Project Documentation

1. INTRODUCTION

1.1 Project Overview

The project aims to leverage deep learning models, specifically Xception, to analyze medical imaging data for the early detection of Alzheimer's disease. Alzheimer's is a progressive neurological disorder leading to memory loss, cognitive impairment, and behavioral changes.

1.2 Purpose

The purpose of this project is to develop a system that can identify early signs of Alzheimer's disease through the analysis of medical imaging data. Early detection allows for timely intervention and support, improving outcomes for patients and their families.

2. LITERATURE SURVEY

2.1 Existing problem

Alzheimer's disease poses a significant healthcare challenge, with late-stage diagnosis limiting treatment options. Existing literature highlights the potential of deep learning models in early detection using medical imaging.

2.2 References

- Cohen, David & Carpenter, Kristy & Jarrell, Juliet & Huang, Xudong. (2019). Deep learning-based classification of multicategorical presenile dementia data. Current Neurobiology. 10. 141-147.
- N. D. Kodikara, R. N. Rajapakse and K. A. N. N. P. Gunawardena "Applying CNN for pre-detection of Alzheimer's disease from structural MRI data" (2017) 24th International Conference on Mechatronics and Machine Vision in Practice (M2VIP), 2017, pp. 1-7, doi:10.1109/M2VIP.2017.8211486.
- Tong L, Venugopalan J, Wang MD, Hassanzadeh HR. Multimodal deep learning models for early detectionof Alzheimers stage. doi 10.1038/s41598-020-74399-w Sci Rep. 2021Feb5;11(1):3254.

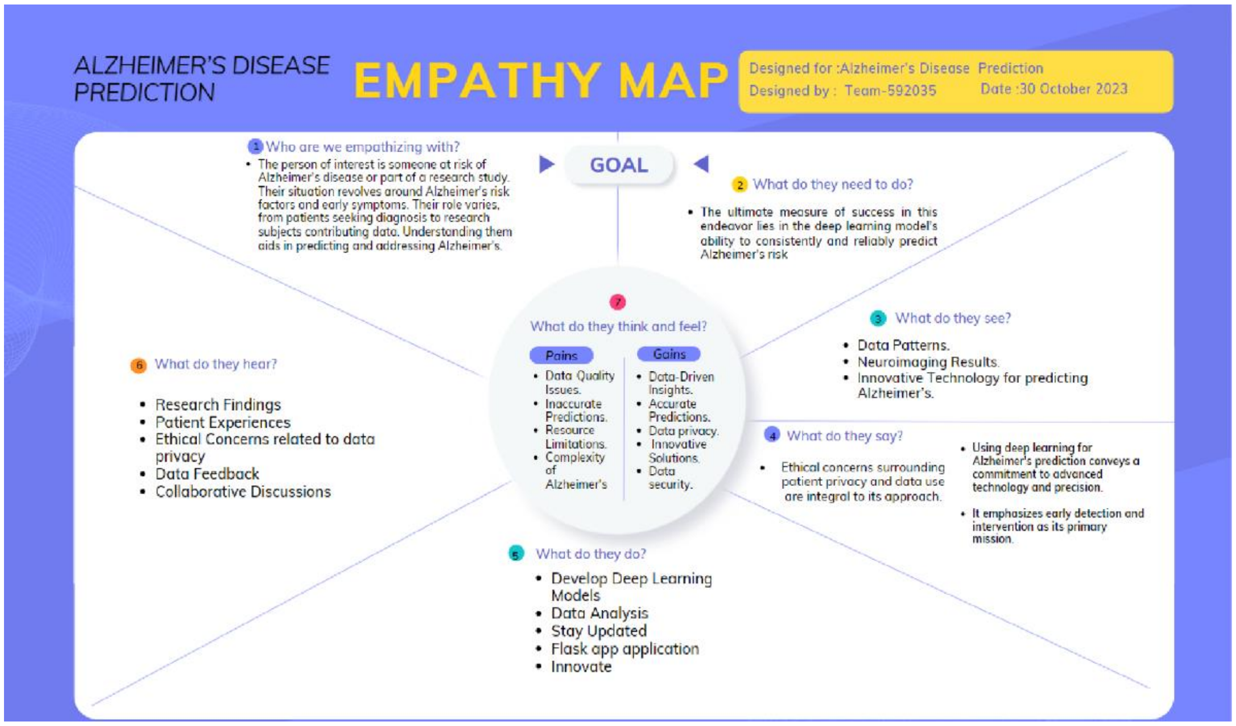
2.3 Problem Statement Definition

The problem statement involves the need for an efficient and accurate tool to identify early signs of Alzheimer's disease through the analysis of medical imaging data using deep learning techniques.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is mainly used fo understanding the needs and challenges faced by individuals affected by Alzheimer's disease to inform the development of a user-centered solution.


ALZHEIMER'S DISEASE PREDICTION

EMPATHY MAP

Designed for :Alzheimer's Disease Prediction
Designed by : Team-592035
Date :30 October 2023

GOAL

1 Who are we empathizing with?

- The person of interest is someone at risk of Alzheimer's disease or part of a research study. Their situation revolves around Alzheimer's risk factors and early symptoms. Their role varies, from patients seeking diagnosis to research subjects contributing data. Understanding them aids in predicting and addressing Alzheimer's.

2 What do they need to do?

- The ultimate measure of success in this endeavor lies in the deep learning model's ability to consistently and reliably predict Alzheimer's risk

3 What do they see?

- Data Patterns.
- Neuroimaging Results.
- Innovative Technology for predicting Alzheimer's.

4 What do they say?

- Ethical concerns surrounding patient privacy and data use are integral to its approach.
- Using deep learning for Alzheimer's prediction conveys a commitment to advanced technology and precision.
- It emphasizes early detection and intervention as its primary mission.

5 What do they do?

- Develop Deep Learning Models
- Data Analysis
- Stay Updated
- Flask app application
- Innovate

6 What do they hear?

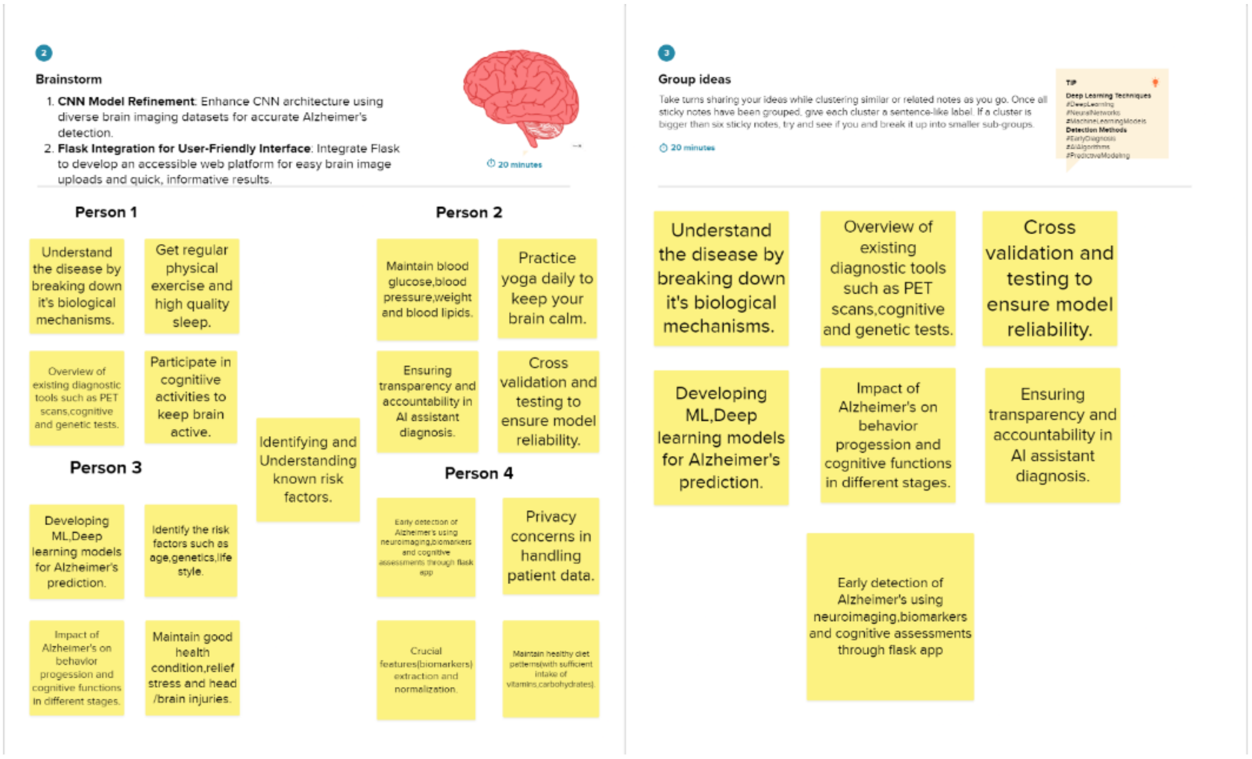
- Research Findings
- Patient Experiences
- Ethical Concerns related to data privacy
- Data Feedback
- Collaborative Discussions

7 What do they think and feel?

Pains	Gains
Data Quality Issues.	Data-Driven Insights.
Inaccurate Predictions.	Accurate Predictions.
Resource Limitations.	Data privacy.
Complexity of Alzheimer's	Innovative Solutions.
	Data security.

3.2 Ideation & Brainstorming

Brainstorming is a method of generating ideas and sharing knowledge to solve a particular commercial or technical problem, in which participants are encouraged to think. It is used for generating innovative ideas for implementing the Xception model for Alzheimer's disease detection, considering both technical and user-centric aspects.


2

Brainstorm

1. **CNN Model Refinement**: Enhance CNN architecture using diverse brain imaging datasets for accurate Alzheimer's detection.
2. **Flask Integration for User-Friendly Interface**: Integrate Flask to develop an accessible web platform for easy brain image uploads and quick, informative results.

20 minutes

Person 1

- Understand the disease by breaking down it's biological mechanisms.
- Get regular physical exercise and high quality sleep.
- Overview of existing diagnostic tools such as PET scans,cognitive and genetic tests.
- Participate in cognitiive activities to keep brain active.

Person 2

- Maintain blood glucose,blood pressure,weight and blood lipids.
- Practice yoga daily to keep your brain calm.
- Ensuring transparency and accountability in AI assistant diagnosis.
- Cross validation and testing to ensure model reliability.

Identifying and Understanding known risk factors.

Person 3

- Developing ML,Deep learning models for Alzheimer's prediction.
- Identify the risk factors such as age,genetics,life style.
- Impact of Alzheimer's on behavior progession and cognitive functions in different stages.
- Maintain good health condition,relief stress and head /brain injuries.

Person 4

- Early detection of Alzheimer's using neuroimaging,biomarkers and cognitive assessments through flask app
- Privacy concerns in handling patient data.
- Crucial features(biomarkers) extraction and normalization.
- Maintain healthy diet patterns(with sufficient intake of vitamins,carbohydrates).

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP
Deep Learning Techniques
#DeepLearning
#NeuralNetworks
#MachineLearningModels
Detection Methods
#EarlyDiagnosis
#AIAlgorithms
#PredictiveModeling

- Understand the disease by breaking down it's biological mechanisms.
- Overview of existing diagnostic tools such as PET scans,cognitive and genetic tests.
- Cross validation and testing to ensure model reliability.
- Developing ML,Deep learning models for Alzheimer's prediction.
- Impact of Alzheimer's on behavior progession and cognitive functions in different stages.
- Ensuring transparency and accountability in AI assistant diagnosis.
- Early detection of Alzheimer's using neuroimaging,biomarkers and cognitive assessments through flask app

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Image Data Input:

- Users interact with the UI to select an image for analysis.
- The system facilitates seamless image input, ensuring user-friendly interaction.

Xception Model Integration:

- The chosen image undergoes processing through a pre-trained Xception deep learning model.
- Integration of the Xception model into a Flask application is essential for efficient analysis.

Result Output for Healthcare Providers:

- The Xception model analyzes the image, generating predictions related to Alzheimer's disease.
- The system displays these predictions on the Flask UI, providing healthcare providers with actionable insights.

4.2 Non-Functional Requirements

System Reliability:

- The solution exhibits high reliability to ensure accurate and consistent results.
- Robust error handling and recovery mechanisms implemented to enhance system reliability.

Accuracy:

- The accuracy of the Xception model in predicting Alzheimer's disease is most important.
- Regular model validation and fine-tuning processes will be employed to maintain a high level of accuracy.

Scalability:

- The system should scale efficiently to handle an increasing volume of image data.
- Scalability considerations will be integrated into the design, allowing the tool to accommodate a growing user base and dataset.

Efficient Processing:

- Timely processing of image data is crucial for user satisfaction.

- Optimization strategies, such as parallel processing, will be employed to enhance the efficiency of the Xception model analysis.

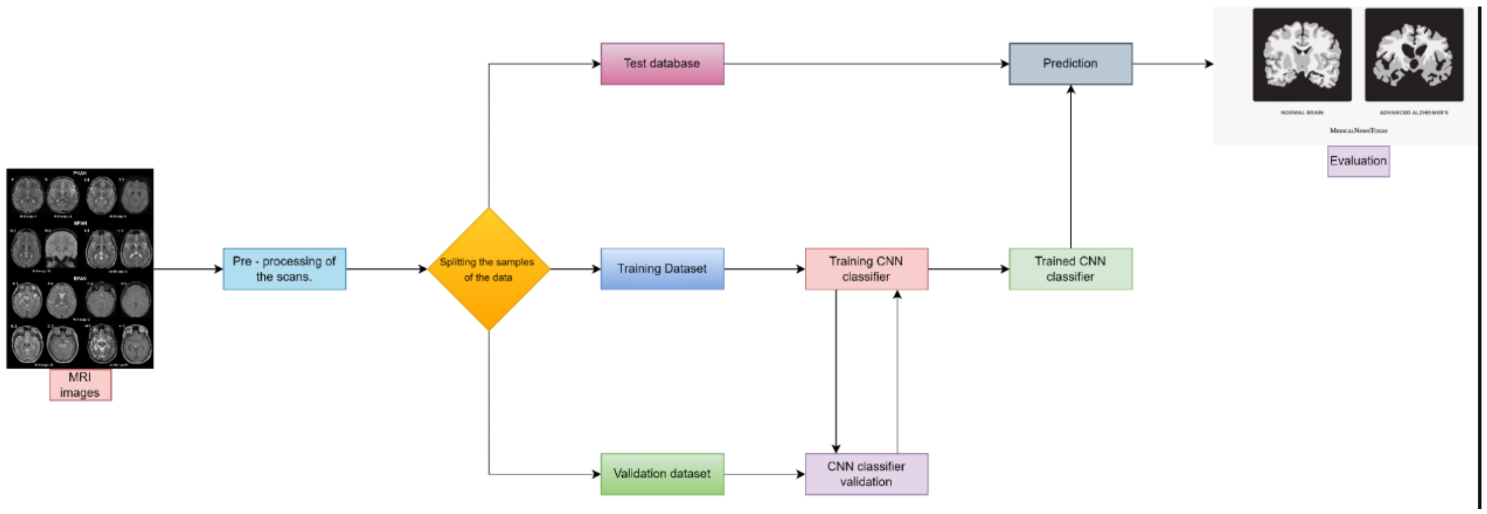
User Interface Responsiveness:

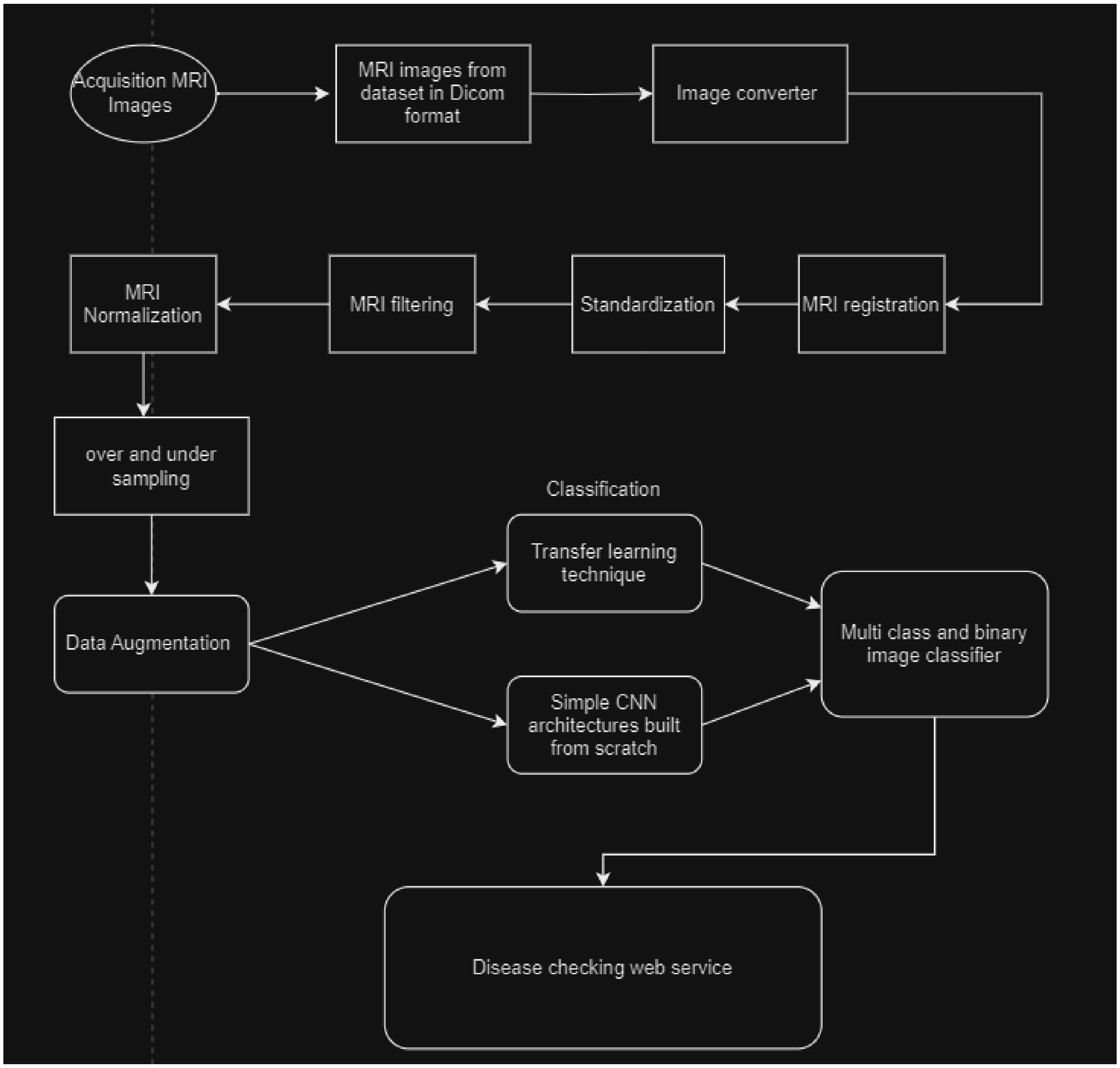
- The Flask UI must be responsive and user-friendly, ensuring a smooth experience for healthcare providers.
- Regular user testing and interface optimizations will be conducted to enhance responsiveness and usability.
-

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories

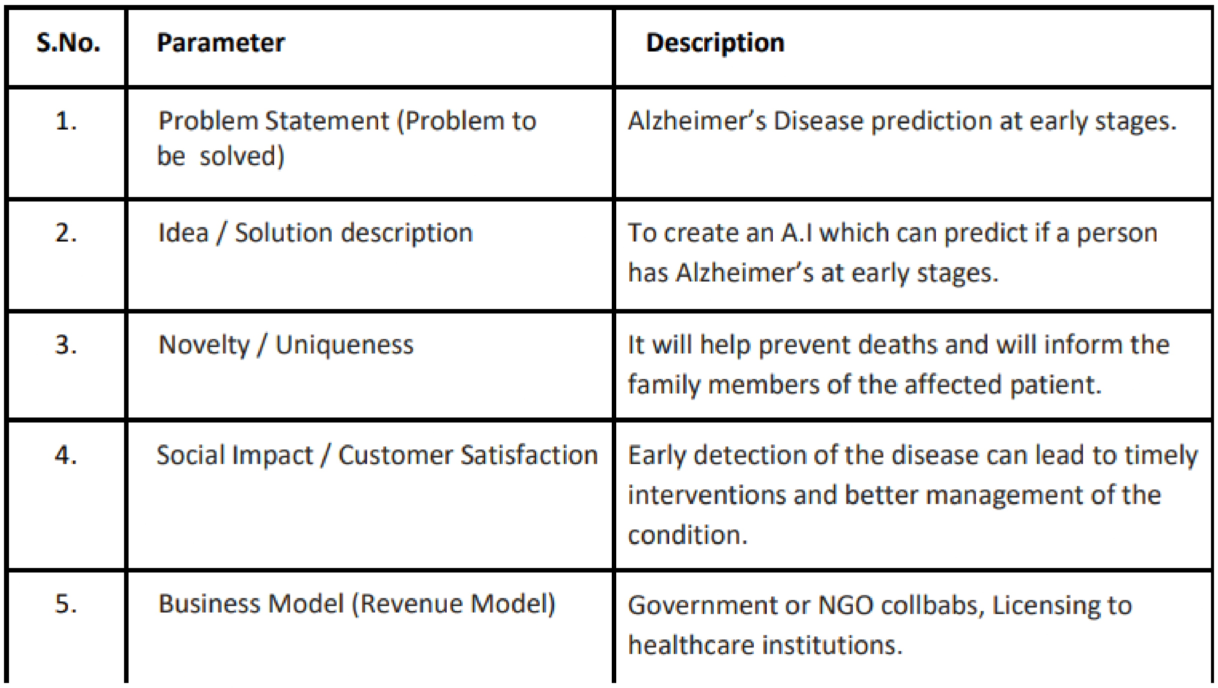
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored. Visualizing the flow of data within the system and outlining user stories to capture the end-to-end user experience and system interactions.




Acquisition MRI Images
MRI images from dataset in Dicom format
Image converter
MRI registration
Standardization
MRI filtering
MRI Normalization
over and under sampling
Data Augmentation
Classification
Transfer learning technique
Simple CNN architectures built from scratch
Multi class and binary image classifier
Disease checking web service

5.2 Solution Architecture

The proposed solution architecture seamlessly integrates the Xception model into a medical imaging analysis framework, allowing users to input images through a user-friendly interface. This robust structure ensures real-time processing and accurate predictions for Alzheimer's disease. The system's characteristics emphasize reliability, accuracy, and scalability to meet the demands of healthcare providers. Key features include image data input, model integration, and result output on the Flask UI. With a phased development approach and specified requirements, this solution is designed to address existing business challenges in Alzheimer's disease detection effectively.

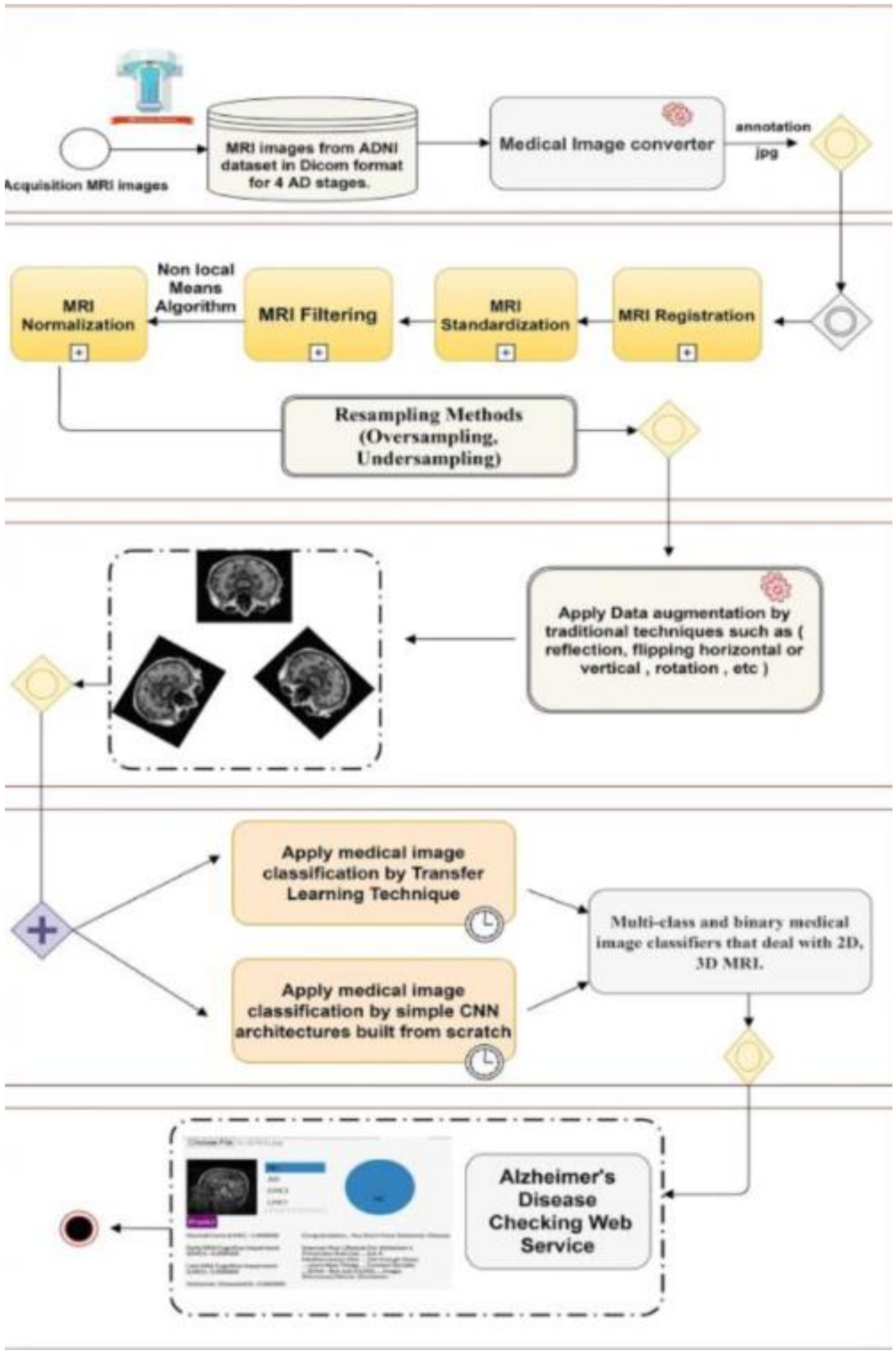


S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Alzheimer's Disease prediction at early stages.
2.	Idea / Solution description	To create an A.I which can predict if a person has Alzheimer's at early stages.
3.	Novelty / Uniqueness	It will help prevent deaths and will inform the family members of the affected patient.
4.	Social Impact / Customer Satisfaction	Early detection of the disease can lead to timely interventions and better management of the condition.
5.	Business Model (Revenue Model)	Government or NGO collbabs, Licensing to healthcare institutions.
6.	Scalability of the Solution	Data availability and quality, Ethical considerations, validations and generalizations.

6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture

The development of a technical blueprint with regard to the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met. It includes the architectural diagram detailing logic, services etc.


Acquisition MRI images
MRI images from ADNI dataset in Dicom format for 4 AD stages.
Medical Image converter
annotation
jpg
MRI Registration
MRI Standardization
MRI Filtering
Non local Means Algorithm
MRI Normalization
Resampling Methods (Oversampling, Undersampling)
Apply Data augmentation by traditional techniques such as (reflection, flipping horizontal or vertical , rotation , etc)
Apply medical image classification by Transfer Learning Technique
Apply medical image classification by simple CNN architectures built from scratch
Multi-class and binary medical image classifiers that deal with 2D, 3D MRI.
Alzheimer's Disease Checking Web Service

6.2 Sprint Planning & Estimation

- Breaking down the development process into sprints, with associated tasks and timelines, to facilitate efficient project management.

- Conduct literature review, implement and train the Xception model for Alzheimer's disease prediction.
- Collaborate with experts to design cognitive tests, to collect and preprocess medical images for training the Xception mode.
- Conduct compatibility testing, submit medical images through the Flask app for Alzheimer's prediction.
- Implement a secure data management system, set up a Flask app with endpoints for model.

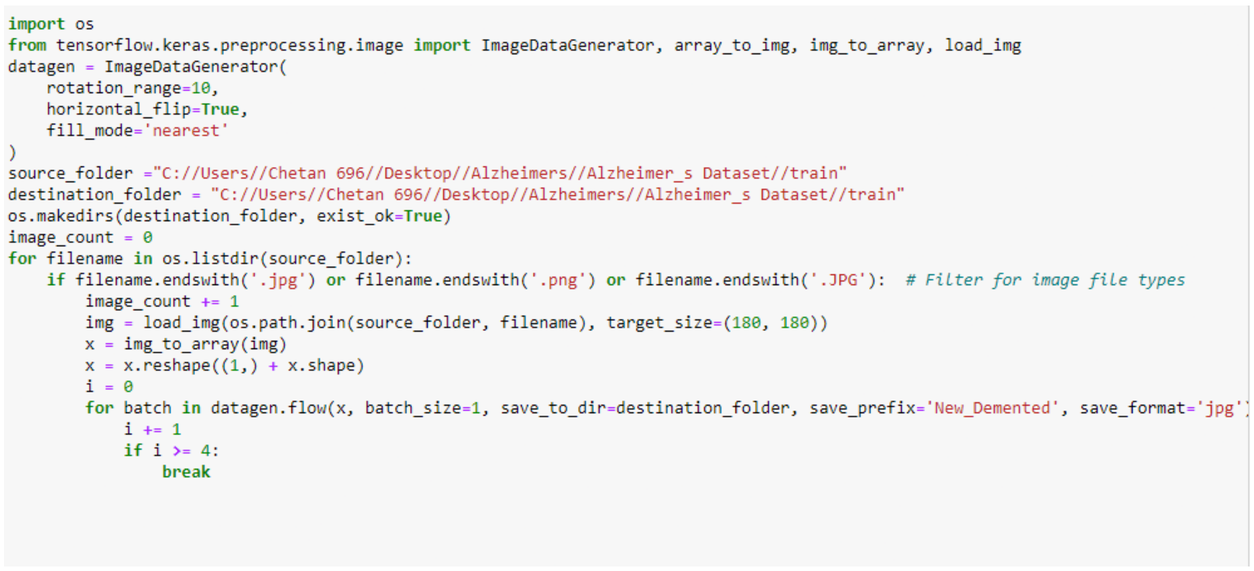
6.3 Sprint Delivery Schedule

1. Sprint 1(Research and Planning) – November 2nd – 5th
2. Sprint 2(Development and Cognitive Tests) – November 10th to 12th
3. Sprint 3(Integration of the Technology) – November 13th and 14th
4. Sprint 4(Model Training) – November 15th to 20th
5. Sprint 5(Data Management and Collaboration) – November 21st to 23rd
6. Sprint 6(Evaluation and Flash Deployment) – November 23rd to 27th

7.Coding and Solution

7.1 Feature 1 - Data Augmentation

Data augmentation is a technique used in machine learning and deep learning to artificially increase the size of a training dataset by applying various transformations to the existing data. The goal of data augmentation is to enhance the model's performance by providing it with a more diverse and robust set of training examples.



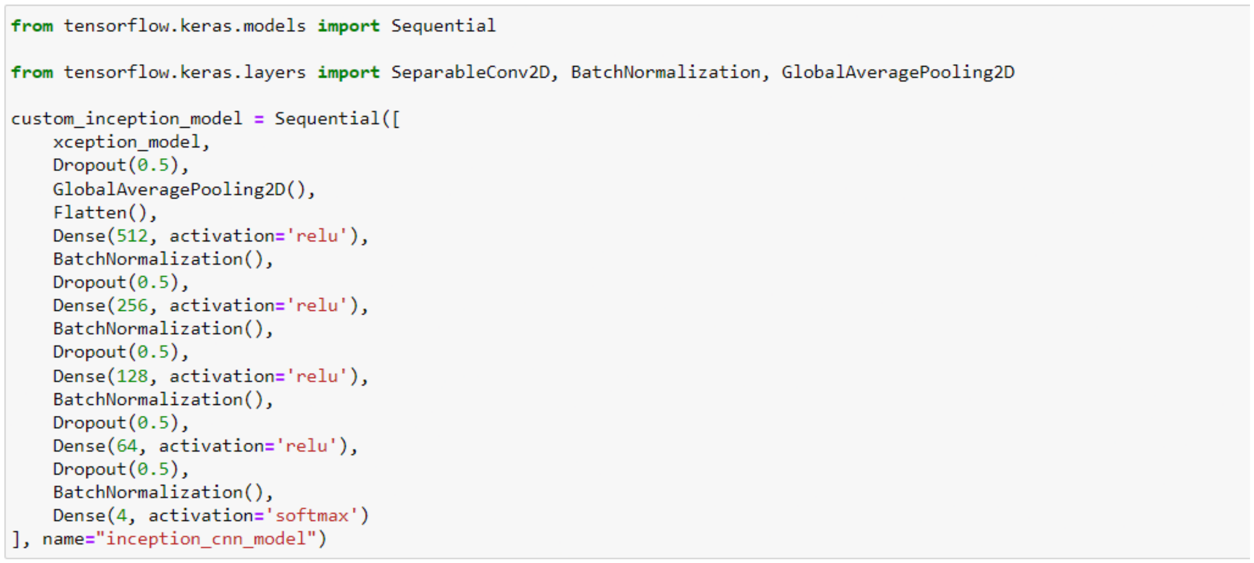
```python
import os
from tensorflow.keras.preprocessing.image import ImageDataGenerator, array_to_img, img_to_array, load_img
datagen = ImageDataGenerator(
    rotation_range=10,
    horizontal_flip=True,
    fill_mode='nearest'
)
source_folder ="C://Users//Chetan 696//Desktop//Alzheimers//Alzheimer_s Dataset//train"
destination_folder = "C://Users//Chetan 696//Desktop//Alzheimers//Alzheimer_s Dataset//train"
os.makedirs(destination_folder, exist_ok=True)
image_count = 0
for filename in os.listdir(source_folder):
    if filename.endswith('.jpg') or filename.endswith('.png') or filename.endswith('.JPG'):  # Filter for image file types
        image_count += 1
        img = load_img(os.path.join(source_folder, filename), target_size=(180, 180))
        x = img_to_array(img)
        x = x.reshape((1,) + x.shape)
        i = 0
        for batch in datagen.flow(x, batch_size=1, save_to_dir=destination_folder, save_prefix='New_Demented', save_format='jpg'
            i += 1
            if i >= 4:
                break
```

1. **Increased Diversity -** By applying transformations such as rotation, flipping, scaling, cropping, and changes in brightness or contrast, the augmented dataset contains variations of the original images. This helps expose the model to a broader range of scenarios and patterns, making it more capable of handling real-world variability**.**
2. **Improved Generalizatio -** Data augmentation helps prevent overfitting, where a model performs well on the training data but struggles with new, unseen data. The augmented dataset introduces variability, making it more likely that the model will learn generalizable features rather than memorizing specific instances from the original data.
3. **Robustness to Variations -** Real-world data is subject to various transformations and distortions. Augmenting the training data with these transformations makes the model more robust and capable of handling noisy or distorted inputs.

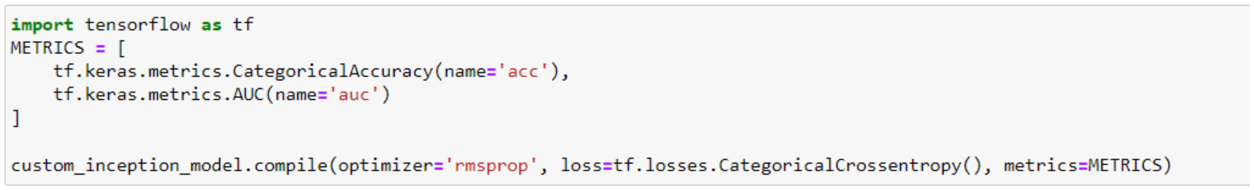
7.2 Feature 2 - Transfer Learning and Architecture

Transfer learning (TL) is a technique in machine learning (ML) in which knowledge learned from a task is re-used in order to boost performance on a related task. It explains the rationale and details of fine-tuning. Evaluation metrics such as accuracy, precision, recall, and model performance on both training and validation datasets are highlighted.

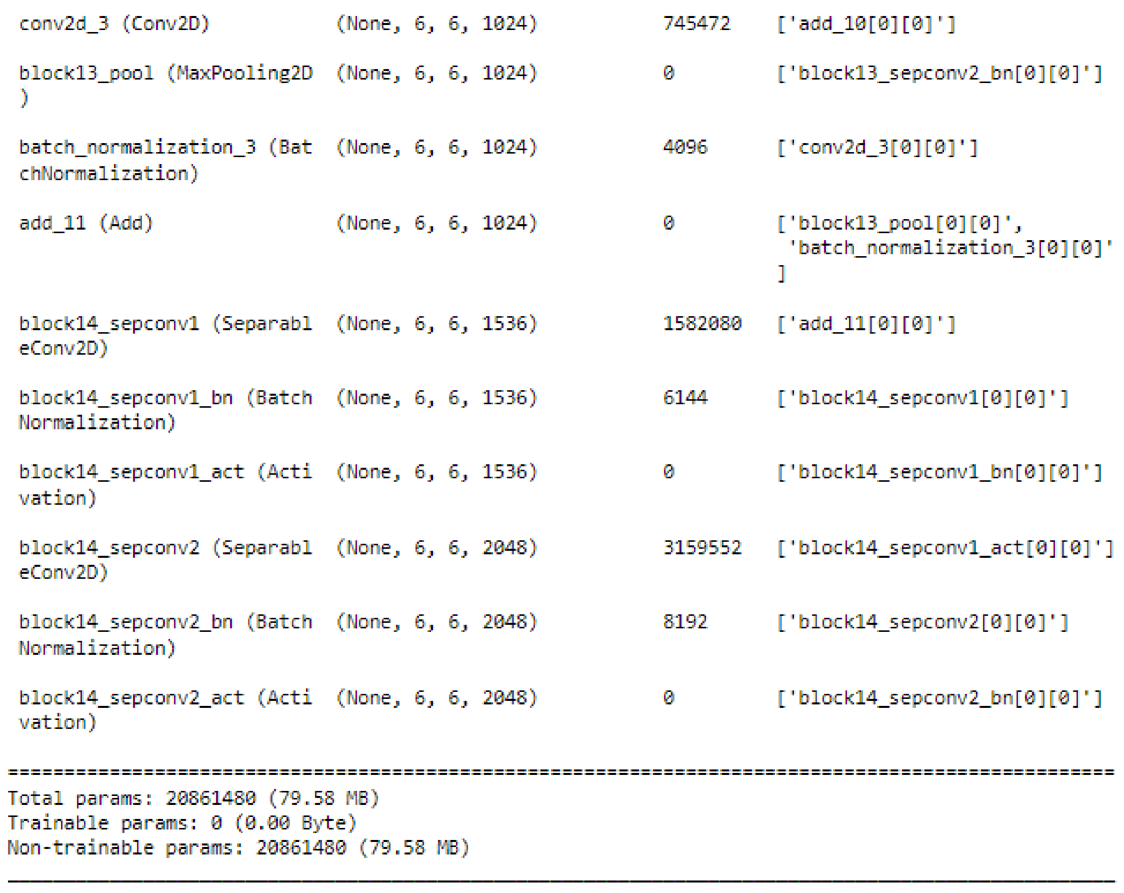
Neural network architecture, including the types and number of layers, data preprocessing steps such as normalization, and specifics of the model training process, encompassing hyperparameter tuning and optimization algorithms.



```
from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import SeparableConv2D, BatchNormalization, GlobalAveragePooling2D

custom_inception_model = Sequential([
    xception_model,
    Dropout(0.5),
    GlobalAveragePooling2D(),
    Flatten(),
    Dense(512, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(256, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(128, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(64, activation='relu'),
    Dropout(0.5),
    BatchNormalization(),
    Dense(4, activation='softmax')
], name="inception_cnn_model")
```



```
import tensorflow as tf
METRICS = [
    tf.keras.metrics.CategoricalAccuracy(name='acc'),
    tf.keras.metrics.AUC(name='auc')
]

custom_inception_model.compile(optimizer='rmsprop', loss=tf.losses.CategoricalCrossentropy(), metrics=METRICS)
```



```
 conv2d_3 (Conv2D)            (None, 6, 6, 1024)          745472    ['add_10[0][0]']

 block13_pool (MaxPooling2D  (None, 6, 6, 1024)          0         ['block13_sepconv2_bn[0][0]']
 )

 batch_normalization_3 (Bat  (None, 6, 6, 1024)          4096      ['conv2d_3[0][0]']
 chNormalization)

 add_11 (Add)                 (None, 6, 6, 1024)          0         ['block13_pool[0][0]',
                                                                     'batch_normalization_3[0][0]'
                                                                    ]

 block14_sepconv1 (Separabl  (None, 6, 6, 1536)          1582080   ['add_11[0][0]']
 eConv2D)

 block14_sepconv1_bn (Batch  (None, 6, 6, 1536)          6144      ['block14_sepconv1[0][0]']
 Normalization)

 block14_sepconv1_act (Acti  (None, 6, 6, 1536)          0         ['block14_sepconv1_bn[0][0]']
 vation)

 block14_sepconv2 (Separabl  (None, 6, 6, 2048)          3159552   ['block14_sepconv1_act[0][0]']
 eConv2D)

 block14_sepconv2_bn (Batch  (None, 6, 6, 2048)          8192      ['block14_sepconv2[0][0]']
 Normalization)

 block14_sepconv2_act (Acti  (None, 6, 6, 2048)          0         ['block14_sepconv2_bn[0][0]']
 vation)

==================================================================================================
Total params: 20861480 (79.58 MB)
Trainable params: 0 (0.00 Byte)
Non-trainable params: 20861480 (79.58 MB)
__________________________________________________________________________________________________
```

8 Performance Testing

8.1 Performance Metrics

Categorical Accuracy (tf.keras.metrics.CategoricalAccuracy) – This metric calculates the accuracy of the model by comparing the true labels to the predicted labels. For a multi-class classification problem (which is indicated by the use of categorical cross entropy as the loss function), categorical accuracy computes the fraction of correctly classified samples over the total number of samples. It is a common metric for classification tasks with more than two classes.

Area Under the Curve (AUC) (tf.keras.metrics.AUC) – AUC is a metric commonly used for binary classification problems, but it can also be applied to multi-class classification by considering each class against the rest (one-vs-all). It measures the area under the Receiver Operating Characteristic (ROC) curve, which plots the true positive rate against the false positive rate.

RMSprop – RMSprop, short for Root Mean Square Propagation, is an optimization algorithm commonly used for training neural networks. It adapts the learning rate independently for each parameter by dividing the learning rate for a weight by a running average of the magnitudes of recent gradients for that weight.

Categorical Crossentropy – Categorical Crossentropy is a common loss function used for multi-class classification problems, where each example belongs to one of several classes. It measures the dissimilarity between the true distribution (ground truth) and the predicted distribution (output of the model).

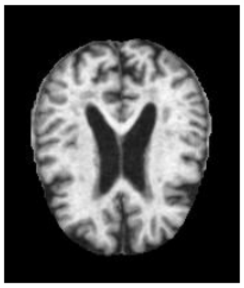
```
import tensorflow as tf
METRICS = [
    tf.keras.metrics.CategoricalAccuracy(name='acc'),
    tf.keras.metrics.AUC(name='auc')
]

custom_inception_model.compile(optimizer='rmsprop', loss=tf.losses.CategoricalCrossentropy(), metrics=METRICS)
```

9. Results

9.1 Output Screenshots

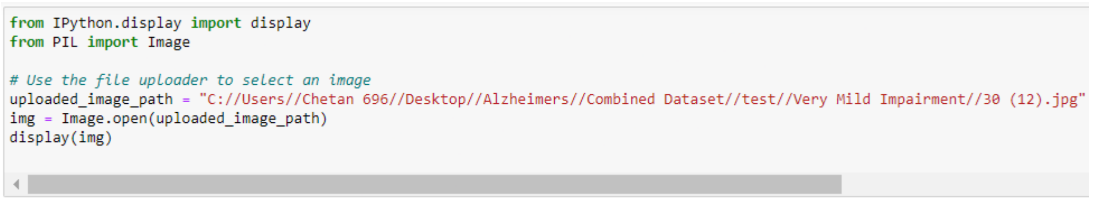
```
history=custom_inception_model.fit(train_data,train_labels,validation_data=(val_data,val_labels), epochs=30)
```

```
Epoch 1/30
130/130 [==============================] - 149s 1s/step - loss: 1.0240 - acc: 0.6002 - auc: 0.8284 - val_loss: 0.5805 - val_ac
c: 0.7923 - val_auc: 0.9429
```

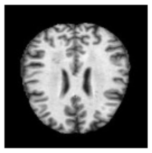


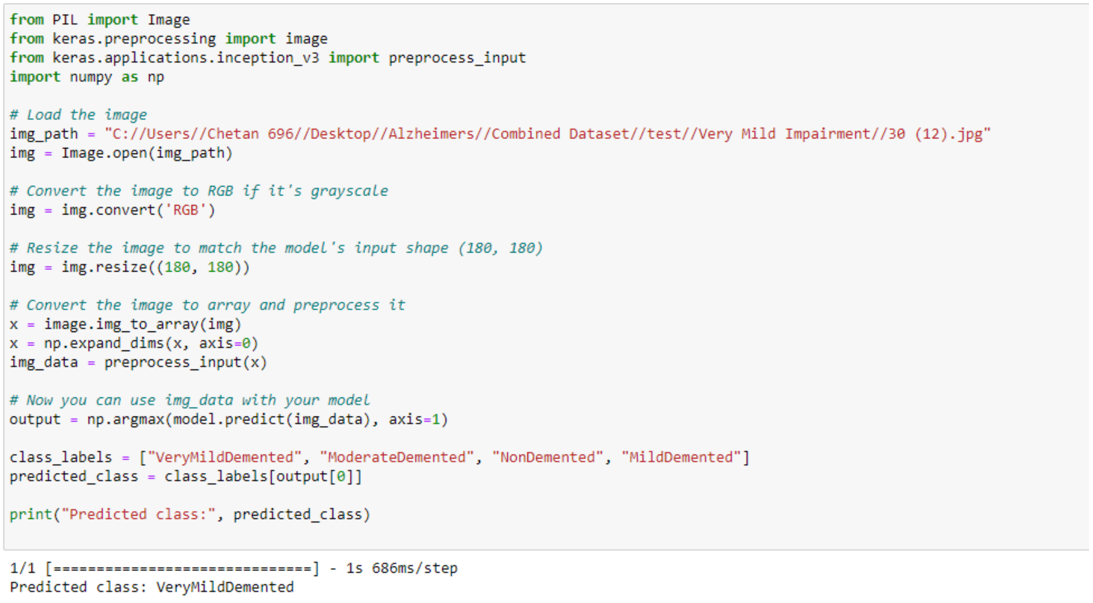
```
from PIL import Image
from keras.preprocessing import image
from keras.applications.inception_v3 import preprocess_input
import numpy as np

# Load the image
img_path = "C://Users//Chetan 696//Desktop//Alzheimers//Alzheimer_s Dataset//test//MildDemented//28.jpg"
img = Image.open(img_path)

# Convert the image to RGB if it's grayscale
img = img.convert('RGB')

# Resize the image to match the model's input shape (180, 180)
img = img.resize((180, 180))

# Convert the image to array and preprocess it
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
img_data = preprocess_input(x)

# Now you can use img_data with your model
output = np.argmax(model.predict(img_data), axis=1)

class_labels = ["MildDemented", "ModerateDemented", "NonDemented", "VeryMildDemented"]
predicted_class = class_labels[output[0]]

print("Predicted class:", predicted_class)
```

```
1/1 [==============================] - 0s 63ms/step
Predicted class: MildDemented
```



```
from IPython.display import display
from PIL import Image

# Use the file uploader to select an image
uploaded_image_path = "C://Users//Chetan 696//Desktop//Alzheimers//Combined Dataset//test//Very Mild Impairment//30 (12).jpg"
img = Image.open(uploaded_image_path)
display(img)
```





```
from PIL import Image
from keras.preprocessing import image
from keras.applications.inception_v3 import preprocess_input
import numpy as np

# Load the image
img_path = "C://Users//Chetan 696//Desktop//Alzheimers//Combined Dataset//test//Very Mild Impairment//30 (12).jpg"
img = Image.open(img_path)

# Convert the image to RGB if it's grayscale
img = img.convert('RGB')

# Resize the image to match the model's input shape (180, 180)
img = img.resize((180, 180))

# Convert the image to array and preprocess it
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
img_data = preprocess_input(x)

# Now you can use img_data with your model
output = np.argmax(model.predict(img_data), axis=1)

class_labels = ["VeryMildDemented", "ModerateDemented", "NonDemented", "MildDemented"]
predicted_class = class_labels[output[0]]

print("Predicted class:", predicted_class)
```

```
1/1 [==============================] - 1s 686ms/step
Predicted class: VeryMildDemented
```

10. Advantages and Disadvantages

10.1 Advantages

Early Detection –

Early identification of Alzheimer's disease through the Xception model facilitates timely intervention, potentially improving patient outcomes.

User-Friendly Interface –

The user interface allows seamless image input, enhancing accessibility for healthcare providers and ensuring ease of use.

Integration with Existing Frameworks –

The solution integrates with medical imaging analysis frameworks, leveraging the strengths of the Xception model within established healthcare systems.

Accurate Predictions –

The deep learning capabilities of Xception contribute to accurate predictions, aiding healthcare providers in making informed decisions.

Scalability –

The system is designed to scale efficiently, accommodating a growing volume of image data and an expanding user base.

10.2 Disadvantages

Dependency on Image Quality –

The accuracy of predictions may be influenced by the quality of input images, potentially leading to variations in performance.

Model Complexity –

The complexity of the Xception model may result in increased computational demands, requiring robust hardware infrastructure.

Training Data Limitations –

The effectiveness of the model is contingent on the availability and diversity of high-quality training data, which might pose challenges in certain scenarios.

Ethical and Privacy Concerns –

The use of medical imaging data raises ethical and privacy considerations, necessitating stringent measures to protect patient information.

Continuous Maintenance Requirements –

Regular updates and maintenance are essential to keep the system aligned with advancements in both medical imaging technology and deep learning methodologies.

11. Conclusion

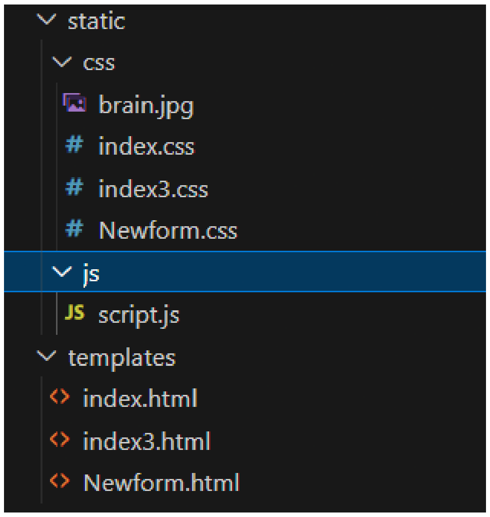
In conclusion, the proposed solution, leveraging the Xception model, stands out for its numerous positive attributes in addressing Alzheimer's disease detection challenges. The system excels in facilitating early intervention, providing a user-friendly interface, and seamlessly integrating with medical imaging analysis frameworks. Emphasizing accuracy, reliability, and scalability, the design ensures a robust tool that aligns with the evolving needs of healthcare providers. The positive aspects of the solution position it as a promising and effective resource in the early detection and management of Alzheimer's disease. The system's design prioritizes accuracy, reliability, and scalability to meet the evolving needs of healthcare providers.

12. Future Scope

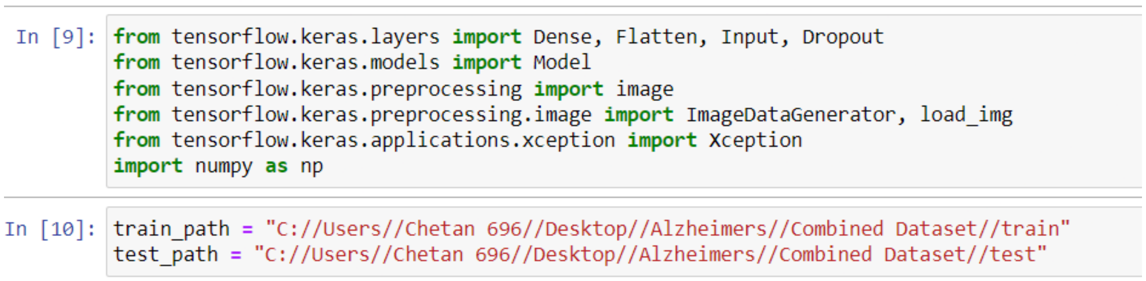
The future scope of this project involves refining the model through continuous learning with diverse datasets, enhancing robustness against varying image qualities. Integration with emerging technologies like edge computing can address computational challenges, making the solution more accessible. Additionally, exploring avenues for collaboration with medical institutions and leveraging advancements in neuroimaging techniques could further enhance the accuracy and applicability of the system. Continuous research and development will play a crucial role in ensuring the longevity and relevance of the proposed Alzheimer's disease detection solution.

13. Appendix

13.1 Source Code

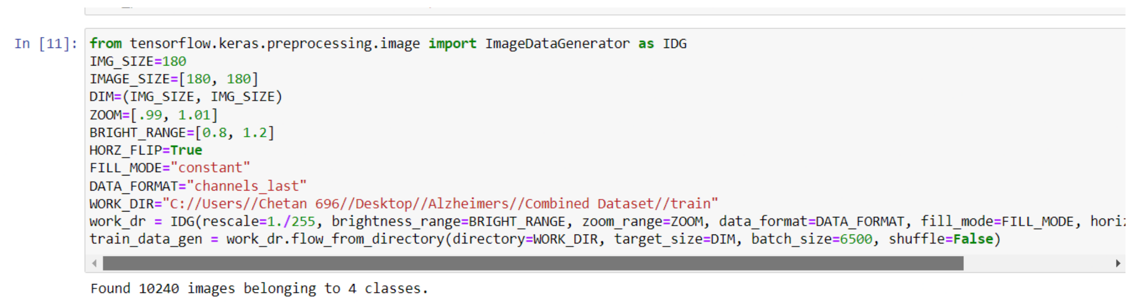


```
static
  css
    brain.jpg
    index.css
    index3.css
    Newform.css
  js
    script.js
templates
  index.html
  index3.html
  Newform.html
```



```
In [9]: from tensorflow.keras.layers import Dense, Flatten, Input, Dropout
        from tensorflow.keras.models import Model
        from tensorflow.keras.preprocessing import image
        from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img
        from tensorflow.keras.applications.xception import Xception
        import numpy as np
```

```
In [10]: train_path = "C://Users//Chetan 696//Desktop//Alzheimers//Combined Dataset//train"
         test_path = "C://Users//Chetan 696//Desktop//Alzheimers//Combined Dataset//test"
```



```
In [11]: from tensorflow.keras.preprocessing.image import ImageDataGenerator as IDG
         IMG_SIZE=180
         IMAGE_SIZE=[180, 180]
         DIM=(IMG_SIZE, IMG_SIZE)
         ZOOM=[.99, 1.01]
         BRIGHT_RANGE=[0.8, 1.2]
         HORZ_FLIP=True
         FILL_MODE="constant"
         DATA_FORMAT="channels_last"
         WORK_DIR="C://Users//Chetan 696//Desktop//Alzheimers//Combined Dataset//train"
         work_dr = IDG(rescale=1./255, brightness_range=BRIGHT_RANGE, zoom_range=ZOOM, data_format=DATA_FORMAT, fill_mode=FILL_MODE, hori
         train_data_gen = work_dr.flow_from_directory(directory=WORK_DIR, target_size=DIM, batch_size=6500, shuffle=False)

Found 10240 images belonging to 4 classes.
```

In [12]:
```
train_data,train_labels=train_data_gen.next()
```

In [13]:
```
print(train_data.shape,train_labels.shape)
```

```
(6500, 180, 180, 3) (6500, 4)
```

In [14]:
```
from sklearn.model_selection import train_test_split

# Splitting the data into train, test, and validation sets
train_data, test_data, train_labels, test_labels = train_test_split(train_data, train_labels, test_size=0.2, random_state=42)
train_data, val_data, train_labels, val_labels = train_test_split(train_data, train_labels, test_size=0.2, random_state=42)
```

In [15]:
```
from keras.applications import Xception

IMAGE_SIZE = [180, 180]
input_shape = tuple(IMAGE_SIZE + [3])

xception_model = Xception(input_shape=input_shape, include_top=False, weights='imagenet')
```

In [16]:
```
for layer in xception_model.layers:
    layer.trainable = False
```

In [17]:
```
from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import SeparableConv2D, BatchNormalization, GlobalAveragePooling2D

custom_inception_model = Sequential([
    xception_model,
    Dropout(0.5),
    GlobalAveragePooling2D(),
    Flatten(),
    Dense(512, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(256, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(128, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(64, activation='relu'),
    Dropout(0.5),
    BatchNormalization(),
    Dense(4, activation='softmax')
], name="inception_cnn_model")
```

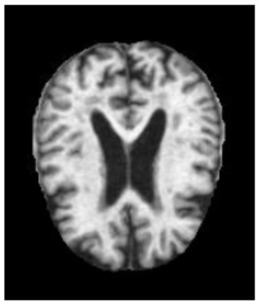
In [18]:
```
import tensorflow as tf
METRICS = [
    tf.keras.metrics.CategoricalAccuracy(name='acc'),
    tf.keras.metrics.AUC(name='auc')
]

custom_inception_model.compile(optimizer='rmsprop', loss=tf.losses.CategoricalCrossentropy(), metrics=METRICS)
```

```
custom_inception_model.save("Xception_model.h5")
```

```
C:\Users\Chetan 696\anaconda3\Lib\site-packages\keras\src\engine\training.py:3079: UserWarning: You are saving your model as an
HDF5 file via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `
model.save('my_model.keras')`.
  saving_api.save_model(
```

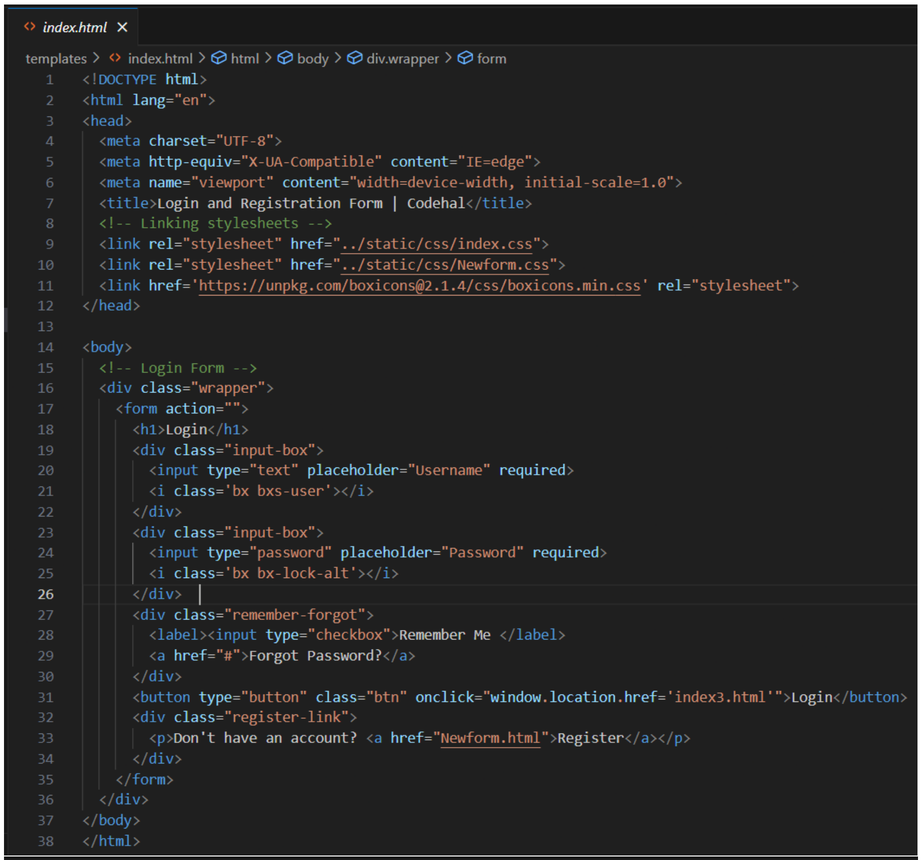
```
import numpy as np
from keras.preprocessing import image
from keras.applications.xception import Xception, preprocess_input, decode_predictions
import numpy as np
from keras.preprocessing import image
from keras.models import load_model
```

```
model_path = "C://Users//Chetan 696//Desktop//Alzheimers//Xception_model_image.h5"
model = load_model(model_path)
```

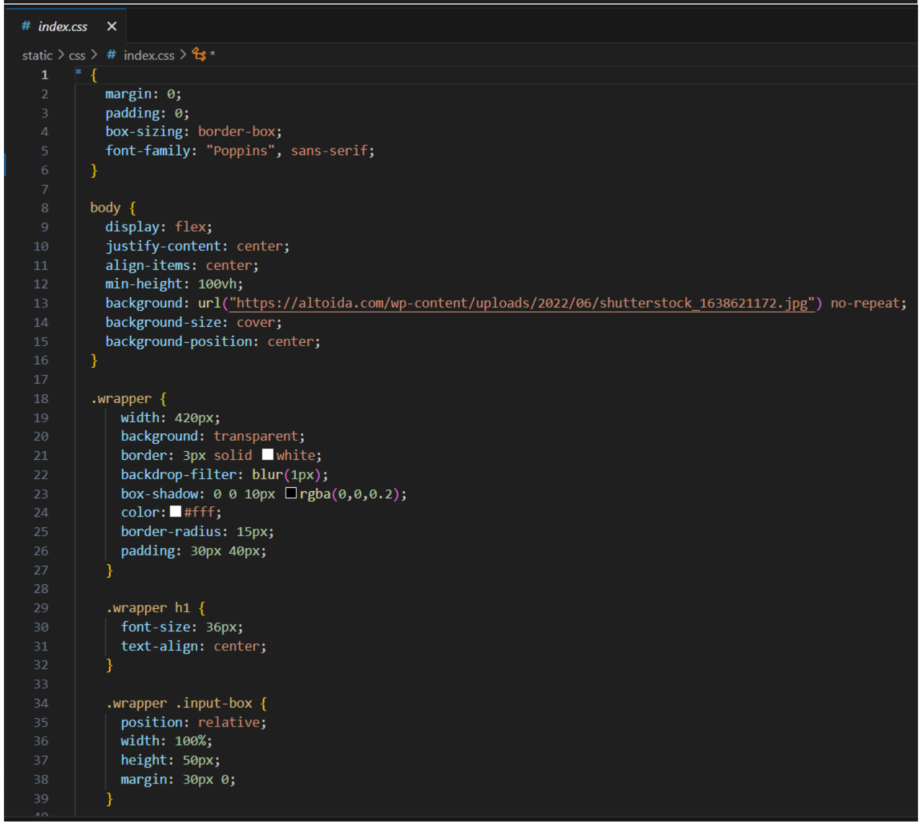


```python
from PIL import Image
from keras.preprocessing import image
from keras.applications.inception_v3 import preprocess_input
import numpy as np

# Load the image
img_path = "C://Users//Chetan 696//Desktop//Alzheimers//Alzheimer_s Dataset//test//MildDemented//28.jpg"
img = Image.open(img_path)

# Convert the image to RGB if it's grayscale
img = img.convert('RGB')

# Resize the image to match the model's input shape (180, 180)
img = img.resize((180, 180))

# Convert the image to array and preprocess it
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
img_data = preprocess_input(x)

# Now you can use img_data with your model
output = np.argmax(model.predict(img_data), axis=1)

class_labels = ["MildDemented", "ModerateDemented", "NonDemented", "VeryMildDemented"]
predicted_class = class_labels[output[0]]

print("Predicted class:", predicted_class)
```

```
1/1 [==============================] - 0s 63ms/step
Predicted class: MildDemented
```

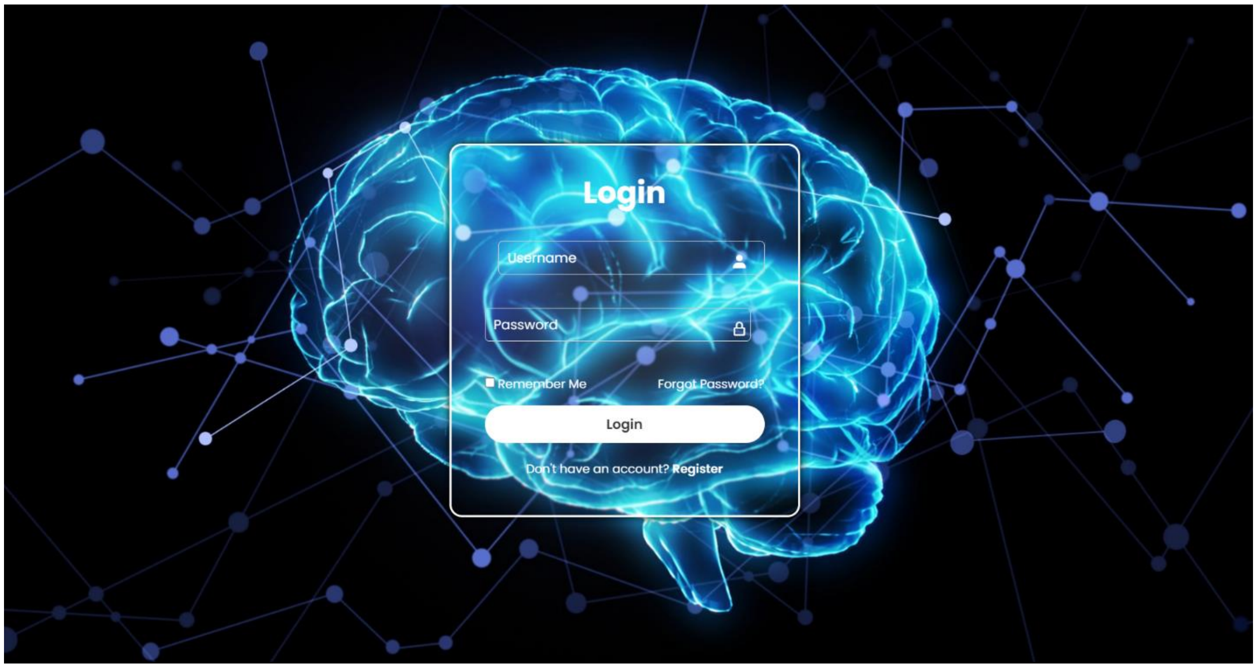


```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login and Registration Form | Codehal</title>
  <!-- Linking stylesheets -->
  <link rel="stylesheet" href="../static/css/index.css">
  <link rel="stylesheet" href="../static/css/Newform.css">
  <link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel="stylesheet">
</head>

<body>
  <!-- Login Form -->
  <div class="wrapper">
    <form action="">
      <h1>Login</h1>
      <div class="input-box">
        <input type="text" placeholder="Username" required>
        <i class='bx bxs-user'></i>
      </div>
      <div class="input-box">
        <input type="password" placeholder="Password" required>
        <i class='bx bx-lock-alt'></i>
      </div>
      <div class="remember-forgot">
        <label><input type="checkbox">Remember Me </label>
        <a href="#">Forgot Password?</a>
      </div>
      <button type="button" class="btn" onclick="window.location.href='index3.html'">Login</button>
      <div class="register-link">
        <p>Don't have an account? <a href="Newform.html">Register</a></p>
      </div>
    </form>
  </div>
</body>
</html>
```



```css
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Poppins", sans-serif;
}

body {
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
  background: url("https://altoida.com/wp-content/uploads/2022/06/shutterstock_1638621172.jpg") no-repeat;
  background-size: cover;
  background-position: center;
}

.wrapper {
    width: 420px;
    background: transparent;
    border: 3px solid white;
    backdrop-filter: blur(1px);
    box-shadow: 0 0 10px rgba(0,0,0.2);
    color: #fff;
    border-radius: 15px;
    padding: 30px 40px;
  }

  .wrapper h1 {
    font-size: 36px;
    text-align: center;
  }

  .wrapper .input-box {
    position: relative;
    width: 100%;
    height: 50px;
    margin: 30px 0;
  }
```

Login Page-You have an account then it will directly be redirected to upload and predict page.



Login

Username

Password

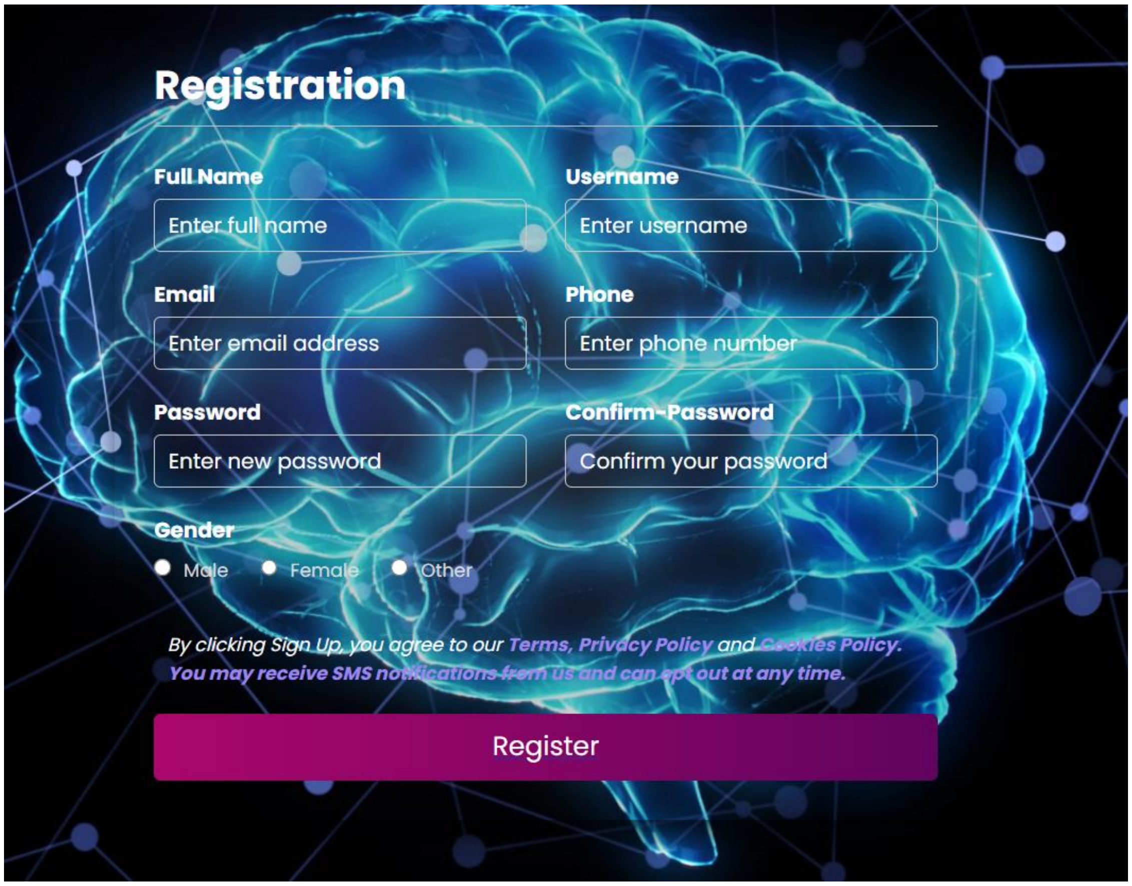
Remember Me

Forgot Password?

Login

Don't have an account? **Register**

If you do not have an account, you can register a new account and come back to login page and login (Automatically redirected after registering)



Registration

Full Name

Enter full name

Username

Enter username

Email

Enter email address

Phone

Enter phone number

Password

Enter new password

Confirm-Password

Confirm your password

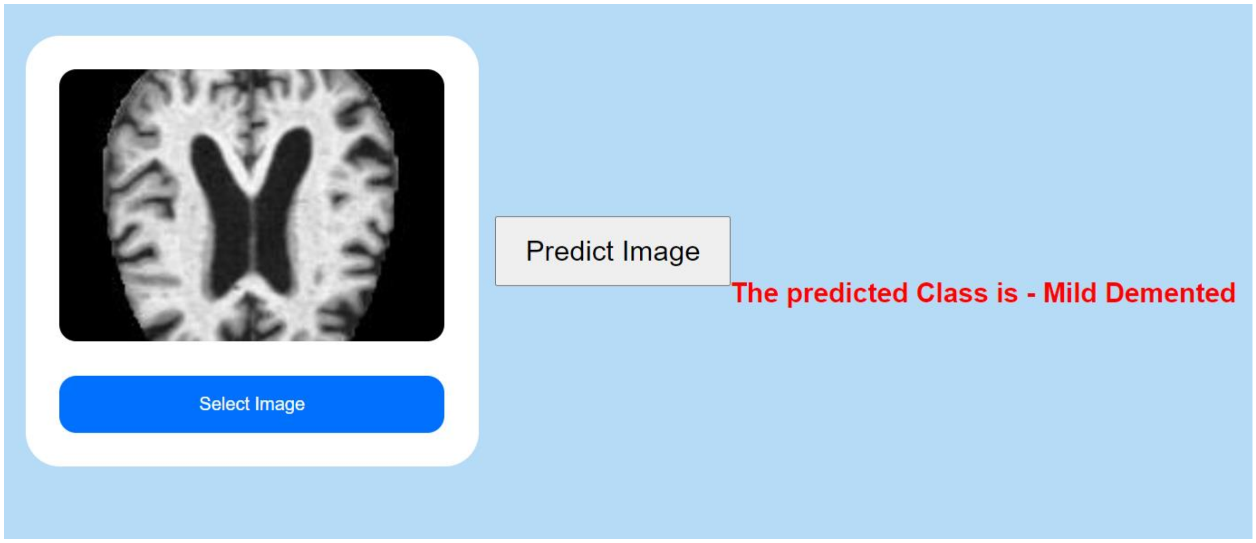
Gender

Male Female Other

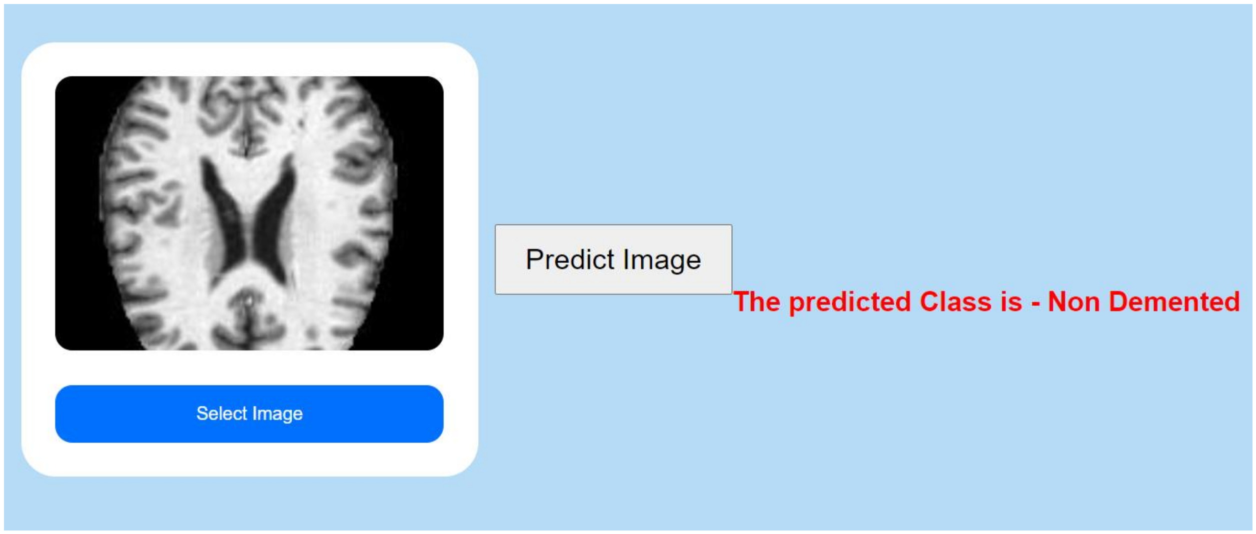
By clicking Sign Up, you agree to our Terms, Privacy Policy and Cookies Policy.
You may receive SMS notifications from us and can opt out at any time.

Register

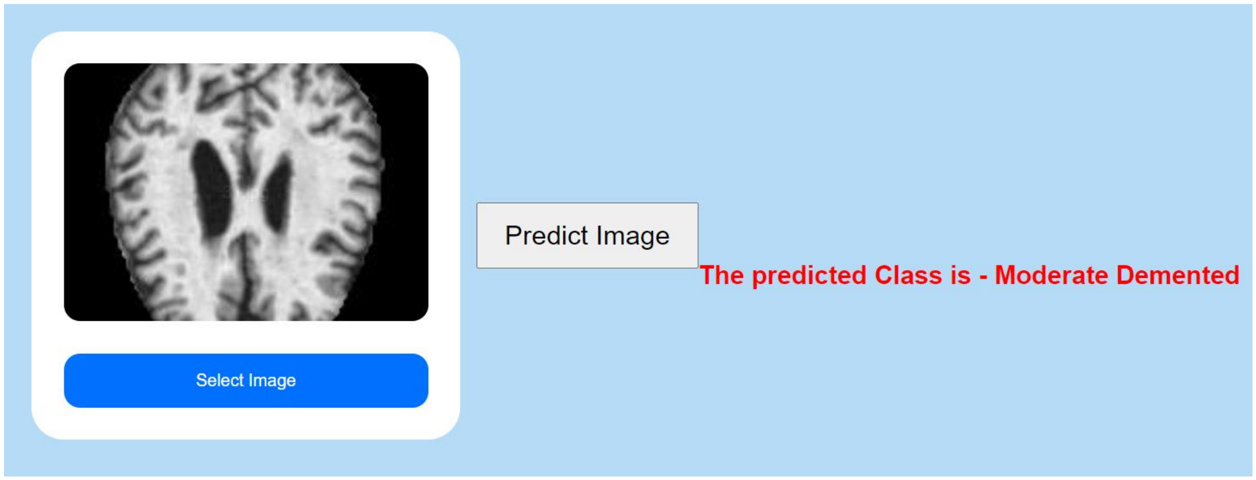
Testing the prediction by uploading each class images -



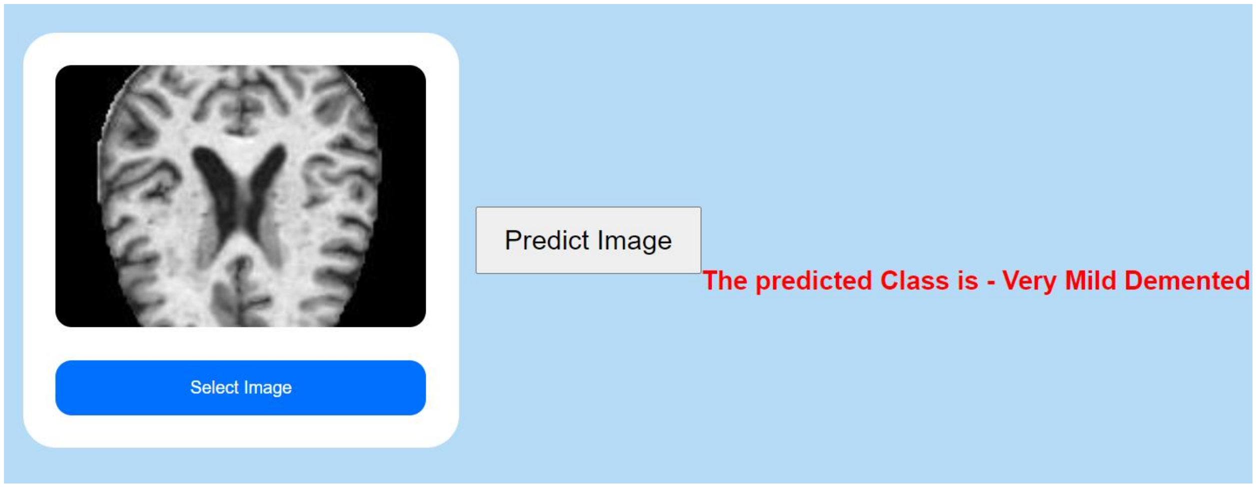
The predicted Class is - Mild Demented



The predicted Class is - Non Demented



The predicted Class is - Moderate Demented



The predicted Class is - Very Mild Demented