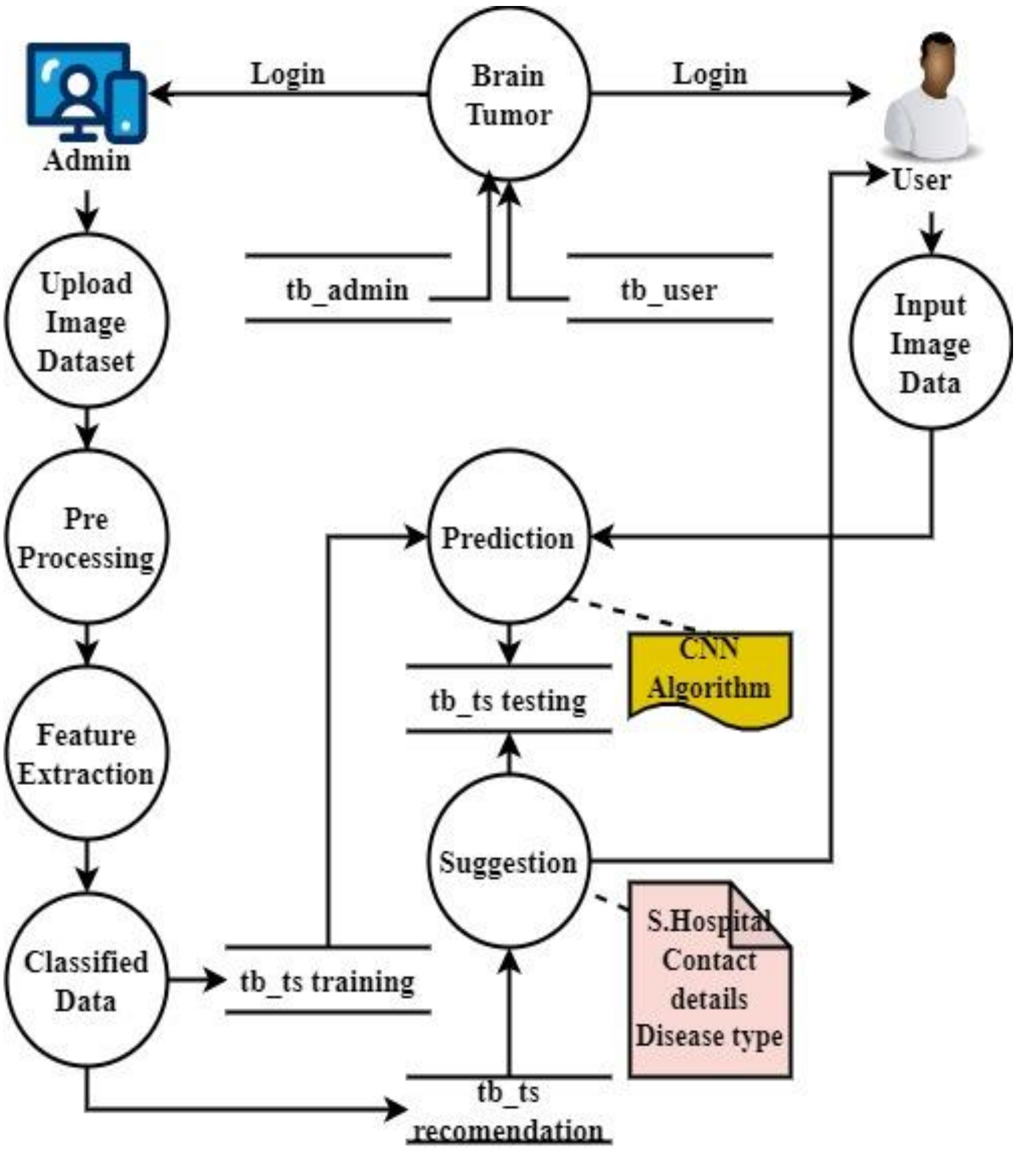
Level -0



Level 1



Level 2



Admin
Login
Brain Tumor
Login
User
Upload Image Dataset
tb_admin
tb_user
Input Image Data
Pre Processing
Prediction
CNN Algorithm
tb_ts testing
Feature Extraction
Suggestion
Classified Data
tb_ts training
S.Hospital Contact details Disease type
tb_ts recomendation

B.SAMPLE CODEING

Packages

```
from flask import Flask, render_template, Response, redirect, request, session, abort, url_for
import mysql.connector
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import cv2
from PIL import Image
from skimage import transform
import seaborn as sns
import keras as k
from keras.layers import Dense
from sklearn.model_selection import train_test_split
Preprocessing
#resize
'''img = cv2.imread('static/data/'+fname)
rez = cv2.resize(img, (256, 256))
cv2.imwrite("static/dataset/"+fname, rez)'''
img = cv2.imread('static/data/'+fname)
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imwrite("static/trained/g_"+fname, gray)
##noice
img = cv2.imread('static/trained/g_'+fname)
dst = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 15)
fname2='ns_'+fname
cv2.imwrite("static/trained/"+fname2, dst)
defkmeans_color_quantization(image, clusters=8, rounds=1):
h, w = image.shape[:2]
samples = np.zeros([h*w,3], dtype=np.float32)
count = 0
```

```
for x in range(h):
for y in range(w):
samples[count] = image[x][y]
count += 1
compactness, labels, centers = cv2.kmeans(samples,
clusters,
None,
(cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10000, 0.0001),
rounds,
cv2.KMEANS_RANDOM_CENTERS)
centers = np.uint8(centers)
res = centers[labels.flatten()]
returnres.reshape((image.shape))
##Binarization
image = cv2.imread('static/data/'+fname)
original = image.copy()
kmeans = kmeans_color_quantization(image, clusters=4)
# Convert to grayscale, Gaussian blur, adaptive threshold
gray = cv2.cvtColor(kmeans, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (3,3), 0)
thresh = cv2.adaptiveThreshold(blur,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
cv2.THRESH_BINARY_INV,21,2)
# Draw largest enclosing circle onto a mask
mask = np.zeros(original.shape[:2], dtype=np.uint8)
cnts = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
cnts = sorted(cnts, key=cv2.contourArea, reverse=True)
for c in cnts:
((x, y), r) = cv2.minEnclosingCircle(c)
cv2.circle(image, (int(x), int(y)), int(r), (36, 255, 12), 2)
cv2.circle(mask, (int(x), int(y)), int(r), 255, -1)
```

```
break
# Bitwise-and for result
result = cv2.bitwise_and(original, original, mask=mask)
result[mask==0] = (0,0,0)'''
cv2.imshow('thresh', thresh)
cv2.imshow('result', result)
cv2.imshow('mask', mask)
cv2.imwrite("static/trained/bb/bin_"+fname, thresh)
#RPN
path_main = 'static/data'
forfname in os.listdir(path_main):
img = cv2.imread('static/data/'+fname)
gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
ret, thresh = cv2.threshold(gray,0,255,cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
kernel = np.ones((3,3),np.uint8)
opening = cv2.morphologyEx(thresh,cv2.MORPH_OPEN,kernel, iterations = 2)
# sure background area
sure_bg = cv2.dilate(opening,kernel,iterations=3)
# Finding sure foreground area
dist_transform = cv2.distanceTransform(opening,cv2.DIST_L2,5)
ret, sure_fg = cv2.threshold(dist_transform,1.5*dist_transform.max(),255,0)
# Finding unknown region
sure_fg = np.uint8(sure_fg)
segment = cv2.subtract(sure_bg,sure_fg)
img = Image.fromarray(img)
segment = Image.fromarray(segment)
path3="static/trained/sg/sg_"+fname
#segment.save(path3)
###Feature extraction & Classification
defDCNN_process(self):
train_data_preprocess = ImageDataGenerator(
rescale = 1./255,
shear_range = 0.2,
```

```
zoom_range = 0.2,
horizontal_flip = True)
test_data_preprocess = (1./255)
train = train_data_preprocess.flow_from_directory(
'dataset/training',
target_size = (128,128),
batch_size = 32,
class_mode = 'binary')
test = train_data_preprocess.flow_from_directory(
'dataset/test',
target_size = (128,128),
batch_size = 32,
class_mode = 'binary')
## Initialize the Convolutional Neural Net
# Initialising the CNN
cnn = Sequential()
# Step 1 - Convolution
# Step 2 - Pooling
cnn.add(Conv2D(32, (3, 3), input_shape = (128, 128, 3), activation = 'relu'))
cnn.add(MaxPooling2D(pool_size = (2, 2)))
# Adding a second convolutional layer
cnn.add(Conv2D(32, (3, 3), activation = 'relu'))
cnn.add(MaxPooling2D(pool_size = (2, 2)))
# Step 3 - Flattening
cnn.add(Flatten())
# Step 4 - Full connection
cnn.add(Dense(units = 128, activation = 'relu'))
cnn.add(Dense(units = 1, activation = 'sigmoid'))
# Compiling the CNN
cnn.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])
history = cnn.fit_generator(train,
steps_per_epoch = 250,
epochs = 25,
```

```
validation_data = test,
validation_steps = 2000)
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('Model Accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
test_image = image.load_img('\\dataset\\', target_size=(128,128))
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis=0)
result = cnn.predict(test_image)
print(result)
if result[0][0] == 1:
print('feature extracted')
else:
print('none')
```

User Registration

```
def register():
msg=""
ifrequest.method=='POST':
name=request.form['name']
address=request.form['address']
dob=request.form['dob']
mobile=request.form['mobile']
```

```
gender=request.form['gender']
email=request.form['email']
uname=request.form['uname']
pass1=request.form['pass']
rdate=date.today()
print(rdate)
mycursor = mydb.cursor()
mycursor.execute("SELECT max(id)+1 FROM register")
maxid = mycursor.fetchone()[0]
ifmaxid is None:
maxid=1
now = datetime.datetime.now()
rdate=now.strftime("%d-%m-%Y")
cursor = mydb.cursor()
sql = "INSERT INTO register(id,name,gender,address,dob,mobile,email,uname,pass,rdate) VALUES
(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"
val = (maxid,name,gender,address,dob,mobile,email,uname,pass1,rdate)
cursor.execute(sql, val)
mydb.commit()
print(cursor.rowcount, "Registered Success")
result="sucess"
ifcursor.rowcount==1:
return redirect(url_for('login',act='1'))
else:
return redirect(url_for('login',act='2'))
```

Add Recommendation

```
defadd_reco():
msg=""
now = datetime.datetime.now()
rdate=now.strftime("%d-%m-%Y")
mycursor = mydb.cursor()
#if request.method=='GET':
# msg = request.args.get('msg')
```

ifrequest.method=='POST':

btype=request.form['btype']

details=request.form['details']

hospital=request.form['hospital']

mycursor.execute("SELECT max(id)+1 FROM recommend")

maxid = mycursor.fetchone()[0]

ifmaxid is None:

maxid=1

sql = "INSERT INTO recommend(id,btype,details,hospital) VALUES (%s, %s, %s, %s)"

val = (maxid,btype,details,hospital)

mycursor.execute(sql,val)

mydb.commit()

return redirect(url_for('add_reco'))

mycursor.execute('SELECT * FROM recommend')

data = mycursor.fetchall()

Test Result Prediction

def test():

msg=""

ss=""

fn=""

fn1=""

fr2=""

predict=""

ff=open("static/trained/class.txt",'r')

ext=ff.read()

ff.close()

cname=ext.split(',')

ifrequest.method=='POST':

file = request.files['file']

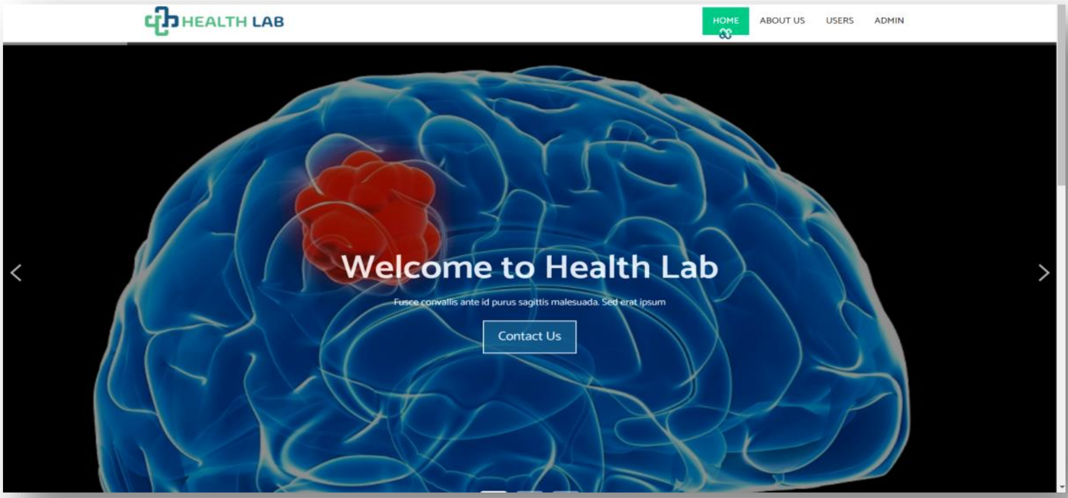
iffile.filename == '':

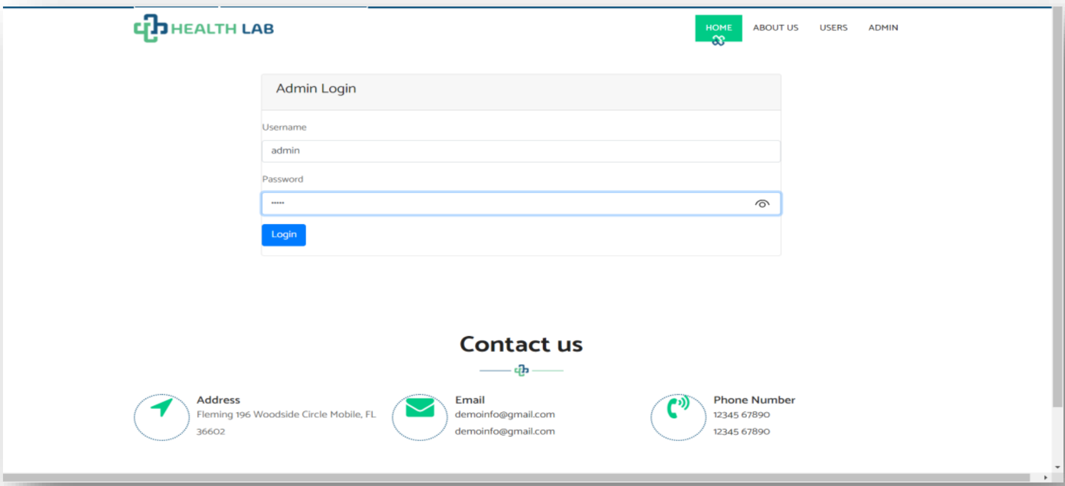
flash('No selected file')

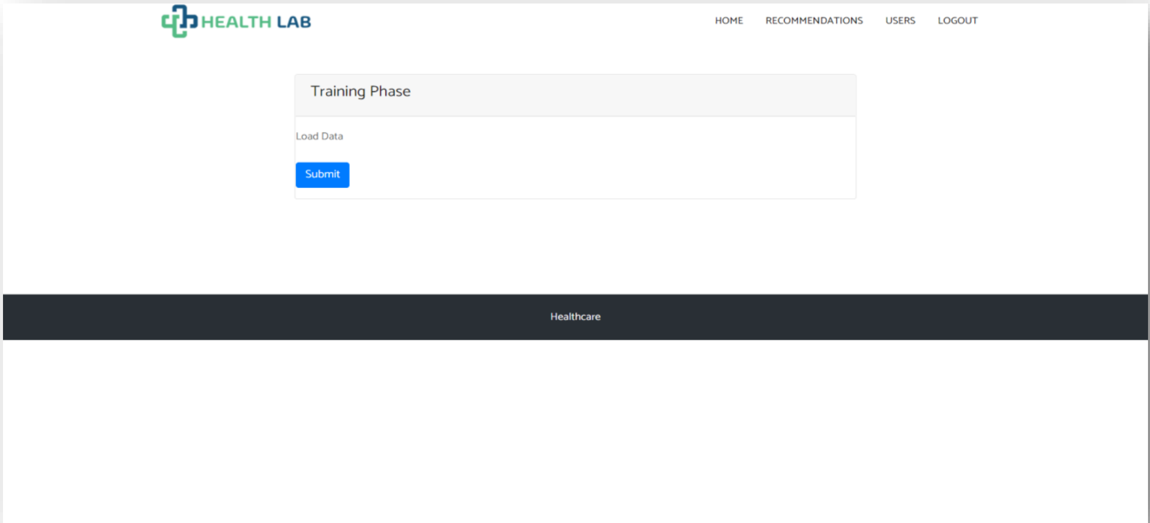
return redirect(request.url)

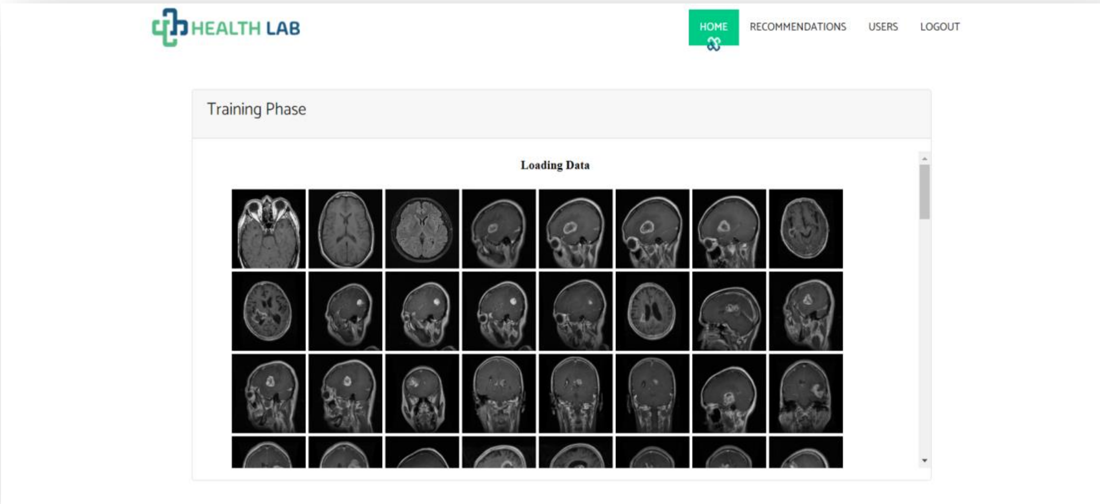
```
if file:
fname = file.filename
filename = secure_filename(fname)
f1=open('static/test/file.txt','w')
f1.write(filename)
f1.close()
file.save(os.path.join("static/test", filename))
cutoff=1
path_main = 'static/dataset'
for fname1 in os.listdir(path_main):
hash0 = imagehash.average_hash(Image.open("static/dataset/"+fname1))
hash1 = imagehash.average_hash(Image.open("static/test/"+filename))
cc1=hash0 - hash1
print("cc="+str(cc1))
if cc1<=cutoff:
ss="ok"
fn=fname1
fr=fn.split('.')
fr2=fr[0]
break
else:
ss="no"
class_name=cname[t]
```

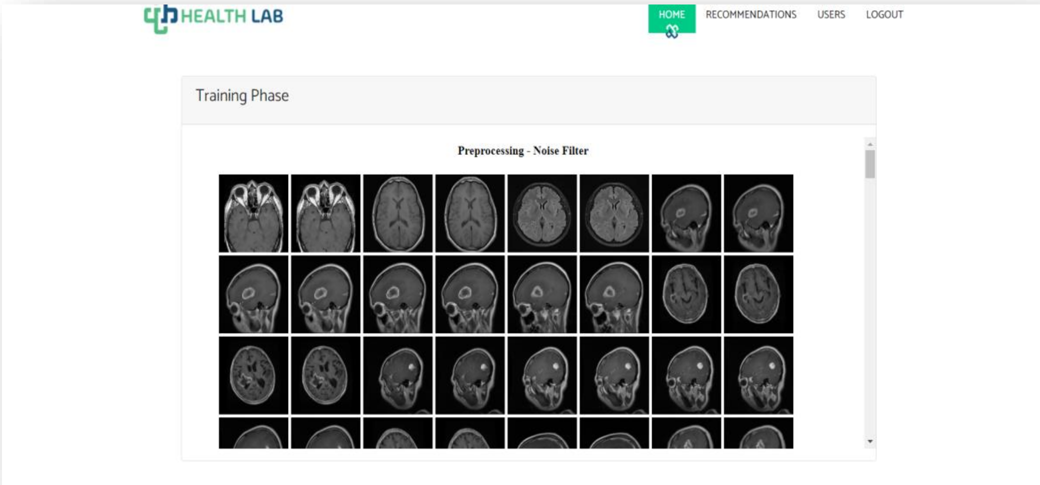
C. SAMPLE INPUT

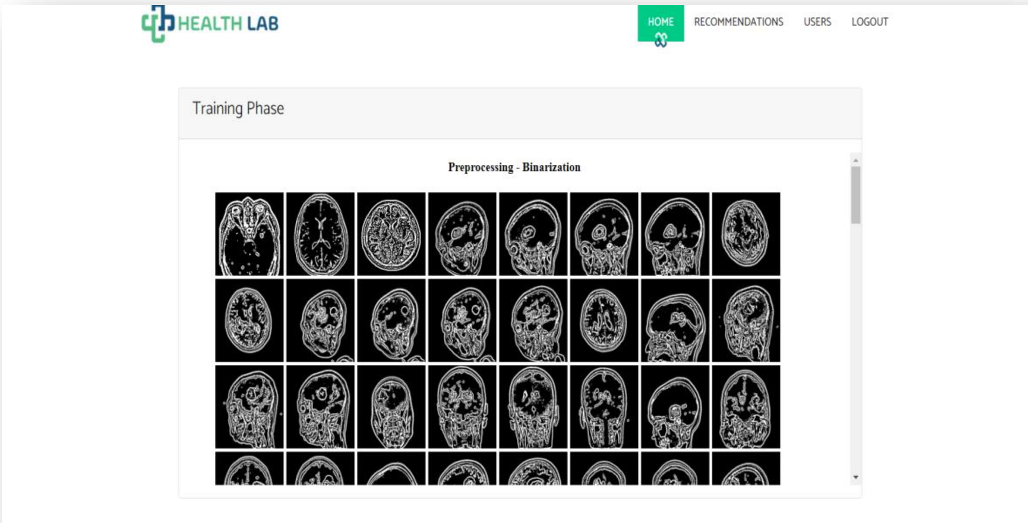


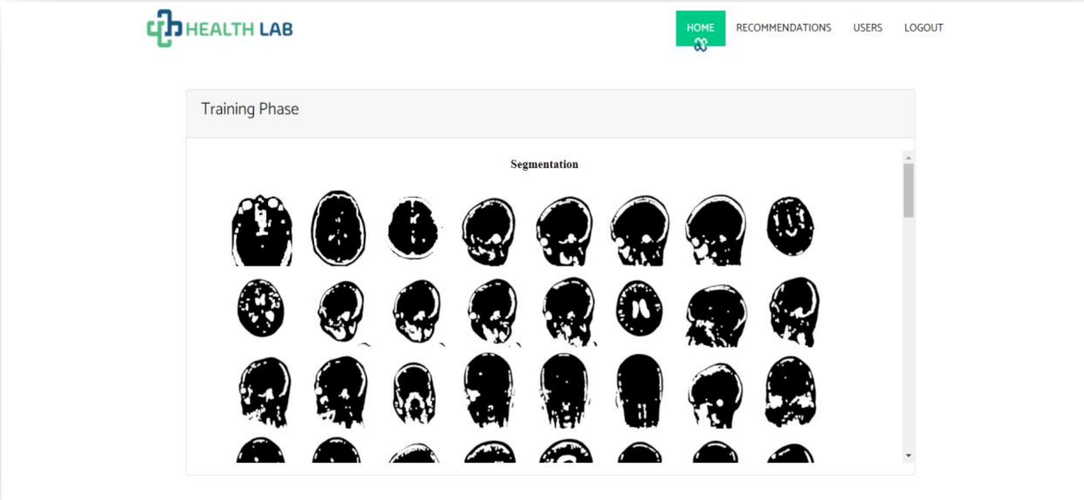


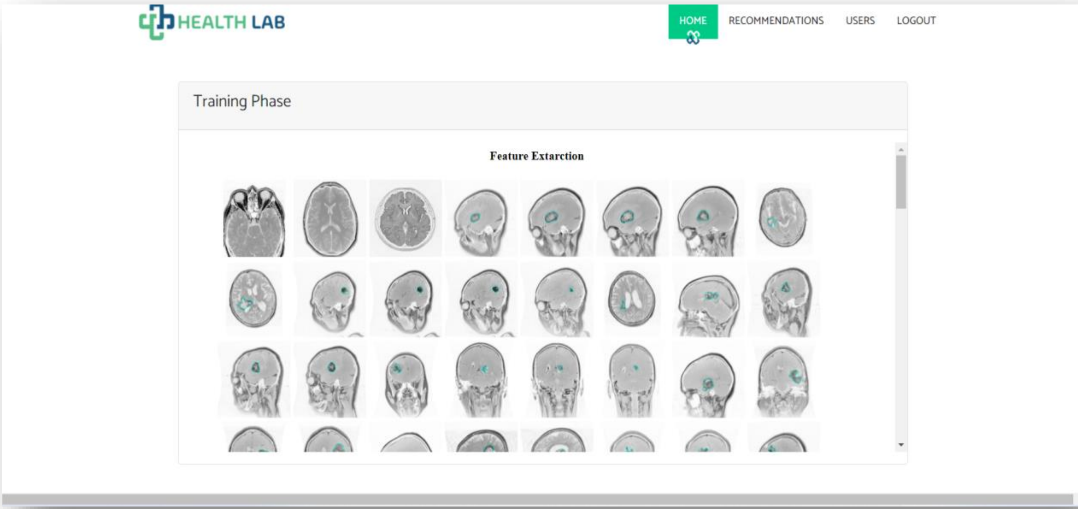

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Training Phase
Load Data
Submit
Healthcare

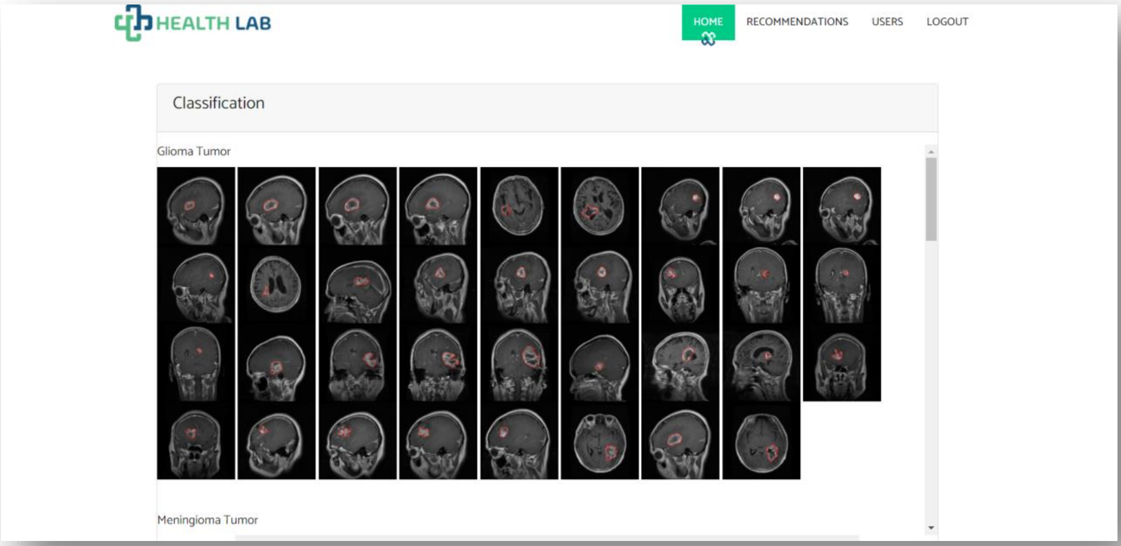

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Training Phase
Loading Data

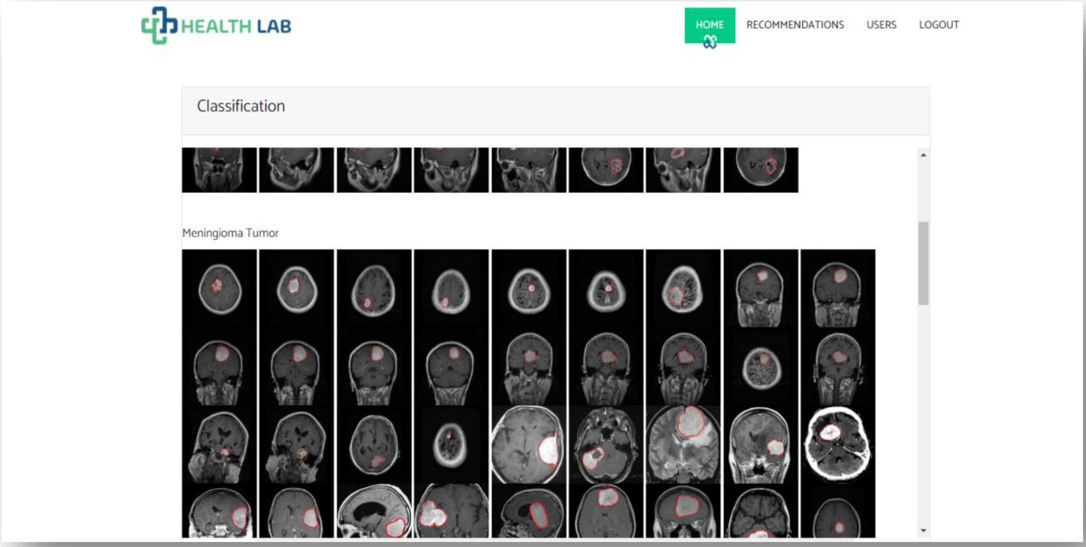

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Training Phase
Preprocessing - Noise Filter

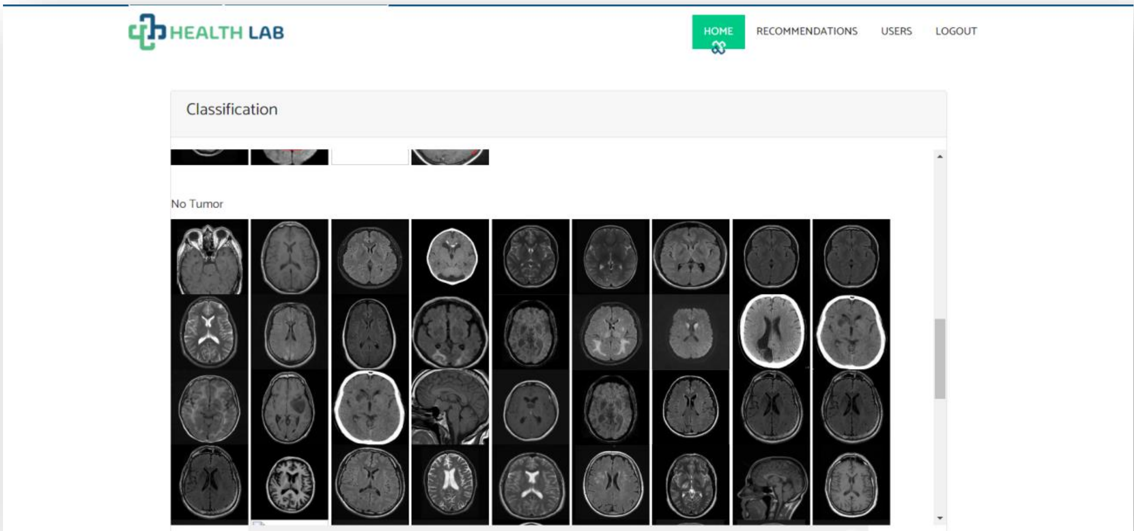

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Training Phase
Preprocessing - Binarization


HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Training Phase
Segmentation

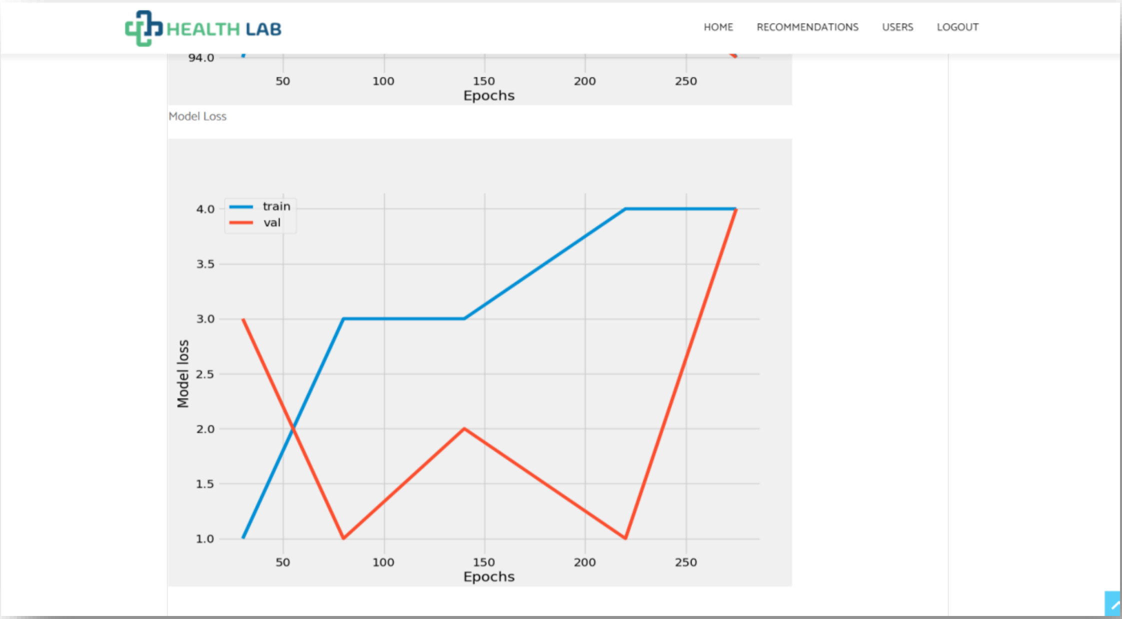

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Training Phase
Feature Extarction

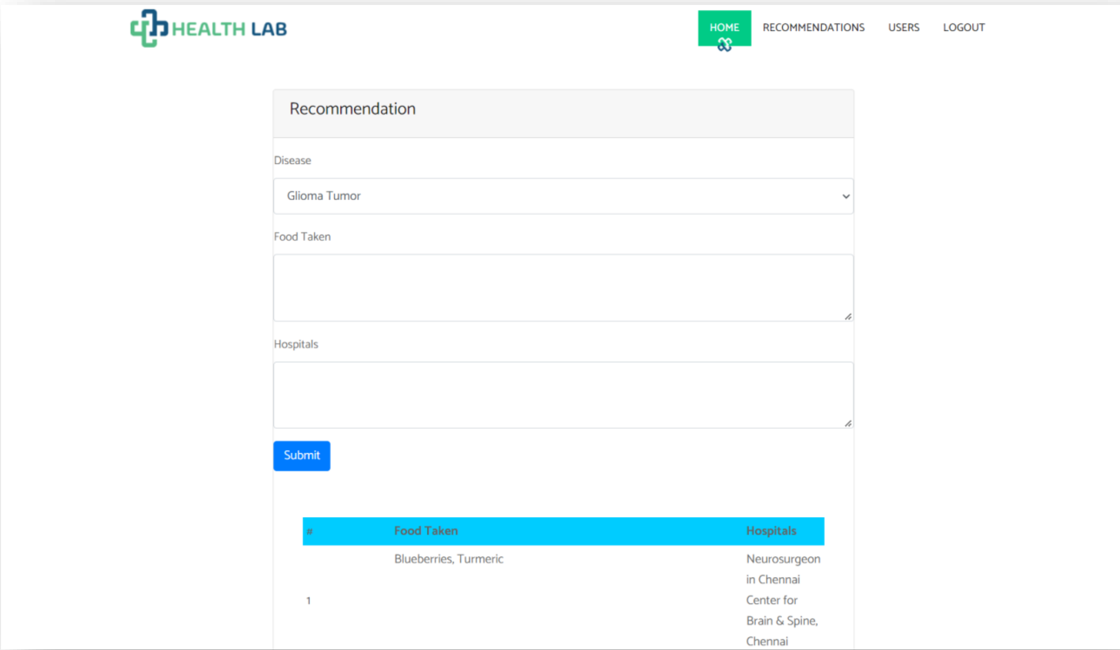

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Classification
Glioma Tumor
Meningioma Tumor


HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Classification
Meningioma Tumor

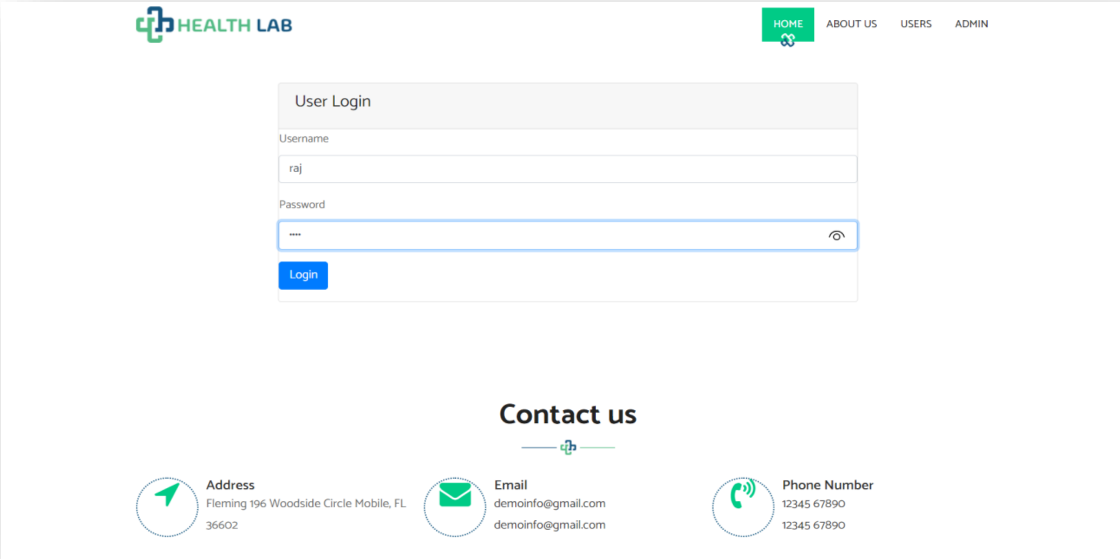

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Classification
No Tumor

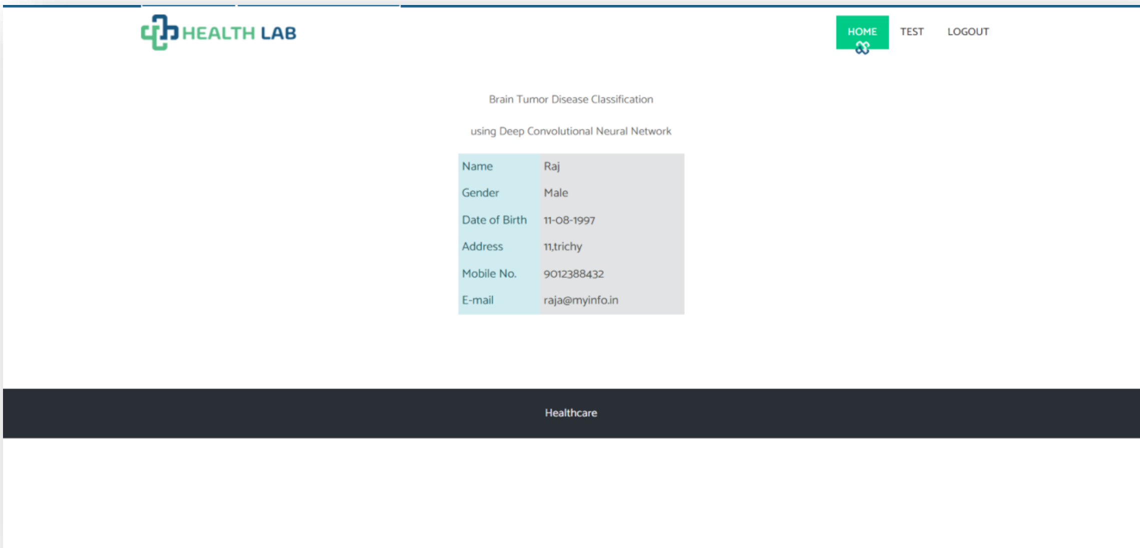

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Performance Analysis
Model Accuracy
train
val
98.0
97.5
97.0
96.5
96.0
95.5
95.0
94.5
94.0
Accuracy %
50
100
150
200
250
Epochs
Model Loss

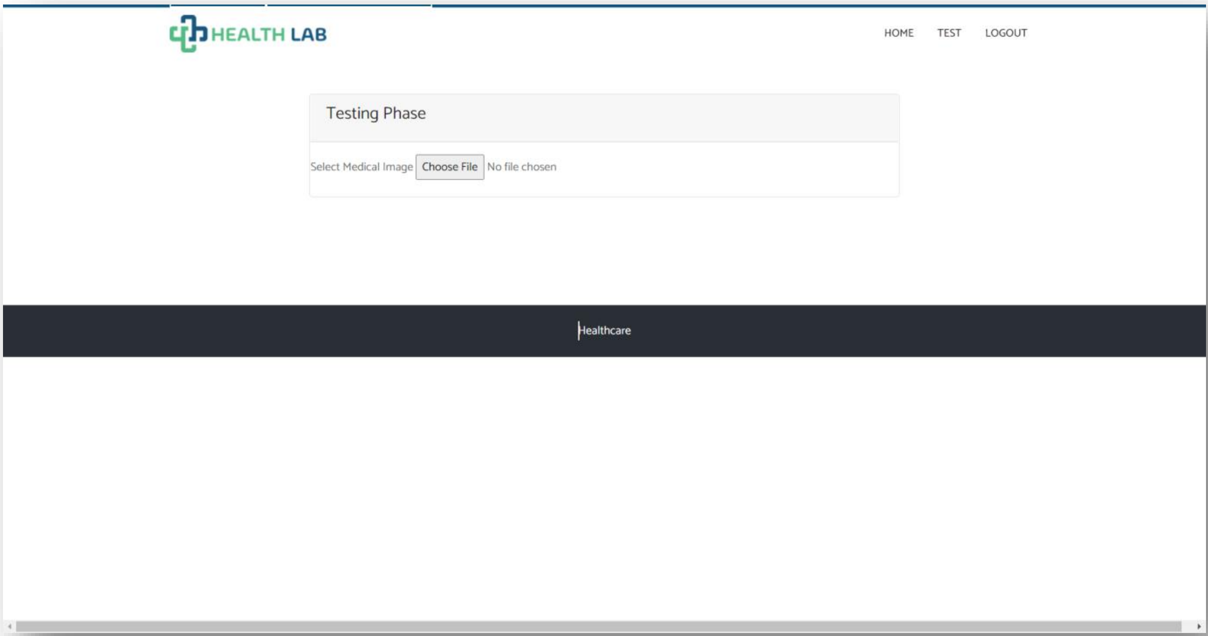

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
94.0
50
100
150
200
250
Epochs
Model Loss
train
val
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Model loss
50
100
150
200
250
Epochs

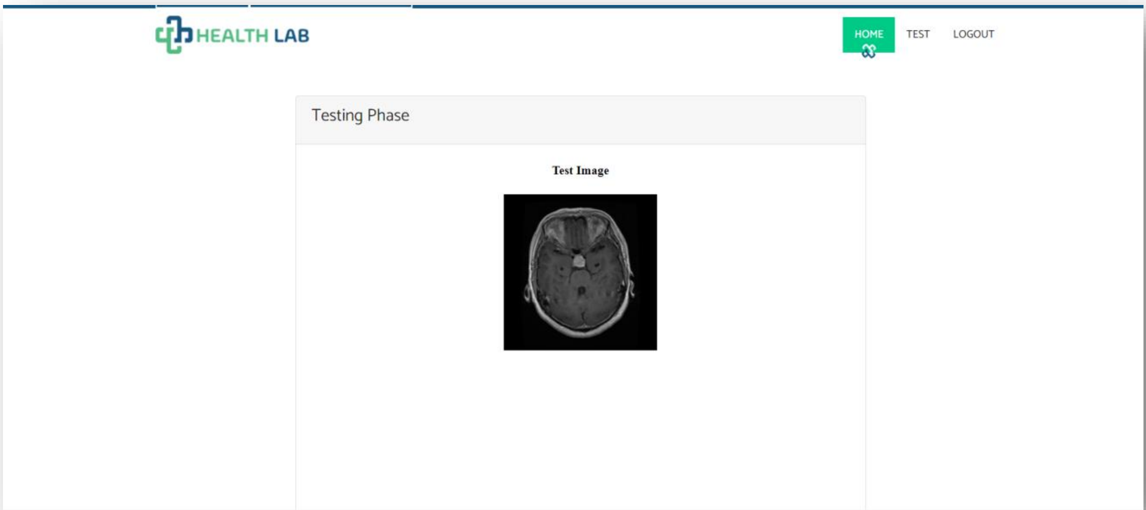

HEALTH LAB
HOME
RECOMMENDATIONS
USERS
LOGOUT
Recommendation
Disease
Glioma Tumor
Food Taken
Hospitals
Submit
#
Food Taken
Hospitals
Blueberries, Turmeric
1
Neurosurgeon in Chennai Center for Brain & Spine, Chennai

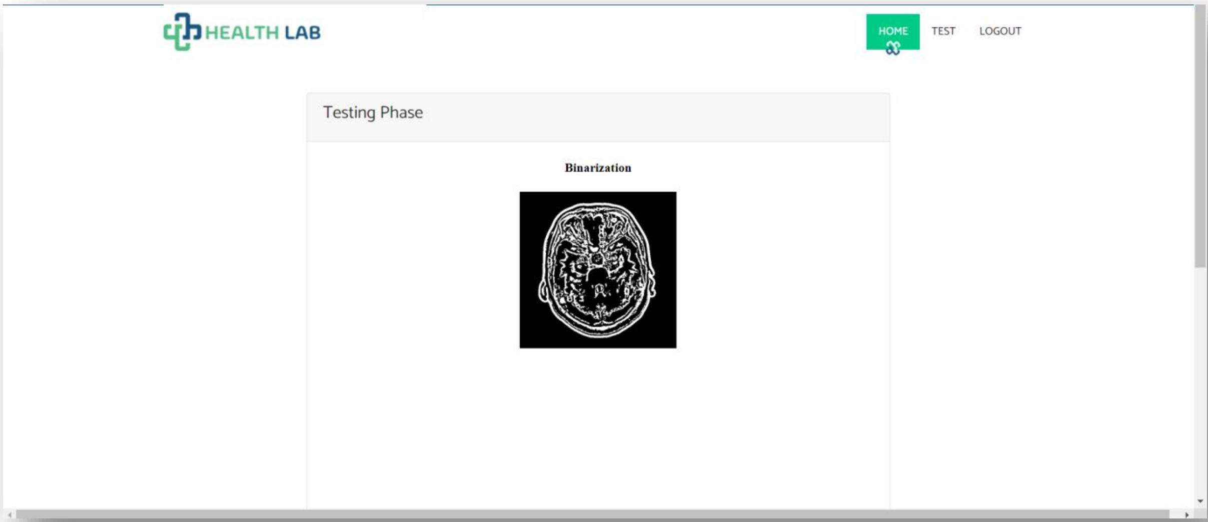
D. SAMPLE OUPTUT

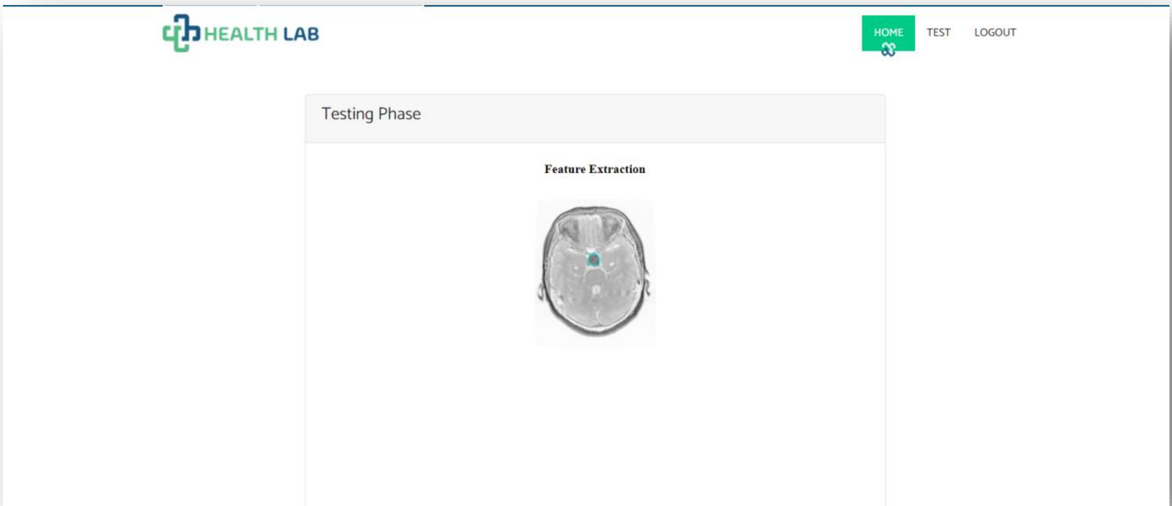


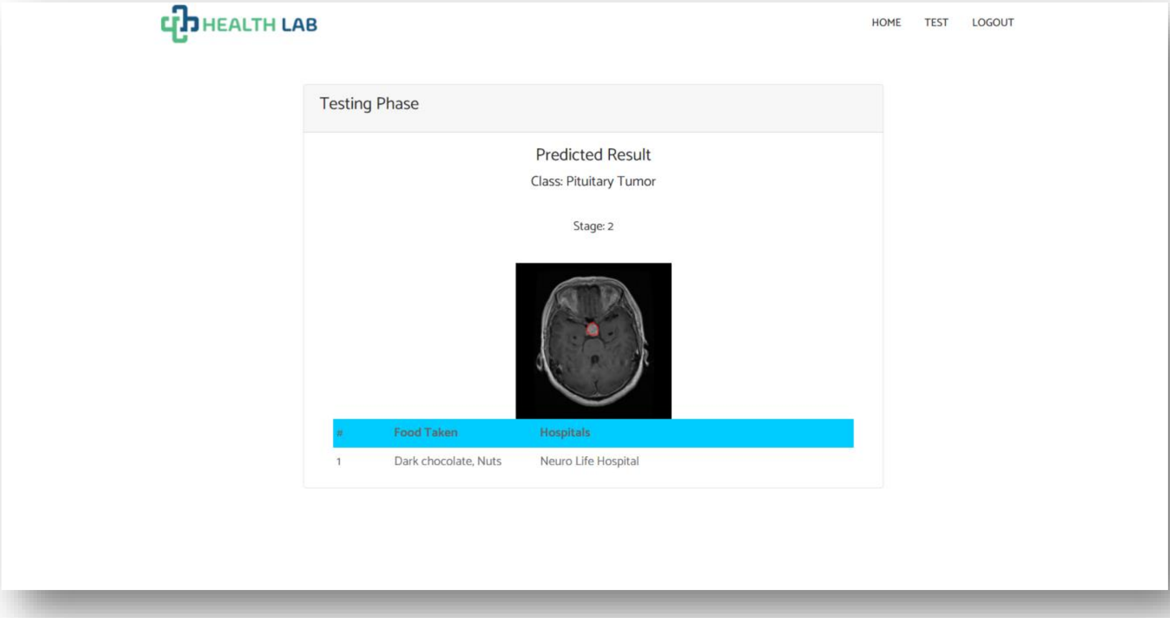


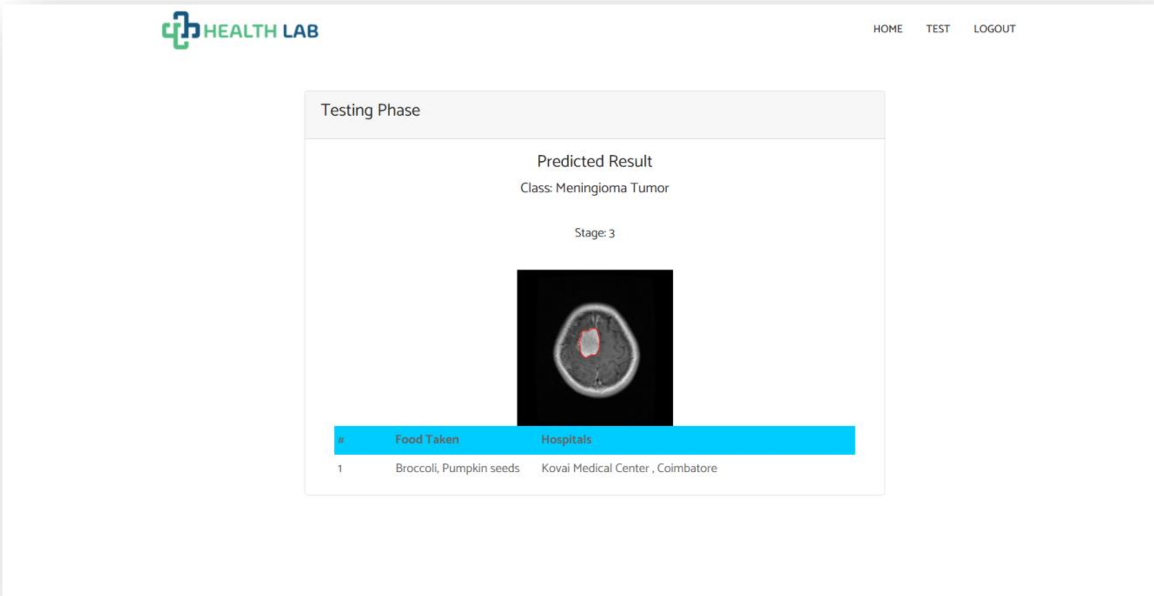

HEALTH LAB
HOME
TEST
LOGOUT
Testing Phase
Select Medical Image
Choose File
No file chosen
Healthcare


HEALTH LAB
HOME
TEST
LOGOUT
Testing Phase
Test Image


HEALTH LAB
HOME
TEST
LOGOUT
Testing Phase
Binarization


HEALTH LAB
HOME
TEST
LOGOUT
Testing Phase
Feature Extraction


HEALTH LAB
HOME
TEST
LOGOUT
Testing Phase
Predicted Result
Class: Pituitary Tumor
Stage: 2
#
Food Taken
Hospitals
1
Dark chocolate, Nuts
Neuro Life Hospital


HEALTH LAB
HOME
TEST
LOGOUT
Testing Phase
Predicted Result
Class: Meningioma Tumor
Stage: 3
#
Food Taken
Hospitals
1
Broccoli, Pumpkin seeds
Kovai Medical Center , Coimbatore