Minoufia University

Faculty of computers & Information

Graduation Project 2021/2022

Team Members

Marvy Ayman

Ghada Ahmed

Fatma Ayman

Fawzia Faraag

Shimaa Radwan

Project Name

7yah

Lung Cancer Detection

Supervisor

Dr. Mohammed Adel

TABLE OF CONTENTS

ACKNOWLEDGEMENT

Firstly, we would like to thank **Allah** and express our gratitude to all those who share their knowledge and offer their help to let us understand and complete this project. Then we are so grateful to our supervisor

Dr. Mohamed Adel

For the guidance, inspiration, and constructive suggestions that helped us in the preparation and execution of the project.

we would like to acknowledge all the assistance and contributions of **Faculty of Computer and Information of Menoufia University** for supporting us with all that is needed starting from the books and ending with the full care that it provided us with, to help us to be professionals in the field of Information Technology.

We sincerely thank our parents, families, and friends for all the support, encouragement, and patience they have provided us with throughout the project.

ABSTRACT

Lung cancer is one of the most deadly types of cancer. Thousands of people have this type of cancer, and if they don't detect it in the early stages of the disease, the patient's chance of survival will be very low. For the reasons suggested above and to help overcome this terrible early diagnosis with the help of AI procedures, there is an urgent need. It is also one of the most common types of cancer and contributes to mortality among all types of cancer. Lung cancer cases are increasing rapidly. There are about 70,000 cases annually in India. Over the past decade, cancer detection using deep learning models has been a hot topic, particularly in medical image classification. It should be noted that RESNET 101V2 is more advanced in treating diagnosed diseases such as lung cancer due to the high performance and high capacity of RESNET. This system presents an approach that uses RESNET to classify tumors in the lungs as malignant or benign and their type: adenocarcinoma, large cell carcinoma, squamous cell carcinoma. The accuracy obtained by the model is 98.5%, which is more efficient when compared to the accuracy obtained by existing conventional systems.

1 CHAPTER ONE: INTRODUCTION

1.1 OVERVIEW

7hya is a medical application used to screen for lung cancer x-rays. This application has several possibilities, including X-ray examination as mentioned previously the ability to search and inquire between doctors and patients, as well as the possibility of taking and analyzing an x-ray image on a mobile phone. There is also a section to identify the symptoms of the disease and how to avoid it.

1.2 PROBLEM DEFINITION

It is well known among the people that since a person does not feel pain, he is fine, right? A lot of problems occur because of this act that does not result from knowledge. Minor diseases worsen and cannot be easily treated after that, and some cases are waiting for their death, so work has been done to solve this problem by facilitating the examination of people.

1.3 OBJECTIVES

- Ease of examination.
- Confirmation of doctor's diagnosis.
- Create a channel of communication between the patient and the doctor.
- Spreading awareness among people about the disease and its symptoms and how to reduce or avoid it.

1.4 TOOLS

- Brackets IDE
- SQL Workbench
- Visual Studio Code
- Git

- Postman
- Anaconda
- Android Studio
- Heroku CLI

1.5 LITERATURE OF LUNG CANCER

1.5.1 Overview

Lung cancer is the second most common cancer, accounting for about one out of five malignancies in men and one out of nine in women. Unfortunately, over the past several years, while the incidence of lung cancer has gradually declined in men, it has been rising alarmingly in women. In 1940 only seven women in 100,000 developed the disease; today the rate is 42 in 100,000. And all the evidence points to smoking as the cause. As one specialist in the field reports, "How long it takes to get cancer depends on how many cigarettes you smoke a day." However, studies prove that quitting smoking does lower the risk.

1.5.2 Symptoms

Sometimes lung cancer does not cause any signs or symptoms. It may be found during a chest x-ray done for another condition. If you do have symptoms, they may include:

- Chest pain or discomfort
- A cough that doesn't go away or gets worse over time
- Trouble breathing
- Wheezing
- Blood in sputum (mucus coughed up from the lungs)

- Hoarseness
- Loss of appetite
- Weight loss for no known reason
- Fatigue
- Trouble swallowing
- Swelling in the face and/or veins in the neck

1.5.3 Risk

Lung cancer can affect anyone, but there are certain factors that raise your risk of getting it:

Smoking this is the most important risk factor for lung cancer. Tobacco smoking causes about 9 out of 10 cases of lung cancer in men and about 8 out of 10 cases of lung cancer in women. The earlier in life you start smoking, the longer you smoke, and the more cigarettes you smoke per day, the greater your risk of lung cancer. The risk is also greater if you smoke a lot and drink alcohol every day or take beta carotene supplements. If you have quit smoking, your risk will be lower than if you had kept smoking. But you will still have a higher risk than people who never smoked. Secondhand smoke is the combination of smoke that comes from a cigarette and smoke breathed out by a smoker. When you inhale it, you are exposed to the same cancer-causing agents as smokers, although in smaller amounts. Family history of lung cancer being exposed to asbestos, arsenic, chromium, beryllium, nickel, soot, or tar in the workplace Being exposed to radiation, such as from Radiation therapy to the breast or chest Radon in the home or workplace Certain imaging tests such as

- CT scans
- HIV infection

- Air pollution

1.5.4 Consequences

Lung cancer can affect more than just your lungs. Once you have a lung tumor, cancer cells can break away and form new tumors nearby or if stray cancer cells enter the lymphatic system or bloodstream, they can travel to other parts of the body. This process is called the secondary growth of malignant tumors. Lung cancer tends to spread to:

- lymph
- a necklace
- bones
- brain
- liver
- next to the college
- glands

At first, it affects only the lungs and respiratory system. Other symptoms vary, depending on where cancer has moved.

1.5.5 Screening

Cancer screening strategies are used to identify early lung cancers before they cause symptoms, at a point where they are more likely to be curable. Lung cancer screening is critically important because of the incidence and prevalence of lung cancer. More than 235,000 new cases of lung cancer are expected in the United States in 2021 with approximately 130,000 deaths expected in 2021. In addition, at the time of diagnosis, 57% of lung cancers are discovered in advanced stages (III and IV), and meaning they are more widespread or aggressive cancers. Because there is a substantially higher probability of long-term survival following treatment of localized (60%) versus advanced stage (6%) lung cancer, lung cancer screening aims to diagnose the disease in the localized (stage I) stage. Results from large

randomized studies have recently prompted a large number of professional organizations and governmental agencies in the U.S. to now recommend lung cancer screening in select populations. The 3 main types of lung cancer screening are low-dose, computerized tomographic (LDCT) screening, chest x-rays, and sputum cytology tests. Currently, multiple professional organizations, as well as the United States Preventive Services Task Force (USPSTF), the Centers for Medicare and Medicaid Services (CMS), and the European Commission's science advisors concur and endorse low-dose, computerized tomographic screening for individuals at high-risk of lung cancer.

2 CHAPTER TWO: SYSTEM OVERVIEW

2.1 FUNCTIONAL SYSTEM REQUIREMENTS

2.1.1 Software Requirements

- For web

Software Tools	Minimum Requirements
Platform	Windows
Operating System	Windows 8 or above
Technology	Database SQL Server – Python (Flask)
Scripting Language	Python
IDE	Sublime Text, Brackets

- For mobile application

Software Tools	Minimum Requirements
Platform	Android
Technology	Flutter
Scripting Language	Dart
IDE	Android Studio, VS Code

2.1.2 Hardware Requirements

Hardware Tools	Minimum Requirements
Processor	i3 or above
Ram	4 GB
Monitor	15.6” colored

Mouse	Optical
Hard Disk	At least 150M

2.2 SYSTEM USER'S IDENTIFICATIONS

Patient: this is the first user type of system. The user is seeking advice or help from doctors. The patient creates an account with username, name, email, job (patient) and password.

Doctor: he is the person who gives the advice for the user (patient). He can create or delete posts about medical info.

Admin: he is the person who is responsible for updating or changing the system capabilities or functions. He can also explore users, doctors, and data.

2.3 FUNCTIONAL USER REQUIREMENTS

- The system allows patients to search and view doctors' profiles.
- The system allows doctors to determine the status of the treatment and can take notes about the treatment.
- The system shall allow its user to update their personal information.
- The system allows patients to search and view medical posts.
- the system allows to its user to interact with comment on a post
- The system allows users to share a post in social media.
- The system allows its user to see the most popular posts in the blog.

2.4 NON-FUNCTIONAL REQUIREMENTS

2.4.1 Usability Requirements

Ease of access to information and ease of use of the program. With just a few clicks, the user can access the destination information.

2.4.2 Reliability & Up-time Requirements

Every data can be accessed and seen just after data entrance. The system will provide at least 99% uptime on web hosting sites. Reliability factors will be supplied through:

- Success track record
- Physical server security
- Disaster recovery plan

2.4.3 Security Requirements

The login system will protect the information in the system from outside users. All user types such as patient, doctors, and admin have distinct pages which only they can access. The password safety will be provided by the encryption of the passwords saved in the database.

2.4.4 Performance & Capability Requirements

The system can be applied to any community of doctors and their patients. The performance of the system will be appropriate for medical practitioners, which requires a high speed of interaction, and so all tasks will be carried out within a few clicks and seconds. The scalability requirements of the system are another important issue as well as the performance requirements.

2.4.5 Maintainability & Upgradeability Requirements

Making changes or upgradeability in the system will not be that much difficult. By having some knowledge of programming, some features of the system might be converted to a new version. According to the needs of upgrade, system requirements might change such as change in hardware or operating system or not.

2.4.6 Supportability & operability requirements

System will be quite easy to use but educational support will be given if needed. Although the system is completely open source. The system can be run on every computer with an internet access. The system can be installed for any operating system e.g., Microsoft Windows Vista/7/8/10, or Linux.

3 CHAPTER THREE: MACHINE LEARNING TECHNIQUES

3.1 ABSTRACT

Lung cancer is one of the leading causes of mortality in every country, affecting both men and women. Lung cancer has a low prognosis, resulting in a high death rate. The computing sector is fully automating it, and the medical industry is also automating itself with the aid of image recognition and data analytics. This paper endeavors to inspect accuracy ratio of some classifiers which is Support Vector Machine (SVM), K-Nearest Neighbor (KNN)and, Decision Tree that classify lung cancer in early stage so that many lives can be saving. Basically, the informational indexes utilized as a part of this examination are taken from images datasets for patients affected by lung cancer. The principle point of this paper is to the execution investigation of the classification algorithms accuracy by WEKA Tool. The experimental results show that SVM gives the best result with 85%, then KNN with KNN 81% and Decision Tree with 70%.

3.2 INTRODUCTION

Lung cancer is one of the causes of cancer deaths. It is difficult to detect because it arises and shows symptoms in final stage. However, mortality rate and probability can be reduced by early detection and treatment of the disease. Best imaging technique CT imaging are reliable for lung cancer diagnosis because it can disclose every suspected and unsuspected lung cancer nodules. However, variance of intensity in CT scan images and anatomical structure misjudgment by doctors and radiologists might cause difficulty in marking the cancerous cell. Recently, to assist radiologists and doctors detect the cancer accurately computer Aided Diagnosis has become supplement and promising tool. There has been many system developed and research going on detection of lung cancer. However, some systems do not have satisfactory accuracy of detection and some systems still has to be improved to achieve highest accuracy tending to 100%. Image processing

techniques and machine learning techniques has been implemented to detect and classify the lung cancer. We studied recent systems developed for cancer detection based on CT scan images of lungs to choose the recent best systems and analysis was conducted on them and new model was proposed.

3.3 MODELS

3.3.1 Decision Trees

Are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Some advantages of decision trees are:

- Simple to understand and to interpret. Trees can be visualized.
- Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
- Able to handle both numerical and categorical data. However scikit-learn implementation does not support categorical variables for now. Other techniques are usually specialized in analyzing datasets that have only one type of variable. See algorithms for more information.
- Able to handle multi-output problems.
- Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by Boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret.
- Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.
- Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.

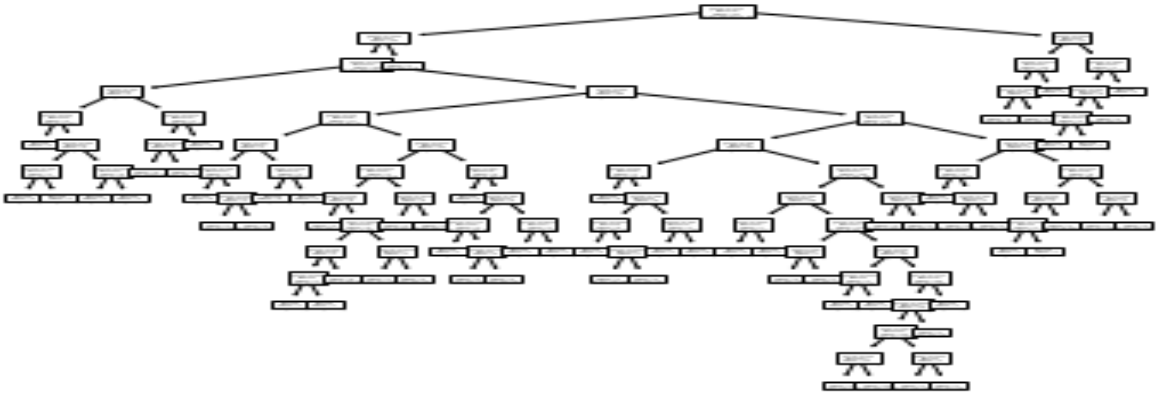
The disadvantages of decision trees include:

- Decision-tree learners can create over-complex trees that do not generalize the data well. This is called overfitting. Mechanisms such as pruning, setting the minimum number of samples required at a

leaf node or setting the maximum depth of the tree are necessary to avoid this problem.

- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.
- Predictions of decision trees are neither smooth nor continuous, but piecewise constant approximations as seen in the above figure. Therefore, they are not good at extrapolation.
- The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.
- There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.
- Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

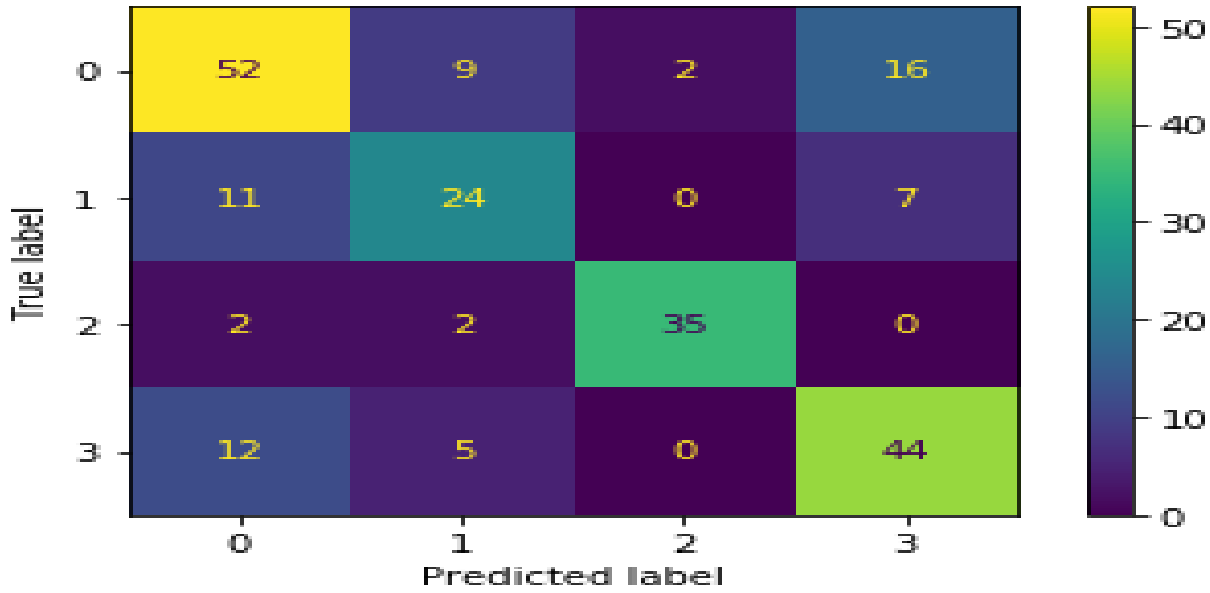
Tree display:



The implementation of decision tree

Accuracy: 0.7013574660633484
Precision: 0.7194967597952673
Recall: 0.7121009480938927
f1_score 0.7151462879760376

Decision tree Confusion Matrix Display:



3.3.2 Naïve Bayes

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other

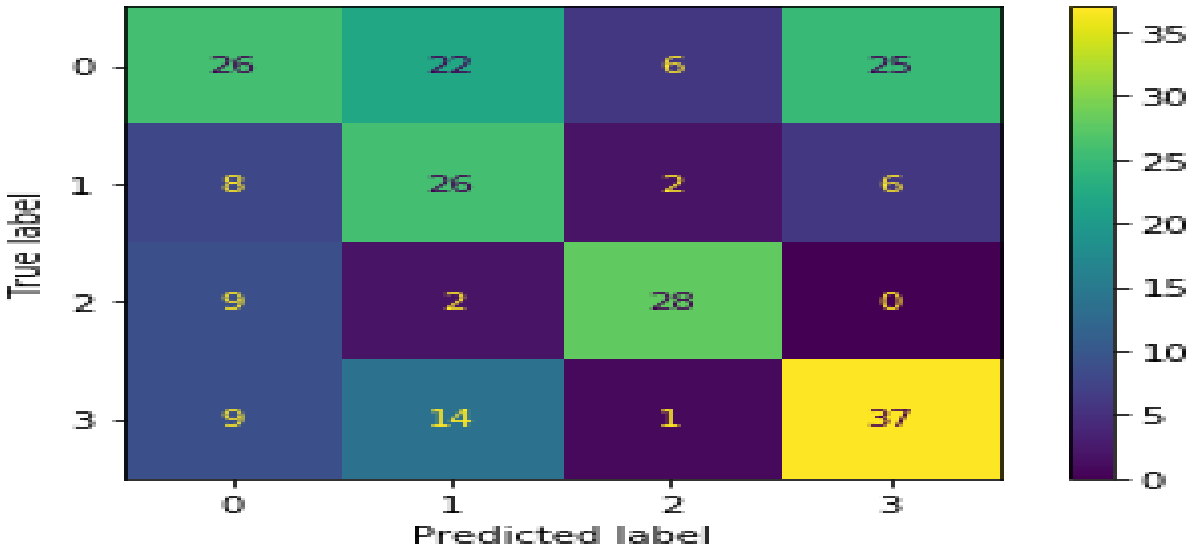
Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

$$P(A/B)=(P(B/A)*P(A))/P(B)$$

The implementation of Naive Bayes

Accuracy: 0.5294117647058824
Precision: 0.5517811009538951
Recall: 0.5681669095240376
f1_score 0.5494995296843042

Naive Bayes Confusion Matrix Display:



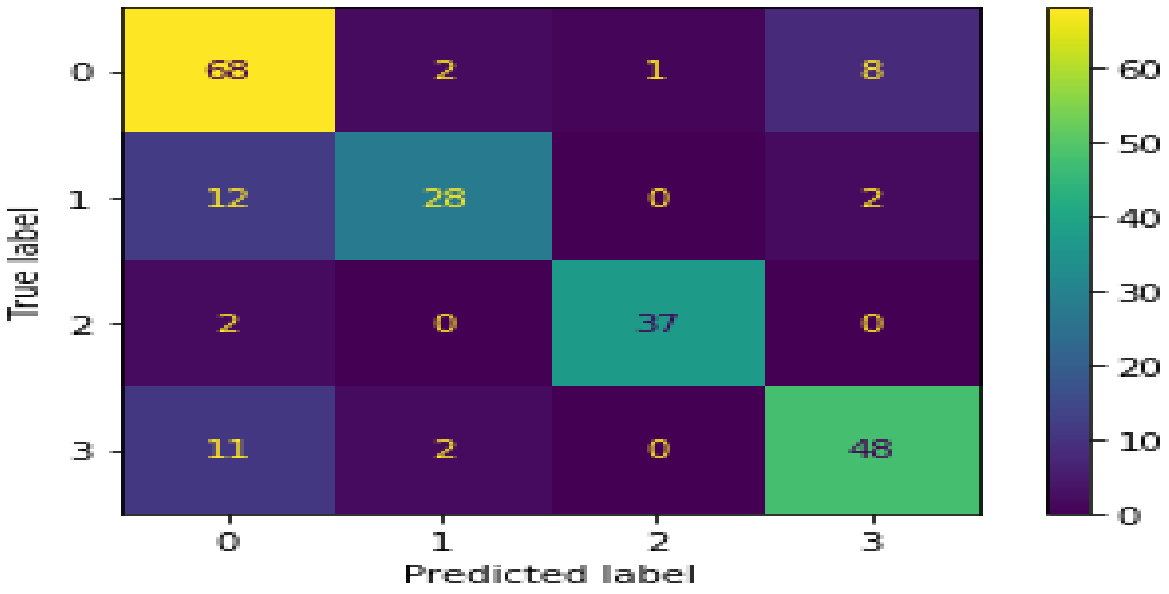
3.3.3 K-Nearest Neighbors

The K-nearest neighbors (KNN) algorithm is a type of supervised machine learning algorithms .KNN is extremely easy to implement in its most basic form, and yet performs quite complex classification tasks. It is a lazy learning algorithm since it doesn't have a specialized training phase. Rather, it uses all of the data for training while classifying a new data point or instance. KNN is a non-parametric learning algorithm, which means that it doesn't assume anything about the underlying data. This is an extremely useful feature since most of the real world data doesn't really follow any theoretical assumption e.g. linear-reparability, uniform distribution, etc.

The implementation of KNN

Accuracy: 0.8190045248868778
Precision: 0.851863303280448
Recall: 0.8157573387392851
f1_score 0.8288040203224882

KNN Confusion Matrix Display:



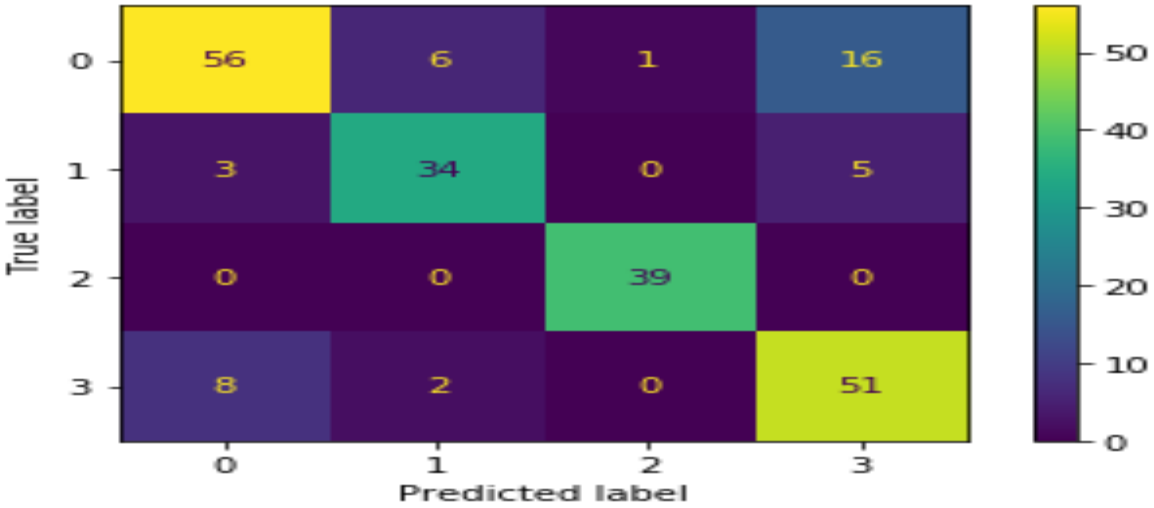
3.3.4 Logistic model

In statistics, the (binary) logistic model (or logit model) is a statistical model that models the probability of one event (out of two alternatives) taking place by having the log-odds (the logarithm of the odds) for the event be a linear combination of one or more independent variables ("predictors"). In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (the coefficients in the linear combination). Formally, in binary logistic regression there is a single binary dependent variable, coded by a indicator variable, where the two values are labeled "0" and "1", while the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names

The implementation of Logistic Regression

Accuracy: 0.8144796380090498
Precision: 0.8321695095948828
Recall: 0.8386125356969931
f1_score 0.832726540645006

Logistic Regression Confusion Matrix Display:



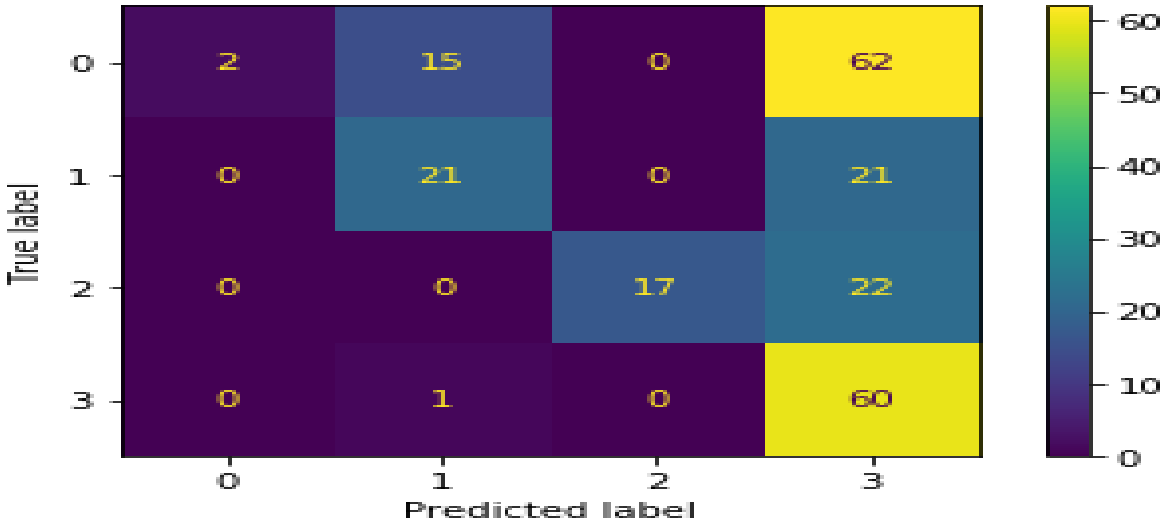
3.3.5 Neural Network

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria.

The implementation of neural network

Accuracy: 0.45248868778280543
Precision: 0.7328009828009828
Recall: 0.48620511224267193
f1_score 0.42978614853498165

Neural network Confusion Matrix Display:



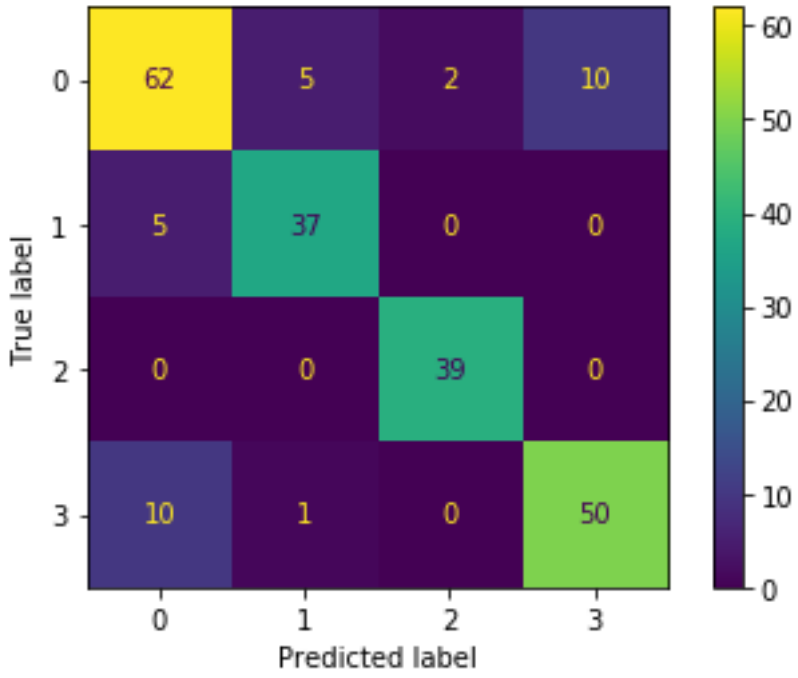
3.3.6 Support Vector Machine

"Support Vector Machine" (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well. Support Vector Machines with ease. It has helper functions as well as code for the Naive Bayes Classifier. The creation of a support vector machine in R and Python follow similar approaches

The implementation of SVM

Accuracy: 0.8506787330316742
Precision: 0.8625531917505825
Recall: 0.8713586596705502
f1_score 0.866726577789412

SVM Confusion Matrix Display:

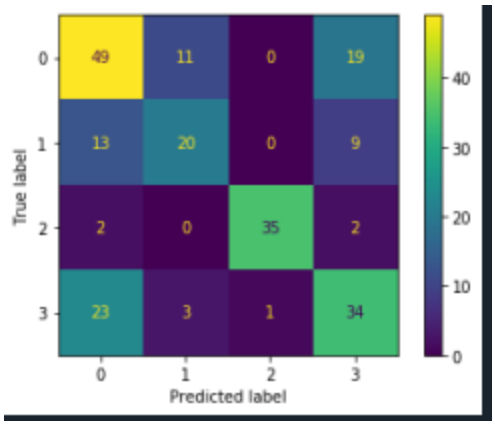


3.4 PREPROCESSING

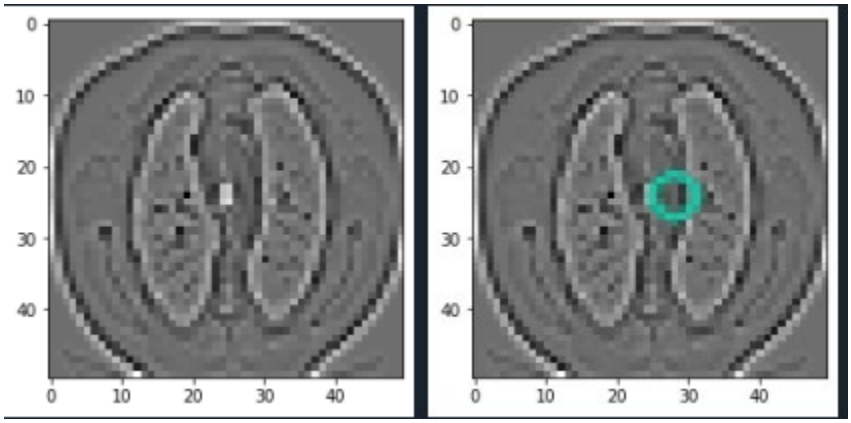
Accuracy after laplacian

accuracy: 0.6244343891402715
Precision: 0.6244343891402715
Recall: 0.6244343891402715
f1_score 0.6244343891402715

ConfusionMatrixDisplay:



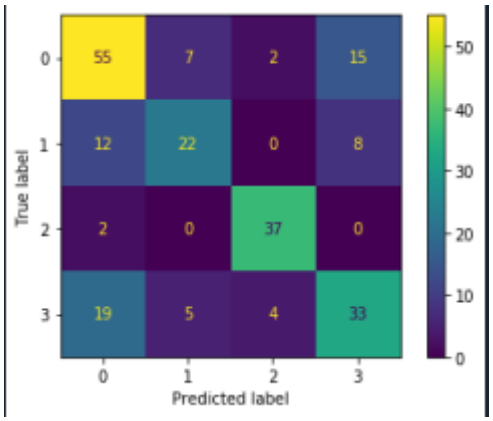
Sample from laplacian **Laplacian after sift**



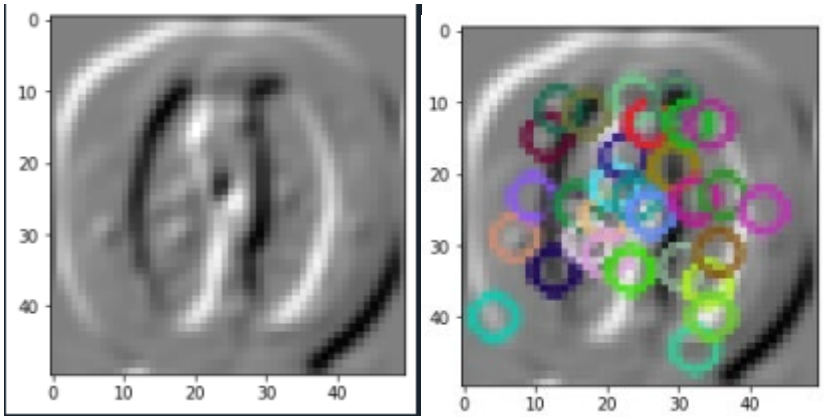
Accuracy after sobel edge detection

accuracy: 0.665158371040724
Precision: 0.665158371040724
Recall: 0.665158371040724
f1_score 0.665158371040724

ConfusionMatrixDisplay:



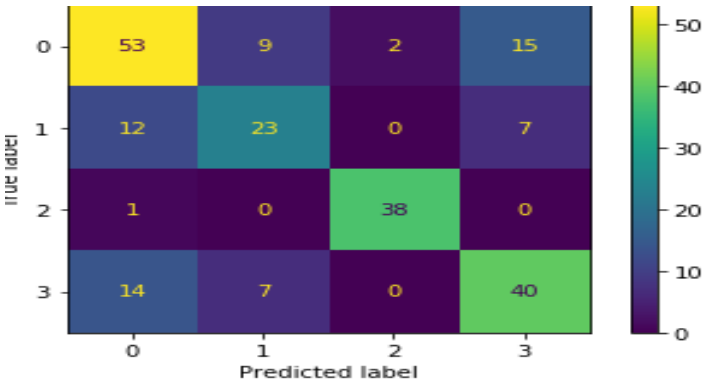
Sample from sobel **Sobel after sift**



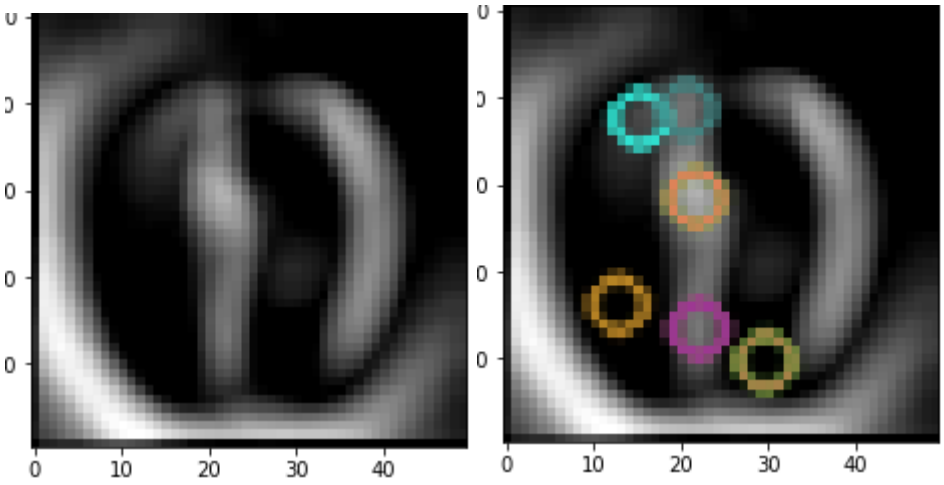
Accuracy after prewitt edge detection

Accuracy: 0.6968325791855203
Precision: 0.6968325791855203
Recall: 0.6968325791855203
f1_score 0.6968325791855203

ConfusionMatrixDisplay



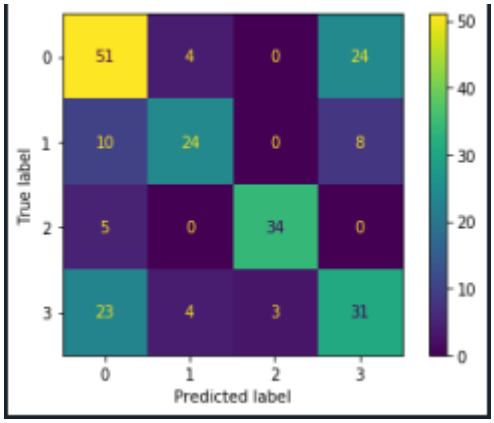
Sample from prewitt **After sift**



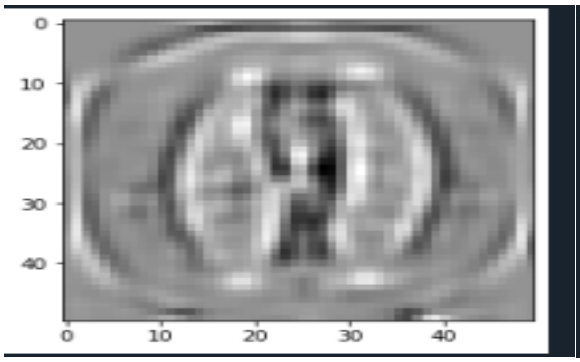
Accuracy after sobel and prewitt

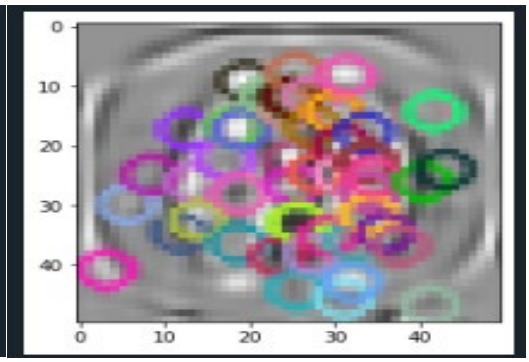
accuracy: 0.6334841628959276
Precision: 0.6334841628959276
Recall: 0.6334841628959276
f1_score 0.6334841628959276

ConfusionMatrixDisplay



Sample from edge detection **After sift**

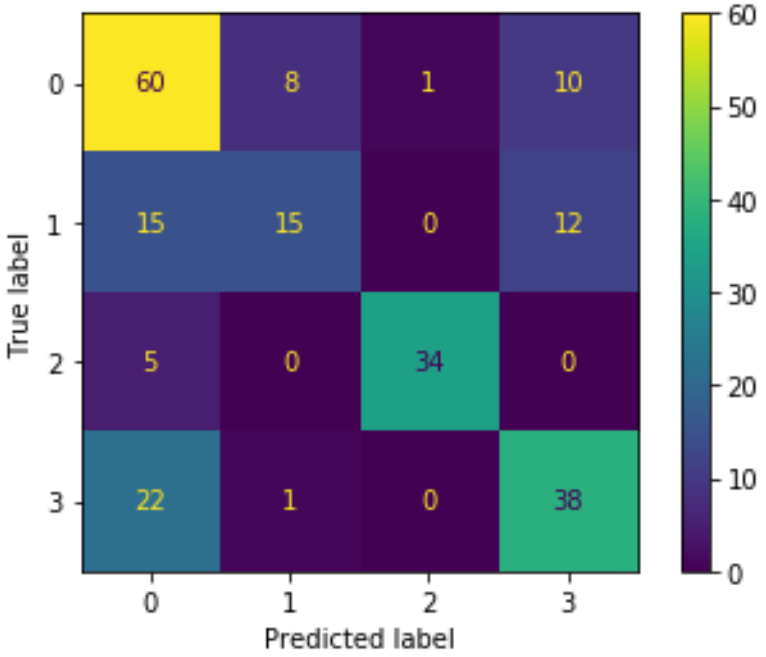




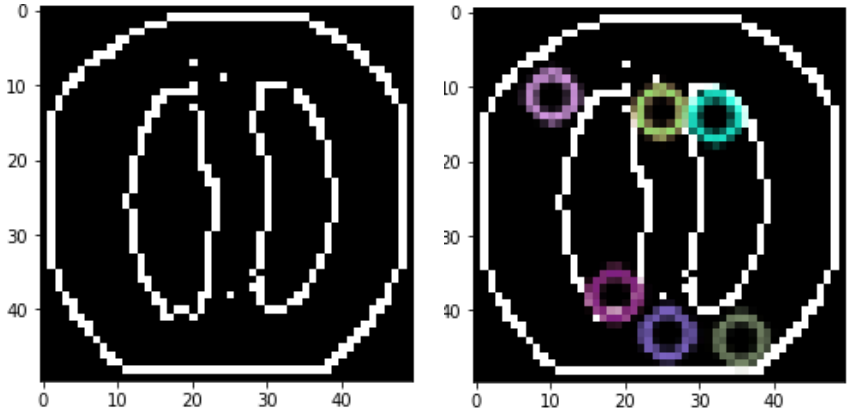
Accuracy after canny

Accuracy: 0.665158371040724
Precision: 0.665158371040724
Recall: 0.665158371040724
f1_score 0.665158371040724

ConfusionMatrixDisplay



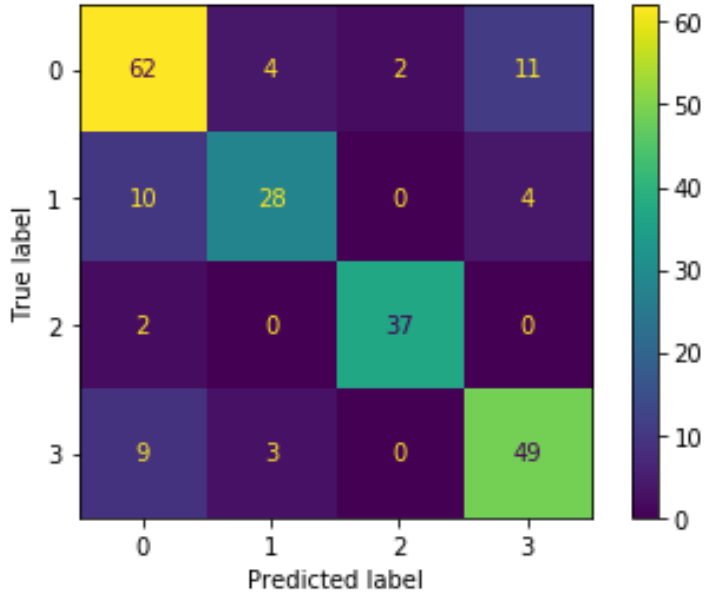
Sample from canny **After sift**



Accuracy after pca

Accuracy: 0.7963800904977375
Precision: 0.7963800904977375
Recall: 0.7963800904977375
f1_score 0.7963800904977375

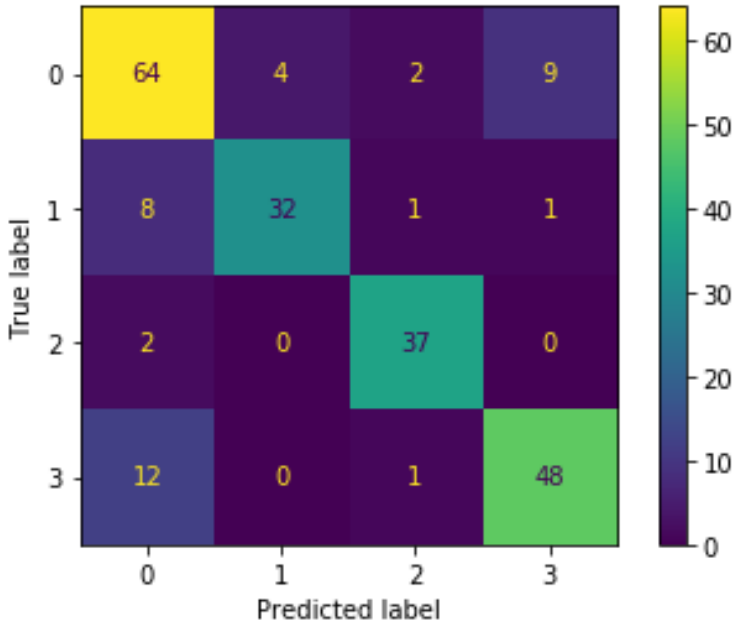
ConfusionMatrixDisplay:



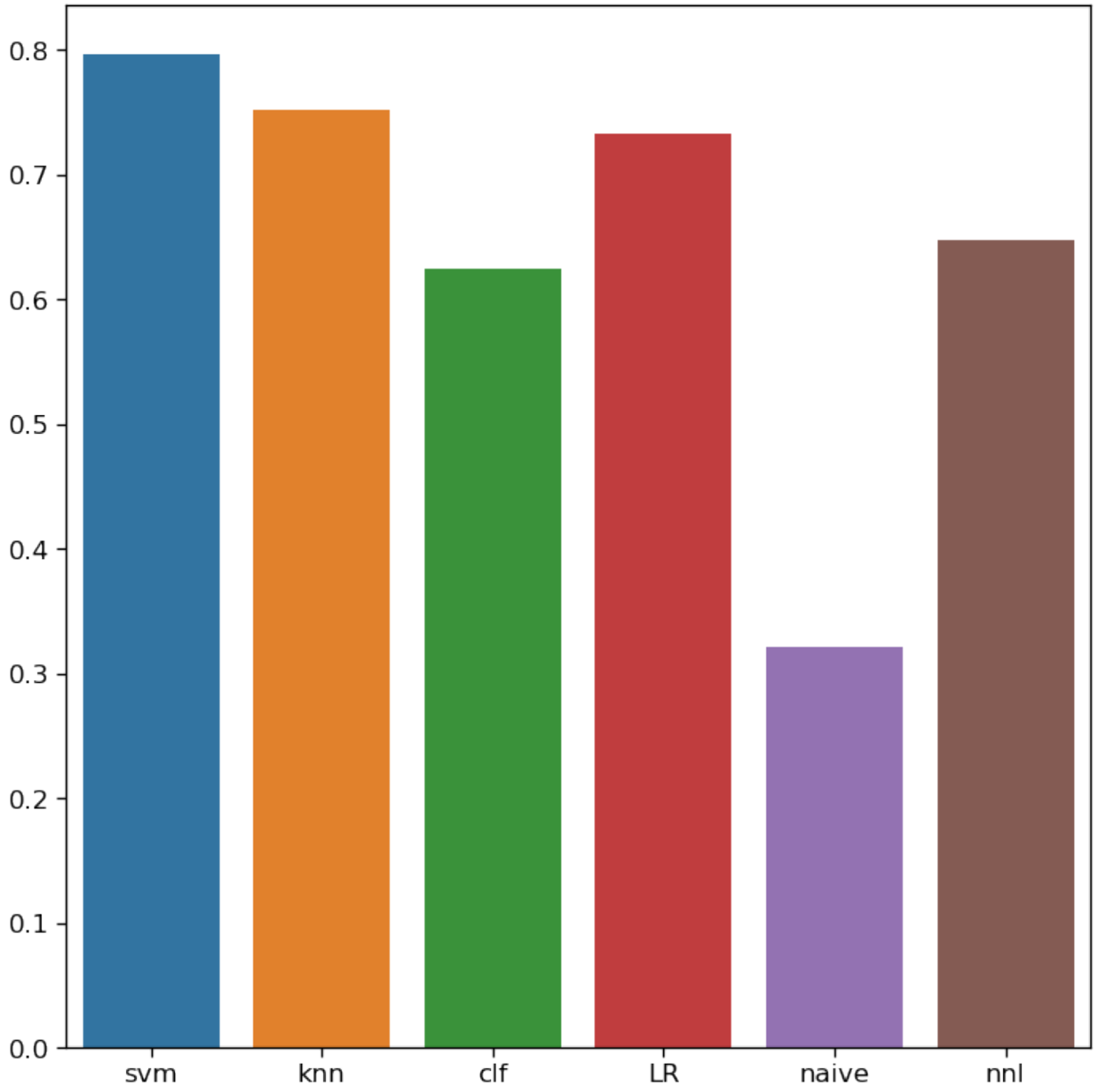
Accuracy after hyperparameter

Accuracy: 0.8190045248868778
Precision: 0.8190045248868778
Recall: 0.8190045248868778
f1_score 0.8190045248868778

ConfusionMatrixDisplay:



3.5 Compare Between All Models



The result presented above SVM has best performance followed by all models after preprocessing

4 CHAPTER FOUR: DEEP LEARNING TECHNIQUES

4.1 DATA STORY

It was a project about chest cancer detection using machine learning and deep learning (Resent 101V2) .we classify and diagnose if the patient have cancer or not using AI model .

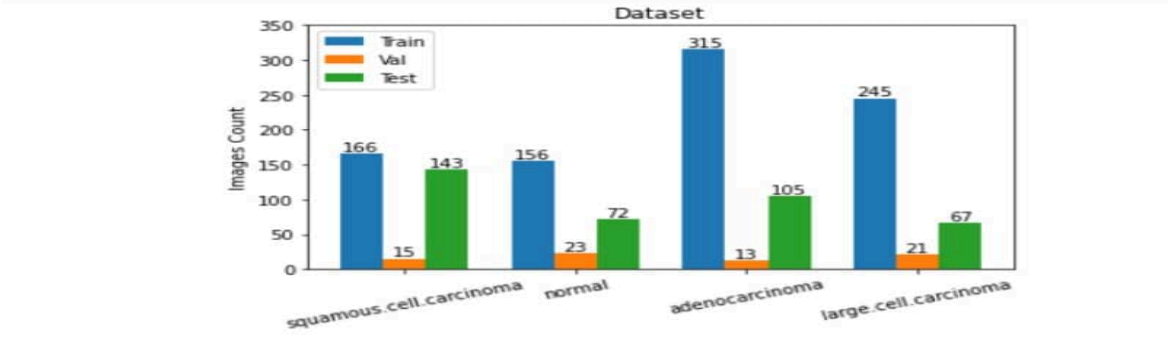
4.2 DATA PREPARATION

Images are not in DCM format, the images are in jpg or PNG to fit the model.

4.2.1 Data contain:

- 3 chest cancer types which are Adenocarcinoma, Large cell carcinoma, Squamous cell carcinoma
- 1 folder for the normal cell

Data folder is the main folder that contains all the step folders inside Data folder are test, train, valid.

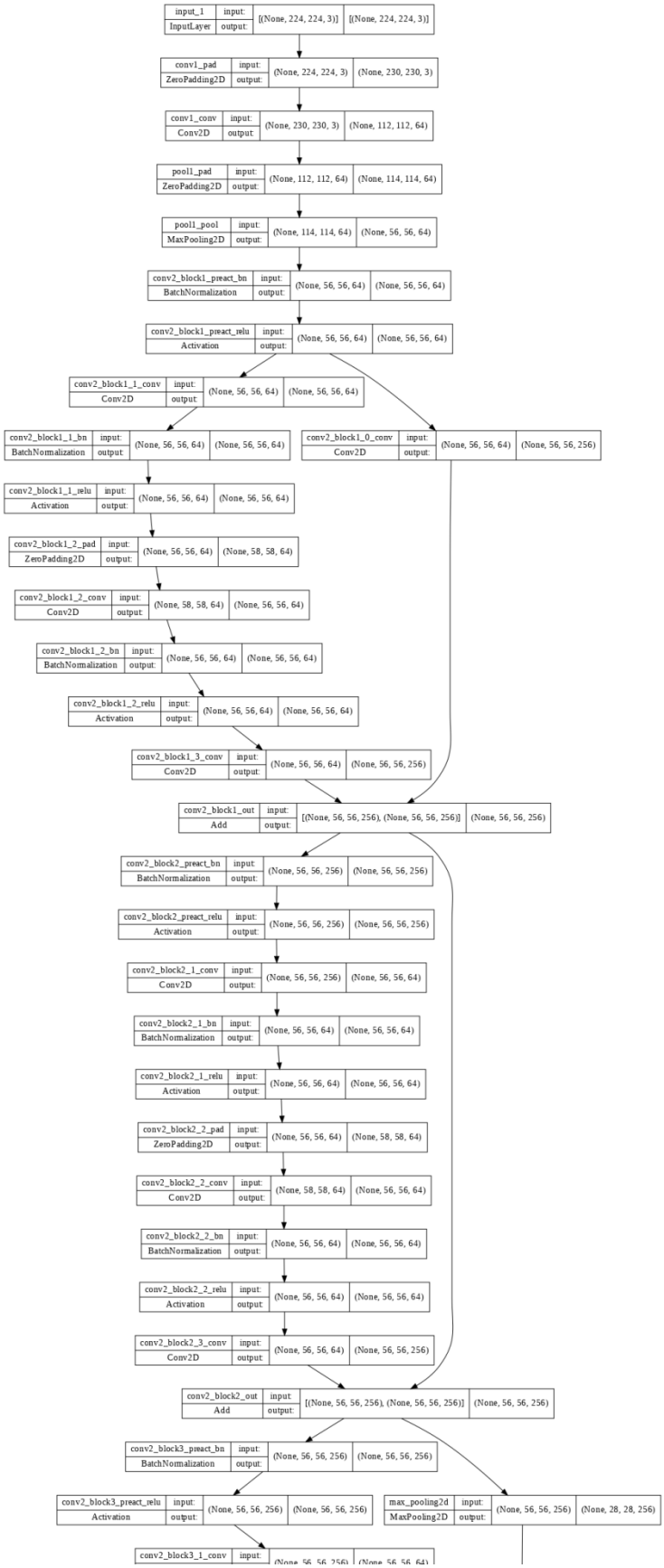


Train:: 882 images belonging to 4 classes.
Valid:: 72 images belonging to 4 classes.
Test:: 387 images belonging to 4 classes.

4.3 RESENT MODEL BLOCKS

model.summary():

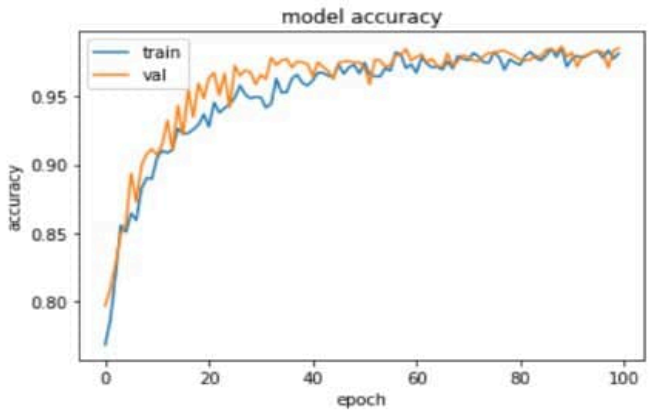
```
Layer (type)                 Output Shape              Param #   Connected to
==================================================================================
input_1 (InputLayer)         [(None, 224, 224, 3  0           []
)]

conv1_pad (ZeroPadding2D)      (None, 230, 230, 3)  0          ['input_1[0][0]']

conv1_conv (Conv2D)          (None, 112, 112, 64  9472        ['conv1_pad[0][0
']']
)

pool1_pad (ZeroPadding2D)      (None, 114, 114, 64  0          ['conv1_conv[0]
[0]']
)

pool1_pool (MaxPooling2D)      (None, 56, 56, 64)   0          ['pool1_pad[0][0]
']

conv2_block1_preact_bn (BatchN  (None, 56, 56, 64)  256         ['pool1_pool[
0][0]']
ormalization)

conv2_block1_preact_relu (Acti  (None, 56, 56, 64)  0           ['conv2_block1_p
react_bn[0][0]']
vation)

conv2_block1_1_conv (Conv2D)   (None, 56, 56, 64)   4096        ['conv2_bloc
k1_preact_relu[0][0]'
]

conv2_block1_1_bn (BatchNormal  (None, 56, 56, 64)  256         ['conv2_bloc
k1_1_conv[0][0]']
ization)


==================================================================================
```

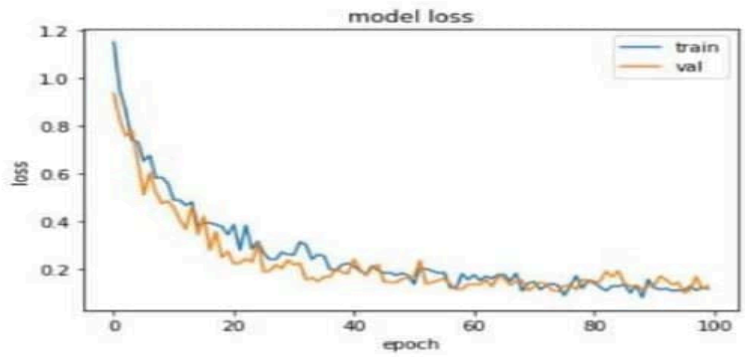


Complete Resent Model blocks in appendix chapter 10.

4.4 RESENT101V2

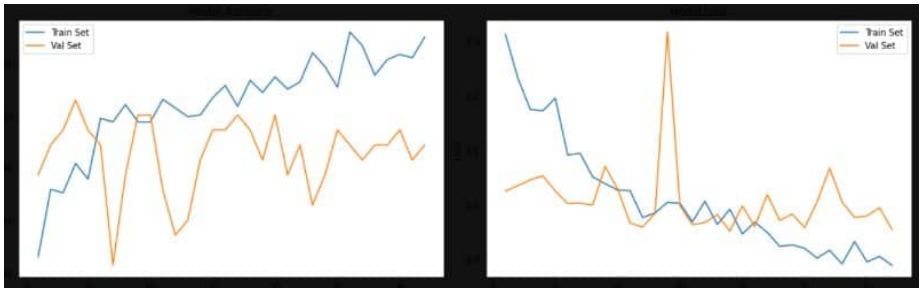
```
The accuracy of the model RESNET 101V2 is = 98.83720874786377 %

Confusion Matrix
[[82  0  7 54]
 [18  2  6 46]
 [ 1  0 64  2]
 [ 6  0  0 99]]
Classification Report
                         precision    recall  f1-score   support

          adenocarcinoma       0.77      0.57      0.66       143
   large.cell.carcinoma       1.00      0.03      0.05        72
                 normal       0.83      0.96      0.89        67
squamous.cell.carcinoma       0.49      0.94      0.65       105

               accuracy                           0.64       387
              macro avg       0.77      0.62      0.56       387
           weighted avg       0.75      0.64      0.58       387
```





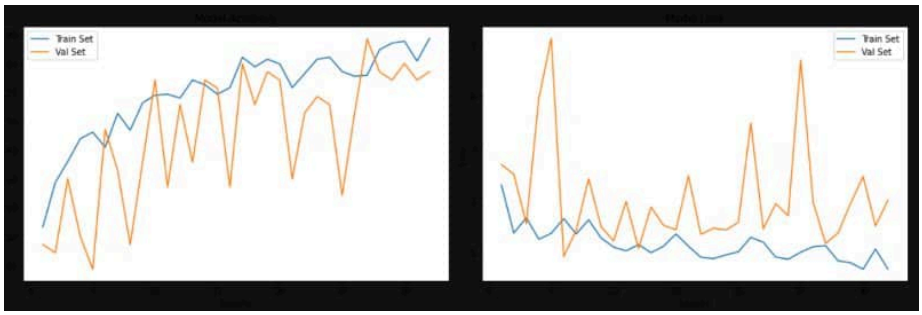
4.5 CNN MODEL:

The accuracy of the model is = 52.3809552192688 %



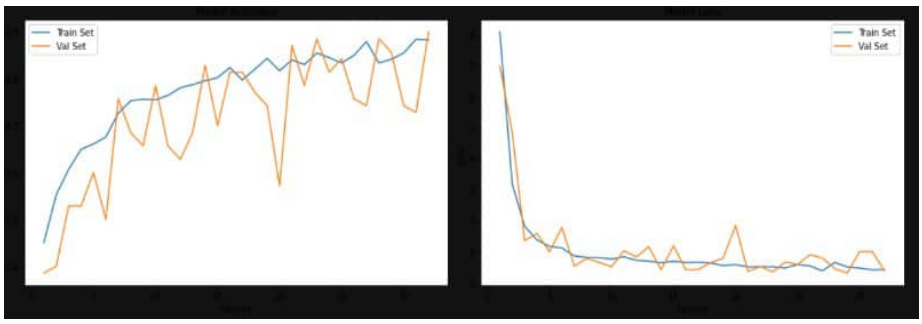
4.6 VGG16 MODEL:

The accuracy of the model is = 80.3174614906311 %



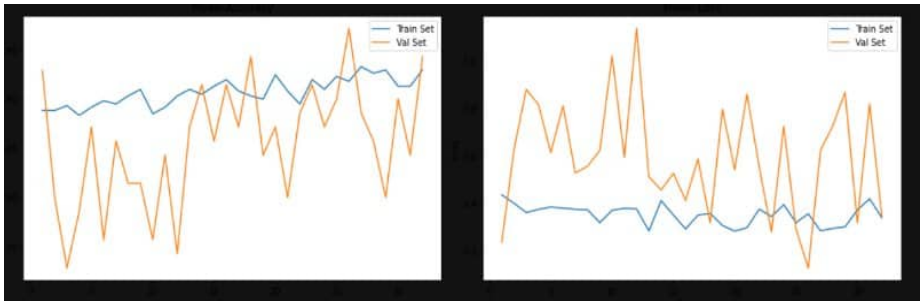
4.7 INCEPTIONV3 MODEL:

The accuracy of the model is = 73.96825551986694 %



4.8 REST NET50 MODEL:

The accuracy of the model is = 84.44444537162781 %



5 CHAPTER FIVE: SYSTEM REQUIREMENTS

5.1 CHALLENGES

To produce an effective system, it must meet several requirements that will make the user and the market convinced of this product. Our system is facing the challenge of reality and cost reduction, our system is designed to be useful for any user and does not require a user with special skills to navigate in it which makes our system easy to use, availability is also required in our system, availability refers to the fact that the system is available for every user using any mobile phone that works With Android or IOS, therefore, we are challenged to make the system more active, easier and of course more useful.

5.2 KEY BENEFITS

Our system has some characteristics to face the challenges of reality and cost reduction, availability, and the easy-to-use it and its operation too. These characteristics are listed in the following:

- Enables the user to use the system anytime and anywhere.
- Supply the user with the information that will make it more attractive and helpful for him.
- Solve the problem of wasting time by uploading the photo and viewing the analyzed results.
- Solving the problems of time losing by knowing any information about the determined branch.
- Easy to control for all users.

5.3 SOFTWARE FEASIBILITY

5.3.1 Economic Feasibility

The project is economically as the only cost involved is having a computer with the minimum requirements mentioned earlier. For the users to access the application, the only cost involved will be in getting access to the internet.

5.3.2 Technical Feasibility

- To deploy the application, the only technical aspects needed are mentioned below:
- Operating environment windows 10
- SQL workbench
- Database SQL Server
- Visual Studio Code
- Git
- Postman
- Anaconda
- Android Studio
- Heroku CLI

For Users:

- Internet Browser
- Internet Connection
- Mobile (Android or IOS)

5.3.3 Behavioral Feasibility

The application no requires special technical guidance, and all the views are available in the application are self-explanatory. the users are well guided with warning and failure methods for all action taken.

6 CHAPTER SIX: SYSTEM ANALYSIS

6.1 OVERVIEW

Our system consists of three parts; Each part is a different service for different users. The first part is for the administrator, the second part is for physicians, and the third part is for patients, and this chapter contains the steps of system analysis. Section 3.2 explains, objectives, and functionality of functions, and Section 3.3, diagrams, explains the functionality of the application.

6.2 GOALS

Our system has many goals to accomplish here we will explain in detail:

- Giving the ability to search about any post.
- Giving the ability to view the trending posts.
- Giving the ability to know more information about the post user selected.
- giving the ability to interact with post through a comment
- Giving the ability to view information about the author of the post user selected.
- Giving the ability to send a comment about anything that related to our website or application.

- Giving the ability to follow the latest news of the application through user email.
- Giving the ability to view branches of the company through a map.
- Giving the ability to share it with your friends on social media.

6.3 DIAGRAMS

6.3.1 Use Case

In system analysis, one of the graphical representations of data movement through the system is the use case diagram which provides a story of how a system, and its actors, will be utilized to achieve a specific goal (see figure 6.3.1-1).

The main components of use case diagram:

1. **Actors:** A role that a user plays with respect to the system, including human users and other systems.
2. **Use case:** A set of scenarios that describing an interaction between a user and a system, including alternatives,
3. **System boundary:** rectangle diagram representing the boundary between the actors and the system.
4. **Association:** communication between an actor and use case; represented by a solid line.

The figure below shows the ability of the patient, doctor, and admin. A patient can view trending posts, add comments, search a post, upload a photo, and can edit their profile and update password. Doctor who can add, edit, delete their own posts besides the ability to search a post, interact with any post with comment, upload photos. Admin controls everything in

the system which the developer can accept posts published from users, add post, update content of post, delete post, add user to system, delete user, and delete comments.

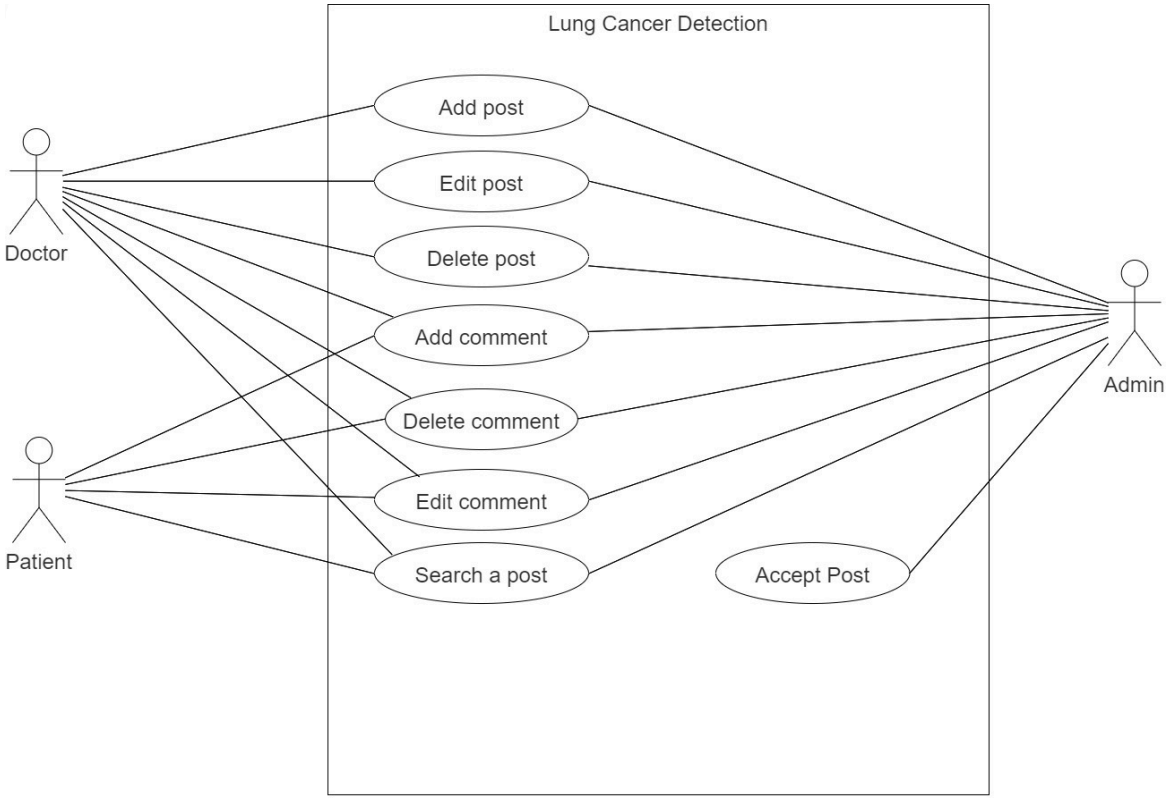


Figure 6.3.1-1 Use Case

6.3.2 Data Flow Diagram Logical

6.3.2.1 Context Diagram

This diagram shows our system in one process 'Access' when a doctor enters data like 'search for any post he wanted, publish a post, and so on' then if the post has the admin approval display the result on mobile application, and give the ability for the logged user to interact with comments. The actor of the system is patient, admin, and doctor.

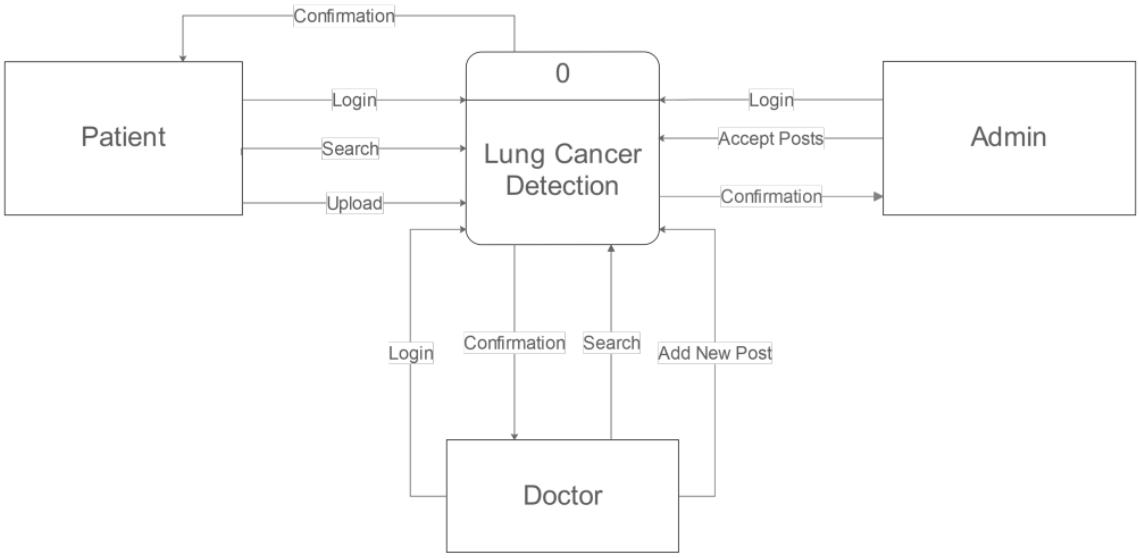


Figure 6.3.2-1 Context Diagram

6.3.2.2 Data Flow Diagram level 0

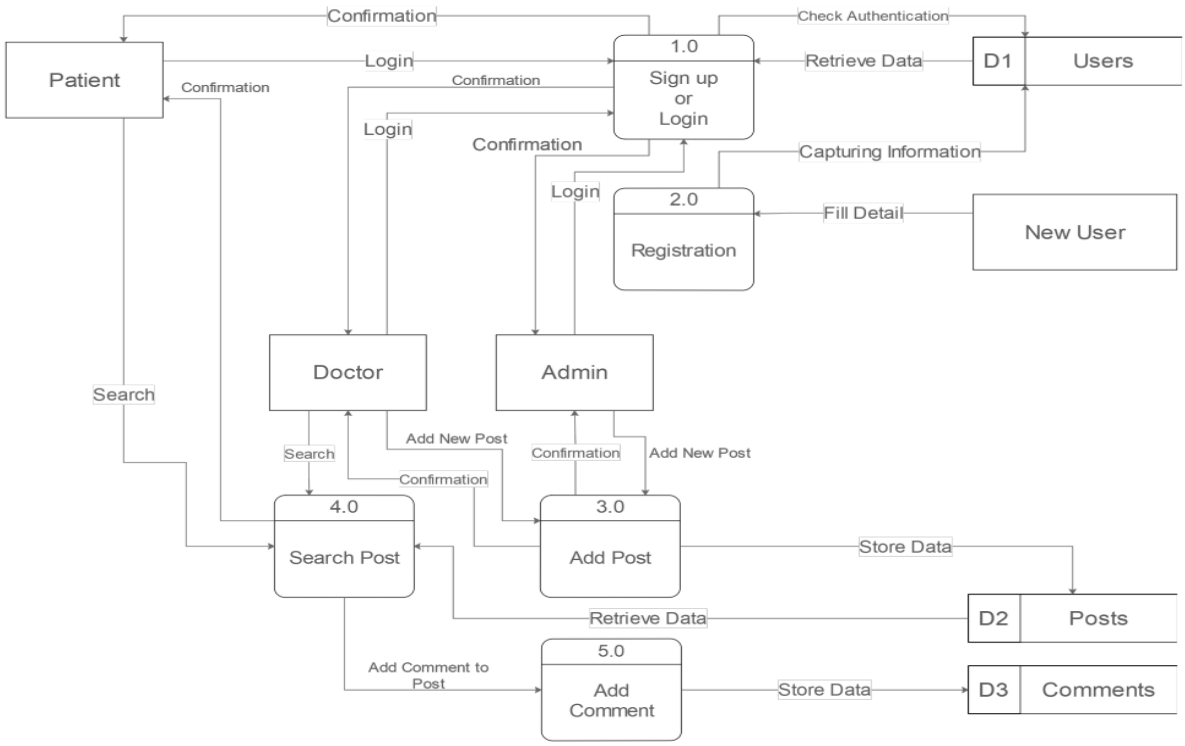


Figure 6.3.2-2 DFD Diagram level 0

6.3.2.3 Data Flow Diagram level 1

1. Add Post Process

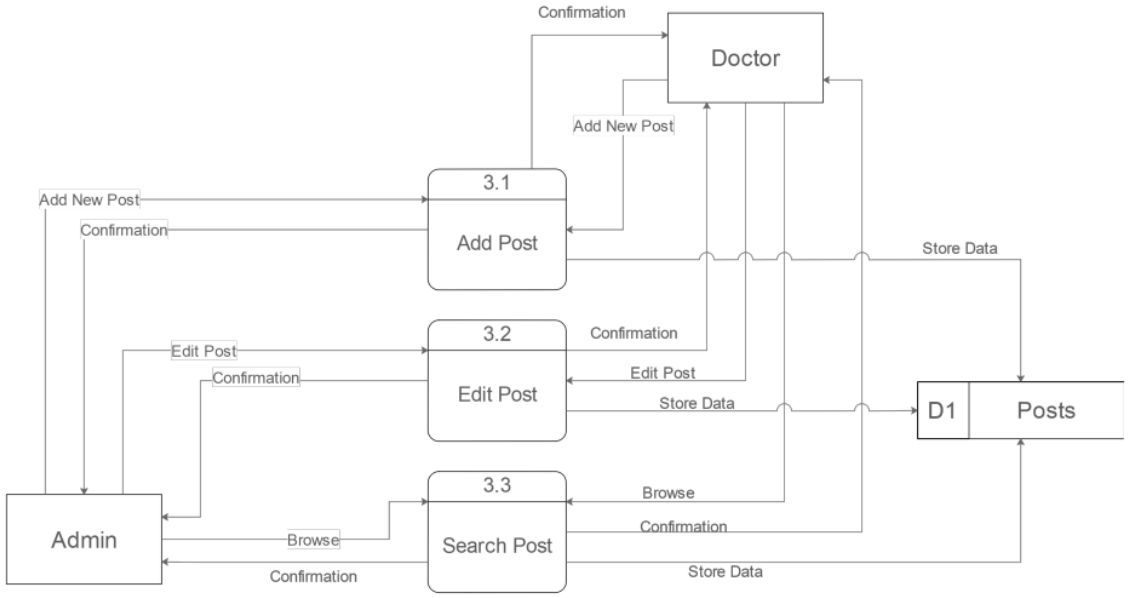


Figure 6.3.2-3 DFD Diagram level 1 for process 4

2. Add Comment Process

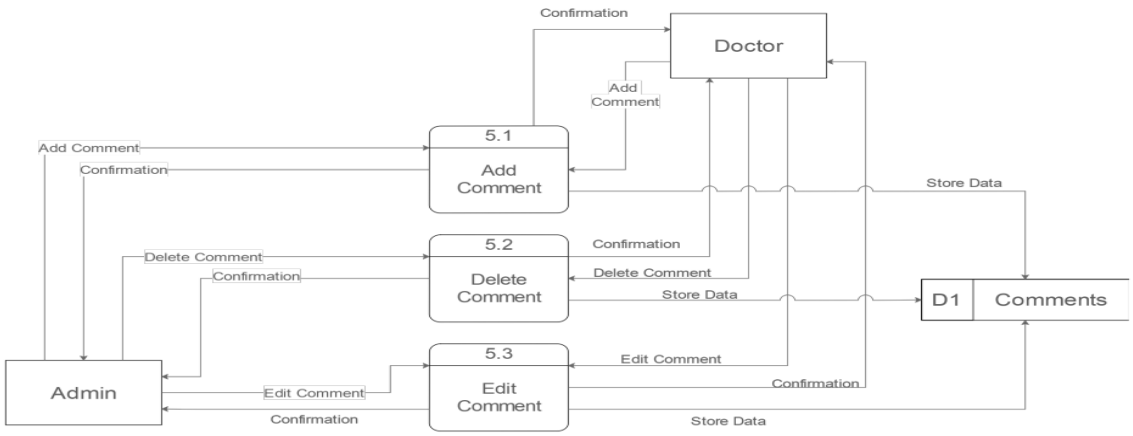


Figure 6.3.2-4 DFD Diagram level 1 for process 6

6.3.3 Entity Relationship Diagram Logical (ERD)

An entity-relationship diagram is a data modeling technique that creates a graphical representation of the entities, and the relationship between entities, within an information system.

The three main components of an ERD are:

1. **The entity** is a person, object, place, or event for which data is collected. For example, if you considered the information system for a business, entities would include not only the doctor, but the patients' phone, and comments as well. The entity is represented by a rectangle and labeled with a singular noun.
2. **The relationship** is the interaction between the entities. In the example above, the patient uploads a photo, so the word "uploads" define the relationship between that instance of a patient and the photo they uploaded. a relationship may represent by a diamond shape, or more simply, by the line connecting the entities. in either case, verbs are used to label the relationships.
3. **The cardinality** defines the relationship between entities in terms of numbers. An entity may be optional: for example, a post could have no comment or could have one or many comments, or mandatory: for example, there must at least one post listed to be approved. There are several different types of cardinality notation; crow's foot notation, used there, is a common one. In crow's foot notation, a single bar indicates one, a double bar indicates one and only one (for example, a single comment has only one user), a circle indicates zero, and a crow's foot indicates many. The three main cardinal relationships

are: one-to-one, expressed as 1:1; one-to-many, expressed as 1: M; and many-to-many, expressed as M: N.

The steps involved in creating an ERD are:

1. Identify the entities.
2. Determined all significant interactions.
3. Analyze the nature of interactions.
4. Draw the ERD.

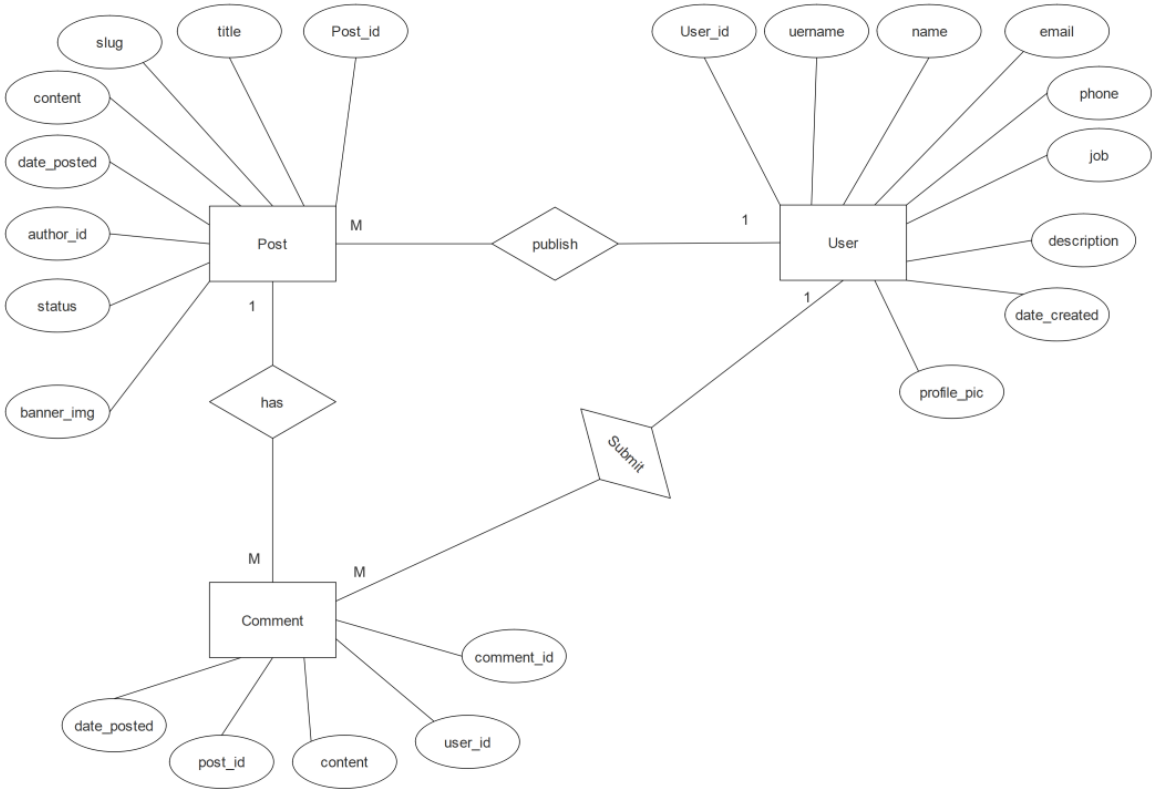


Figure 6.3.3-1 ER Diagram

6.3.4 ERD Mapping

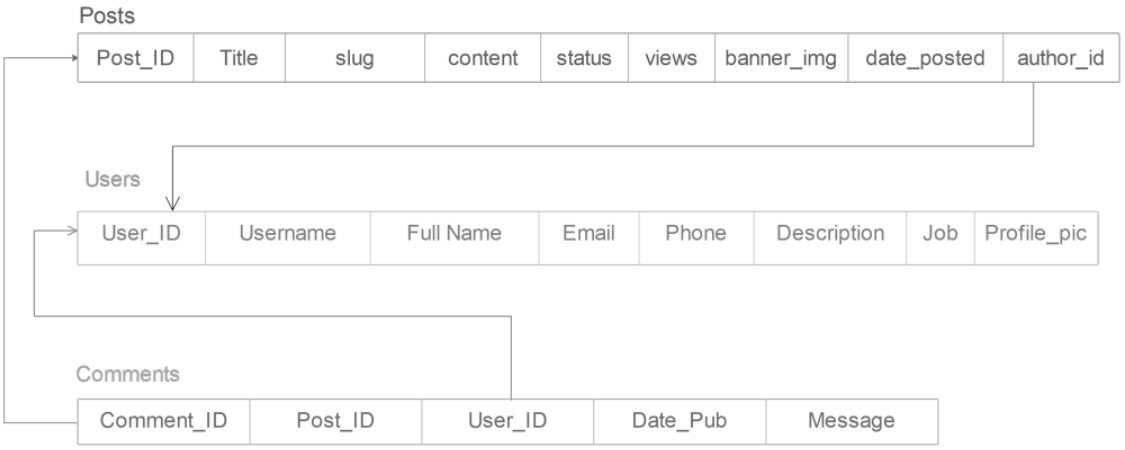


Figure 6.3.4-1 ERD Mapping Diagram

6.3.5 Activity Diagram

An activity diagram offers a dynamic view of the system and is like flowchart; they are useful for describing workflow business processes or the procedural logic of the system.

The main components of activity diagram:

1. **Action:** a task to be performed; represented by a rectangle with rounded corners, in the activity diagram, each activity consists of a series of actions.
2. **Control flows:** actions are connected by arrows, and they show the direction of the flow in the diagram.
3. **Decision nodes**: decision points represent a test condition to ensure that the control flow or object flow only goes down one path
4. **Fork Node**: Split behavior into a set of parallel or concurrent flows of activities (or actions).

5. **Join Node**: Bring back together a set of parallel or concurrent flows of activities (or actions).

The activity diagram starts with **Initial Node**; Portrays the beginning of a set of actions or activities and ends with **Activity Final Node**; Stop all control flows and object flows in an activity (or action).

1. **Patient Activity**

The figure 6.3.5-1 shows the allowed activities for patients: view all posts, search for specific posts, interact with comments and upload photos for view results if it is cancerous or not.

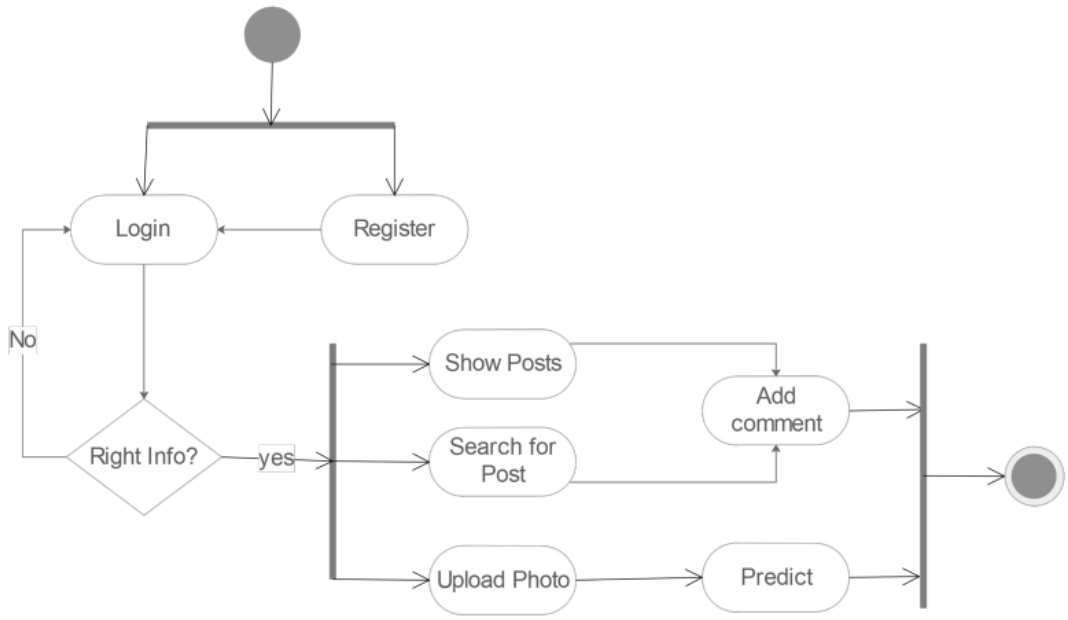


Figure 6.3.5-2 Activity Diagram for patient

2. **Doctor activity**

The figure 6.3.5-2 shows the allowed activities for doctors: view all posts, search for specific posts, interact with comments, publish posts to the website after the admin approval and upload photos for view results if it is cancerous or not.

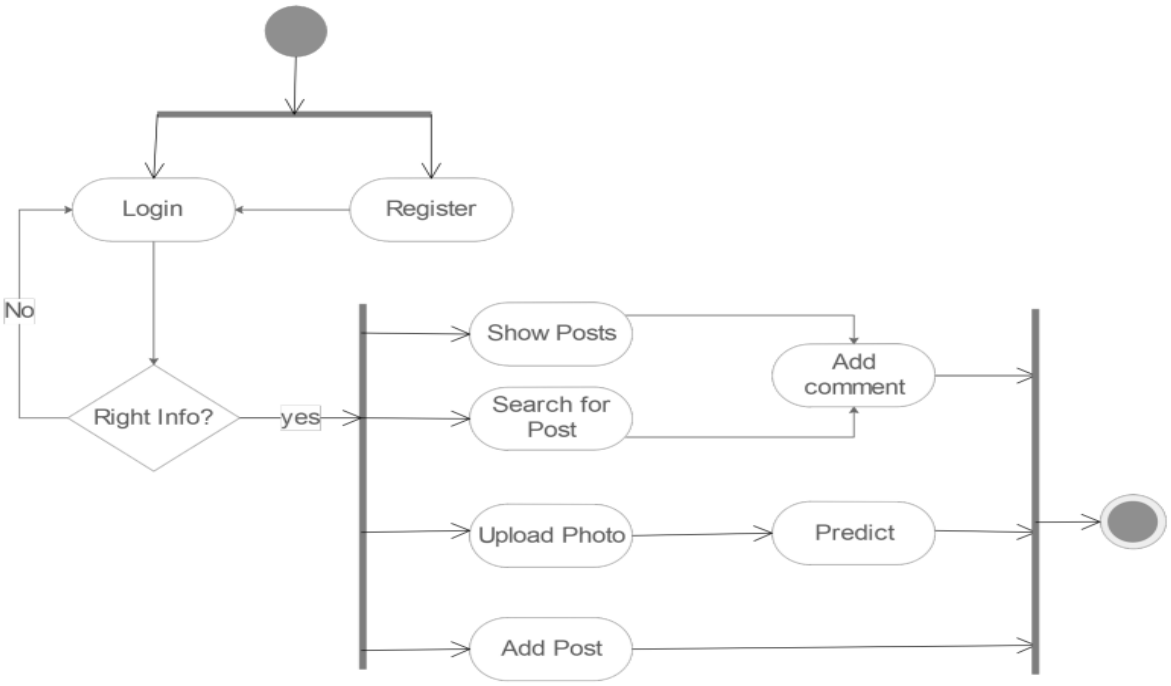


Figure 6.3.5-3 Activity Diagram for doctor

3. admin activity

The figure 6.3.5-3 shows the allowed activity for admin: add user, delete user, accept the pending posts and delete inappropriate comments.

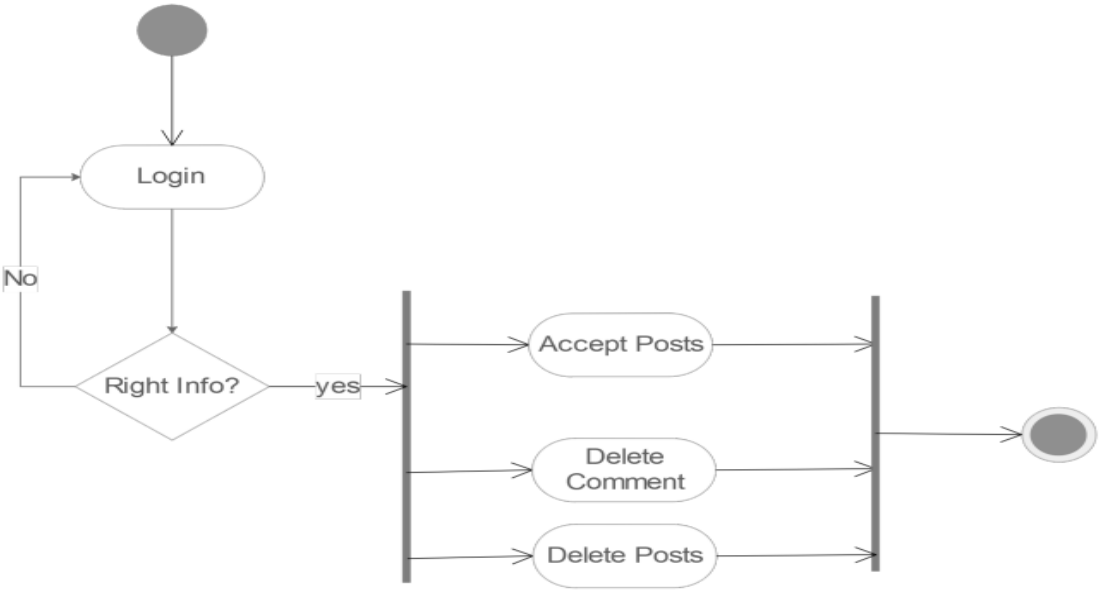


Figure 6.3.5-4 Activity Diagram for admin

7 CHAPTER FIVE: SYSTEM DESIGN

7.1 DESIGN GOALS

The design of the web application involves the design of the forms for listing the posts, display the complete documentation of the post, and upload image section.

Design of interactive application that enables the user to show the product based on the search term that user entered.

Design of an application that has feature like drag and drop

Design of application that decreases data transfers between the client and the server.

7.2 ARCHITECTURAL DESIGN

7.2.1 Architecture Context Diagram

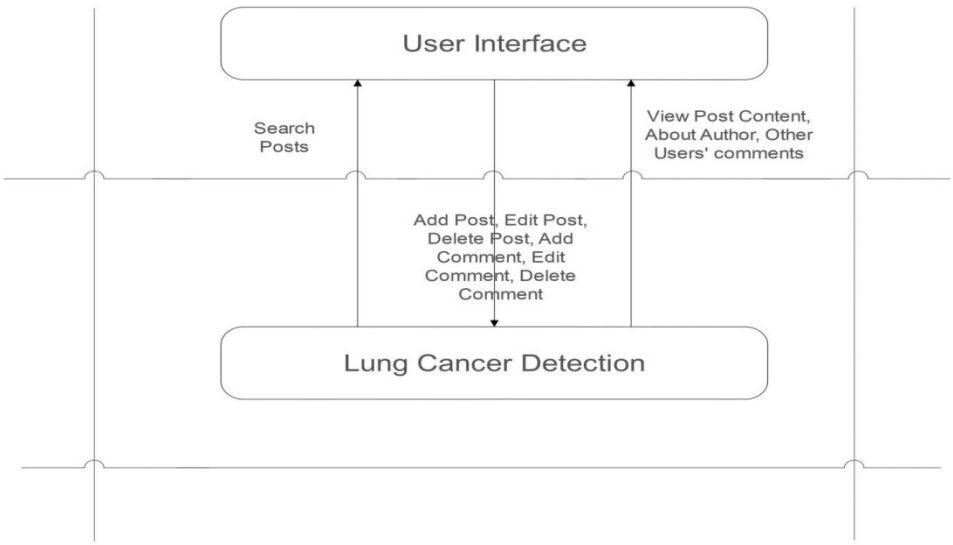


Figure 7.2.1-1 Architecture Context Diagram

7.2.2 Description of Architectural Design

In this context diagram, the information provided to and received from the "Lung Cancer Detection" is identified. The arrows represent the information received or generated by the application. The closed boxes represent the set of sources and set of information. In the system, we can observe that the user interacts with the application through a graphical user interface. The inputs to the system are the search criteria provided by the user and grid views which present the users with a list of acceptable posts. The users can view complete specifications, view comments by other users.

7.3 PROCEDURAL/MODULAR APPROACH

7.3.1 Post List Module

This module starts when the user visits the home page or when a user searches for a post by entering a search term. This part of the application includes displaying all the posts that match the searched term entered by the user. The user browses through the posts and each post contain a header image, the main content of the post, headings and sub images, date posted, the author of this article, and comment section.

7.3.2 Post Description Module

This module starts when a user visits the post description page. a user can view a parallax header image and the complete specification of the post like headings and figures that make post easier. Finally, there is About Author that gives info about the creator of post and comment section from other users.

7.3.3 Upload Image Module

This module starts when the user log in. This part of application has container where user can upload photo through browsing or drag and drop in the specified area. After uploading, user can view the result of analysis the photo that determined if there is cancer or not.

8 CHAPTER EIGHT: METHODOLOGY

8.1 BACK-END AND API

8.1.1 Libraries

8.1.1.1 Flask

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries.It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Flask depends on the Jinja template engine and the Werkzeug WSGI toolkit.

8.1.1.2 Jinja

Jinja is a fast templating engine. Special placeholders in the template allow writing code similar to Python syntax. Then the template is passed data to render the final document.

8.1.1.3 Flask SQL-alchemy

Flask-SQLAlchemy is an extension for Flask that adds support for SQLAlchemy to your application. It aims to simplify using SQLAlchemy with Flask by providing useful defaults and extra helpers that make it easier to accomplish common tasks.

8.1.1.4 Flask -Login

Flask-Login provides user session management for Flask. It handles the common tasks of logging in, logging out, and remembering your users' sessions over extended periods of time. It will:

- Store the active user's ID in the session, and let you log them in and out easily.
- Let you restrict views to logged-in (or logged-out) users.
- Handle the normally-tricky "remember me" functionality.
- Help protect your users' sessions from being stolen by cookie thieves.
- Possibly integrate with Flask-Principal or other authorization extensions later on.

However, it does not:

- Impose a particular database or other storage method on you. You are entirely in charge of how the user is loaded.
- Restrict you to using usernames and passwords, Open IDs, or any other method of authenticating.
- Handle permissions beyond "logged in or not."

8.1.1.5 Flask- Share

Create social share component in Jinja2 template based on share.js. enable to share content of your website easily.

8.1.1.6 Marshmallow

Marshmallow library for converting complex datatypes, such as objects, to and from native Python datatypes. In short, marshmallow schemas can be used to:

- **Validate** input data.
- **Deserialize** input data to app-level objects.
- **Serialize** app-level objects to primitive Python types. The serialized objects can then be rendered to standard formats such as JSON for use in an HTTP API.

It is integrate with flask through flask-marshmallow

8.1.1.7 Flask- JWT- extended

Features

Flask-JWT-Extended not only adds support for using JSON Web Tokens (JWT) to Flask for protecting routes, but also many helpful (and **optional**) features built in to make working with JSON Web Tokens easier. These include:

- Adding custom claims to JSON Web Tokens
- Custom claims validation on received tokens
- Refresh tokens
- First class support for fresh tokens for making sensitive changes.
- Token revoking/blacklisting

8.1.1.8 Flask -Migrate

Flask-Migrate is an extension that handles SQLAlchemy database migrations for Flask applications using Alembic. The database operations are provided as command-line arguments under the `flask db` command.

8.1.1.9 Sqlalchemy

SQLAlchemy is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL. It provides a full suite of well-known enterprise-level persistence patterns, designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language.

8.1.1.10 Flask CKEditor

Flask-CKEditor CKEditor integration for Flask, including image upload, code syntax highlighting, and more. Features: Integrate with Flask-WTF/WTForms. Configure CKEditor through Flask's configuration system. Image upload support. Code snippet highlighting.

8.1.1.11 MySQL connector python

This manual describes how to install and configure MySQL Connector/Python, a self-contained Python driver for communicating with MySQL servers, and how to use it to develop database applications.

8.2 FRONT-END

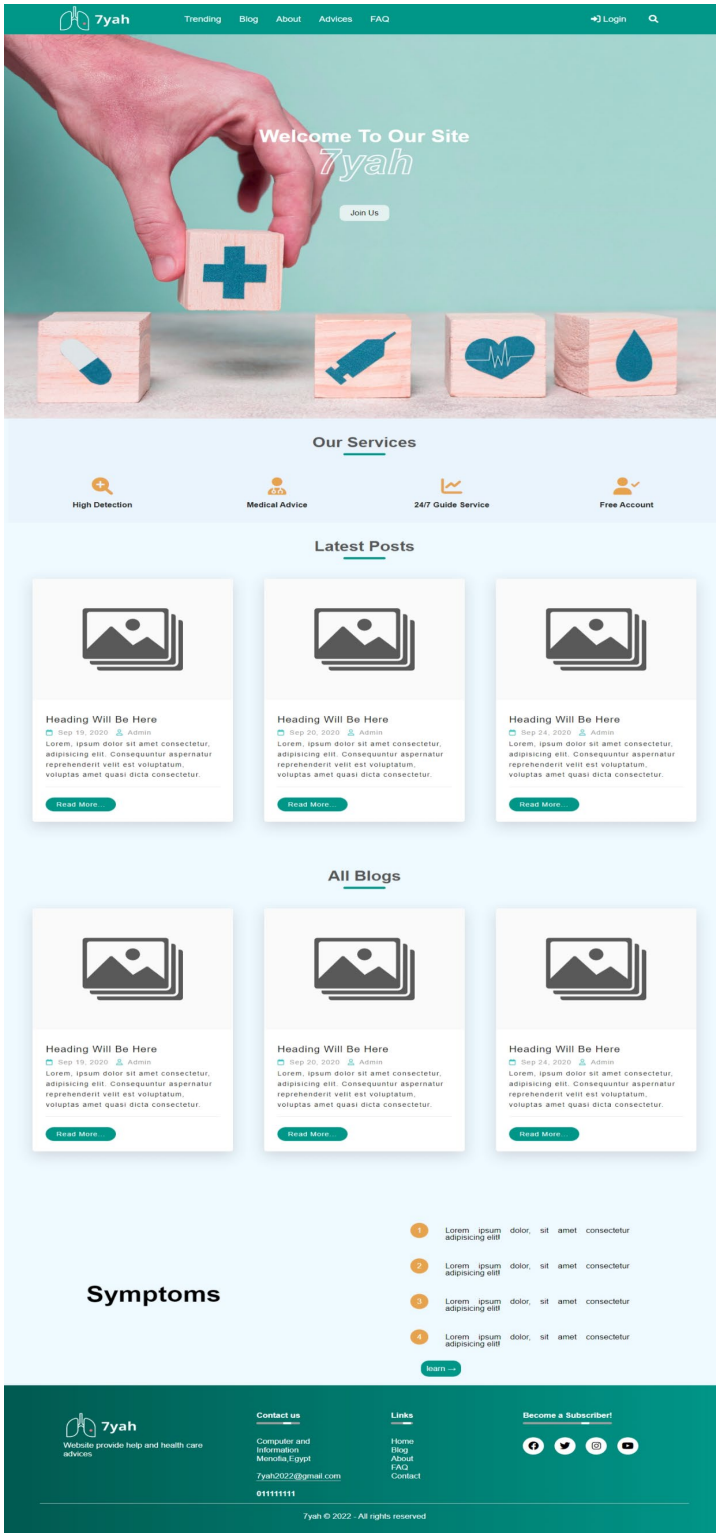
HTML Pages:

1. Home
2. About
3. FAQ
4. Advice
5. Login
6. Register
7. Predict
8. results
9. Search
10. Pending posts – for admin
11. Pending posts – for doctor
12. Manage Account – edit profile
13. Manage account – privacy
14. User profile
15. Add post
16. Edit post
17. Top posts
18. All posts
19. Show individual post
20. logout

8.2.1 Web Development

Home:

This is home page contain welcome land page , section of service that website serve, trending blogs, list of blogs ,and some advices that gives awareness to the guest about lung cancer.



Top Posts:

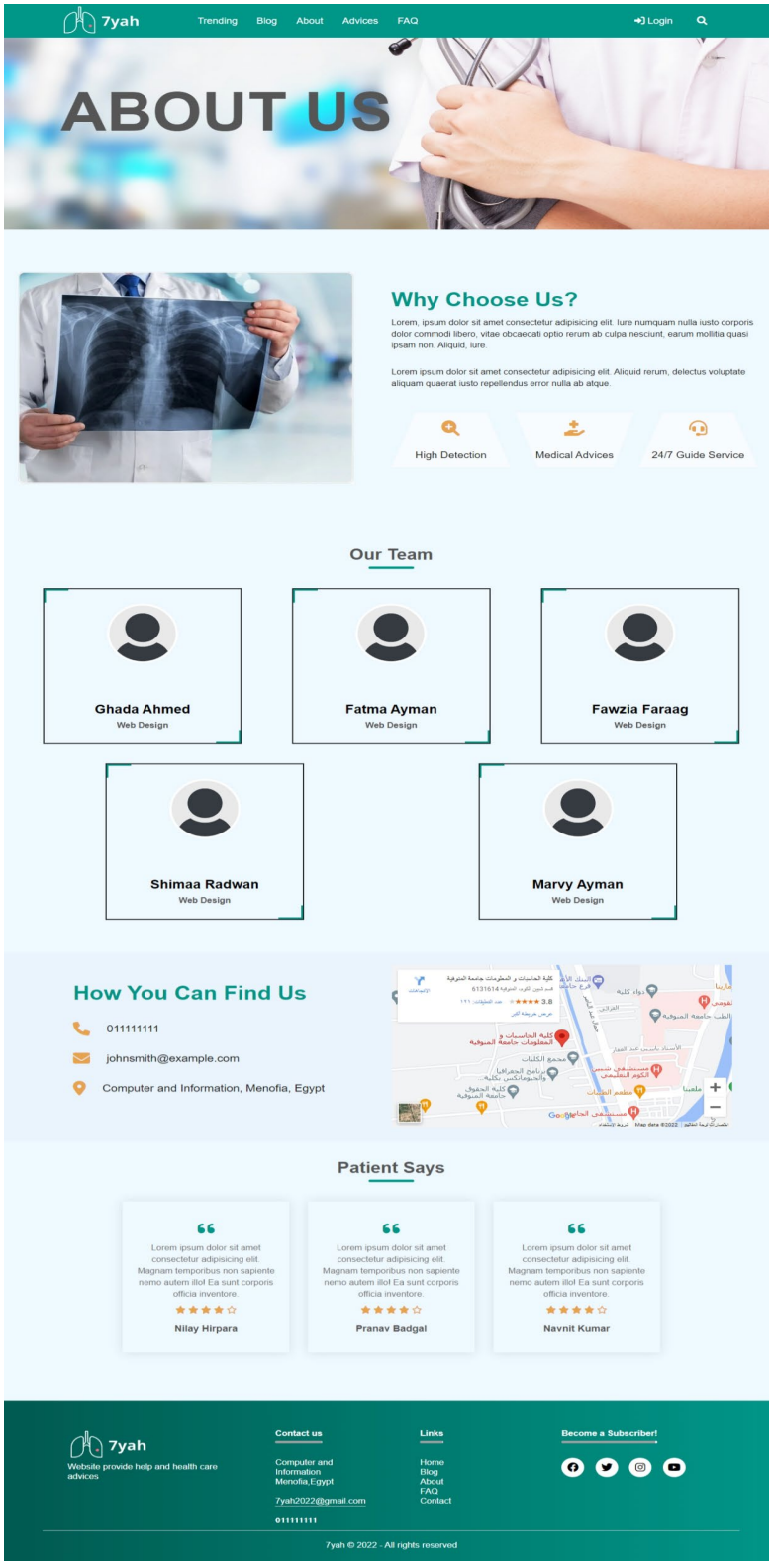
This page contains most popular posts that users care about.

Posts:

This page contains all the posts in the website.

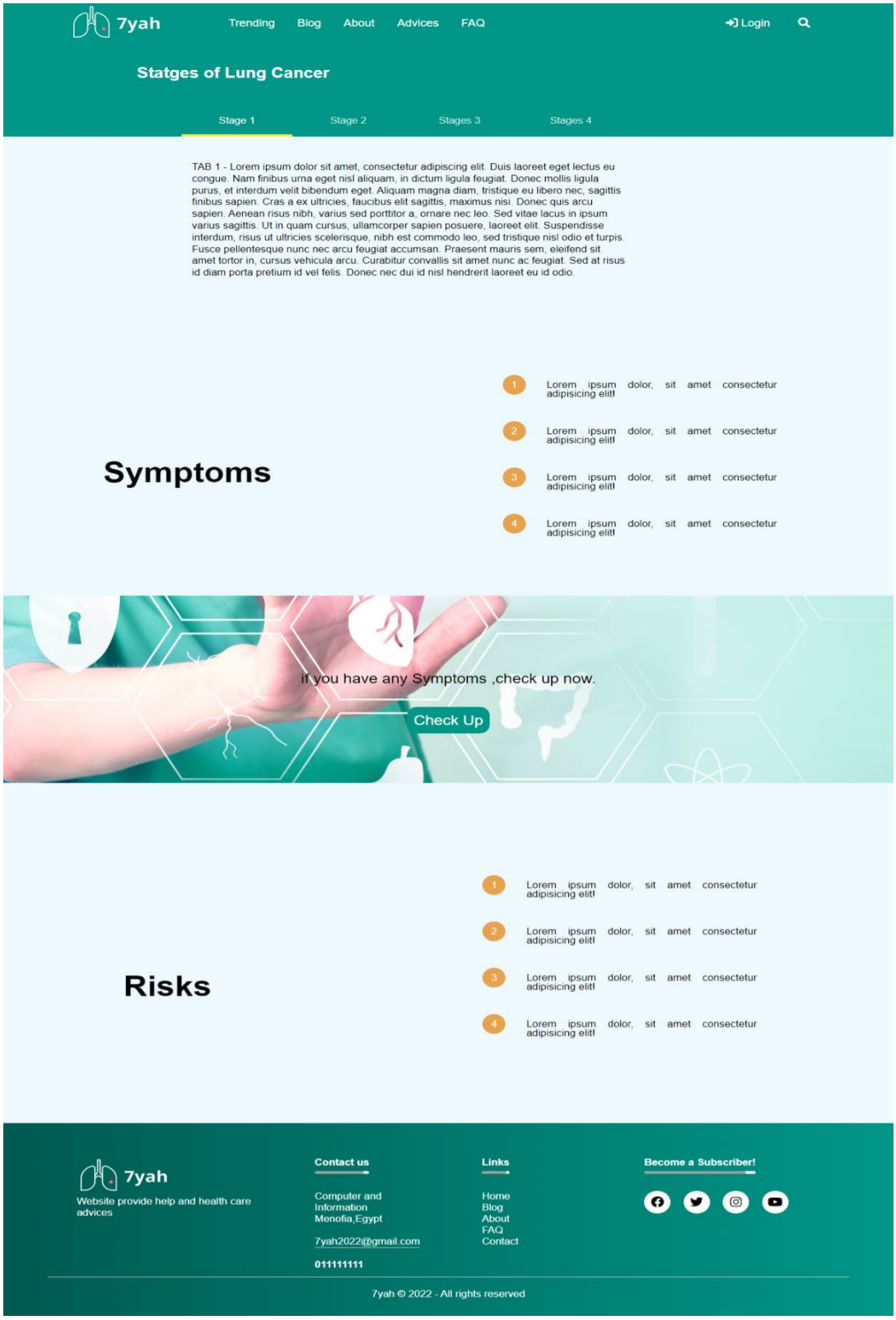
About:

This page contain information about the team and the purpose of publish websites besides containing a map that guide user to location of company and feedbacks from users about the website.



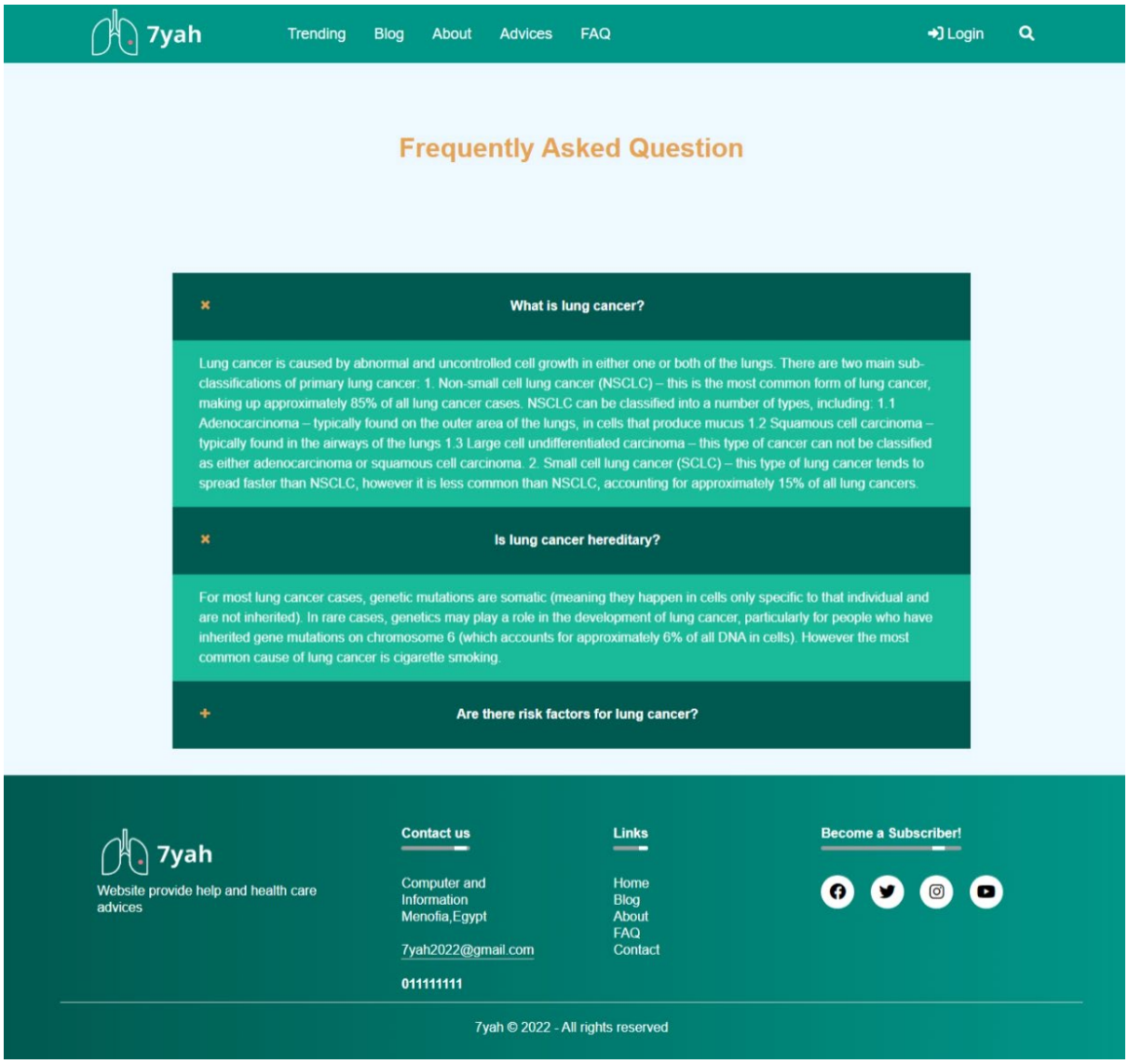
Advices:

This page contain most information that guest should be aware of.



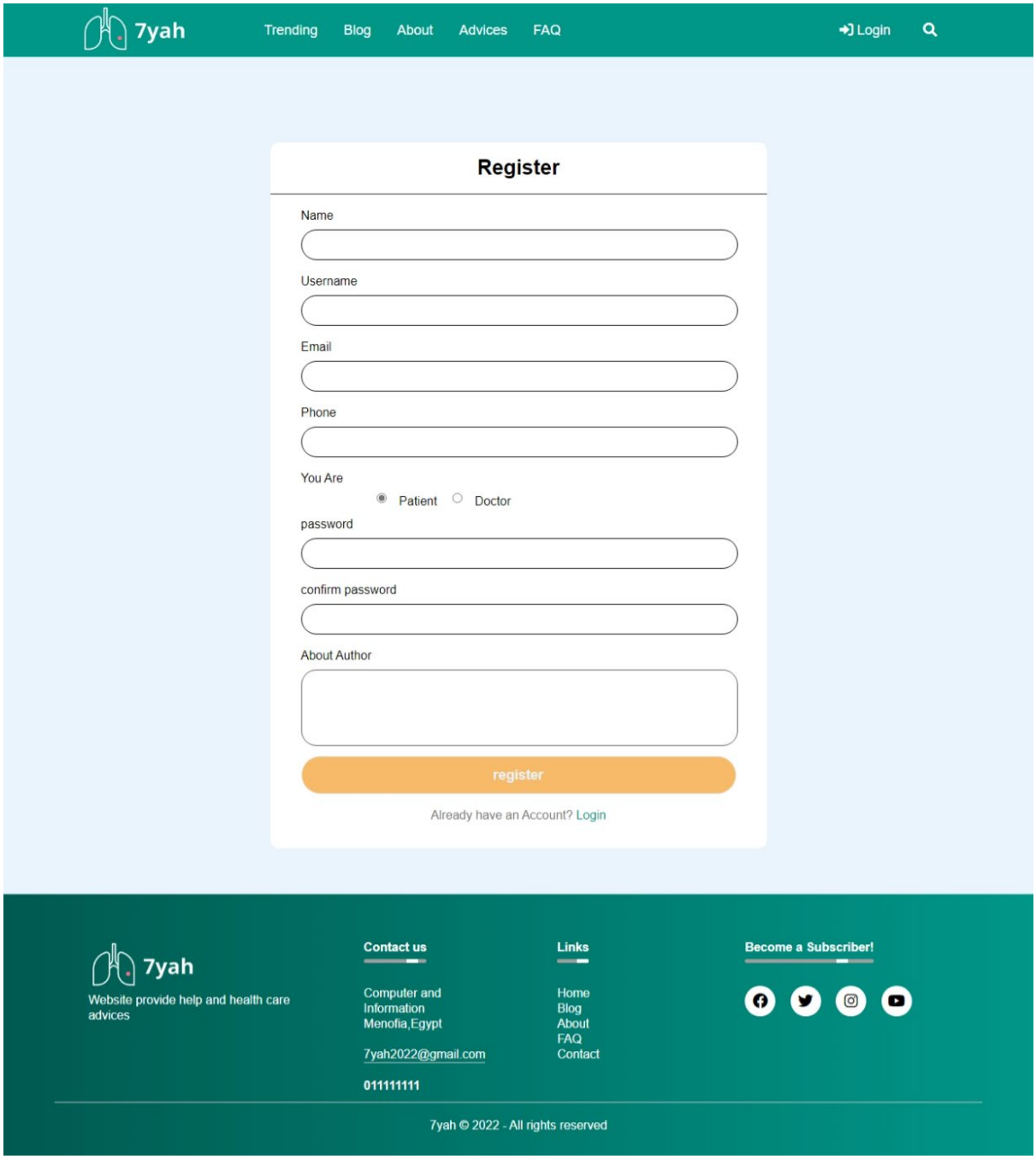
Frequently Asked Questions (FAQ):

This page contain most frequent question on the website and answers .



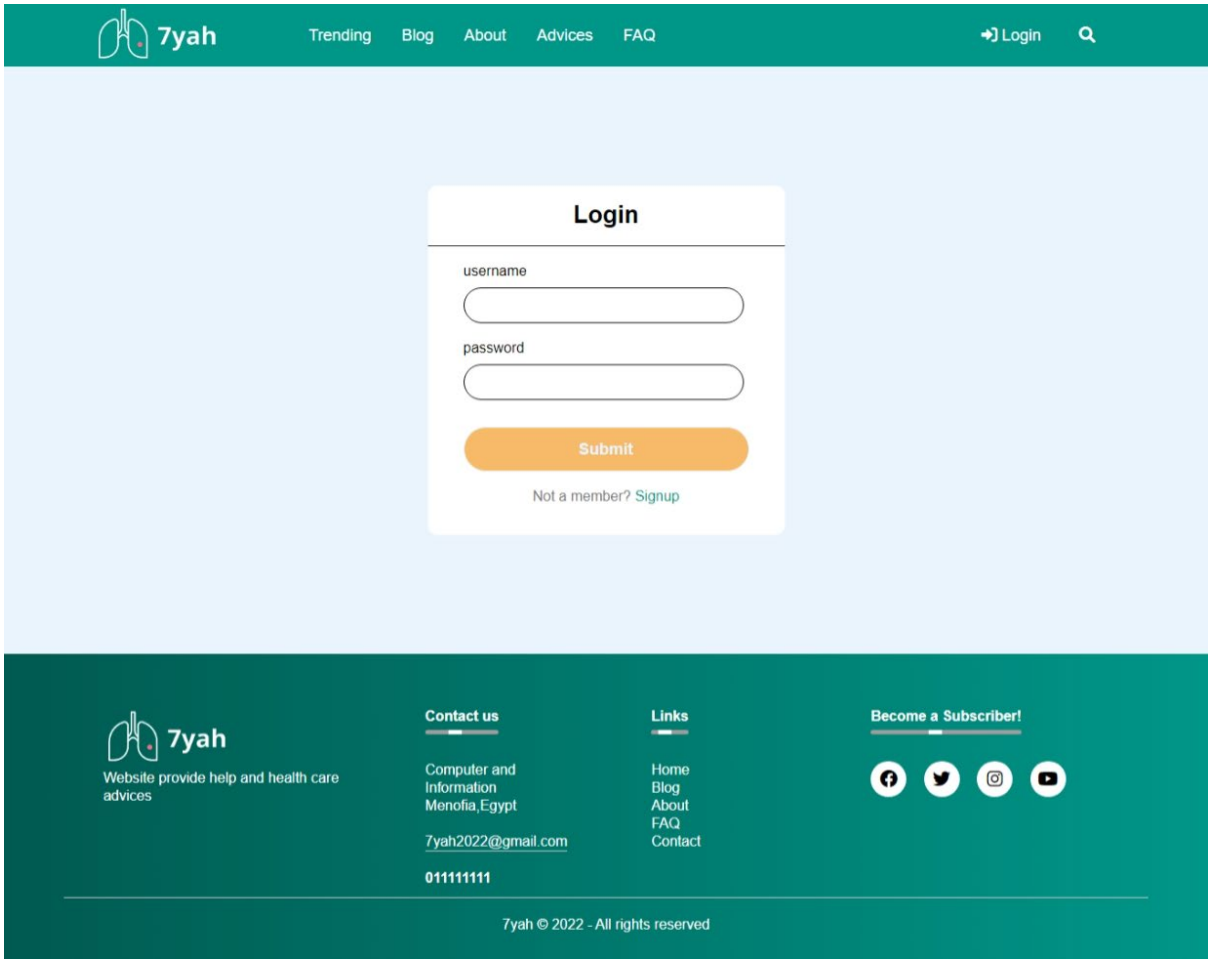
Register:

This page contain the form that user fill to give him access for more features.



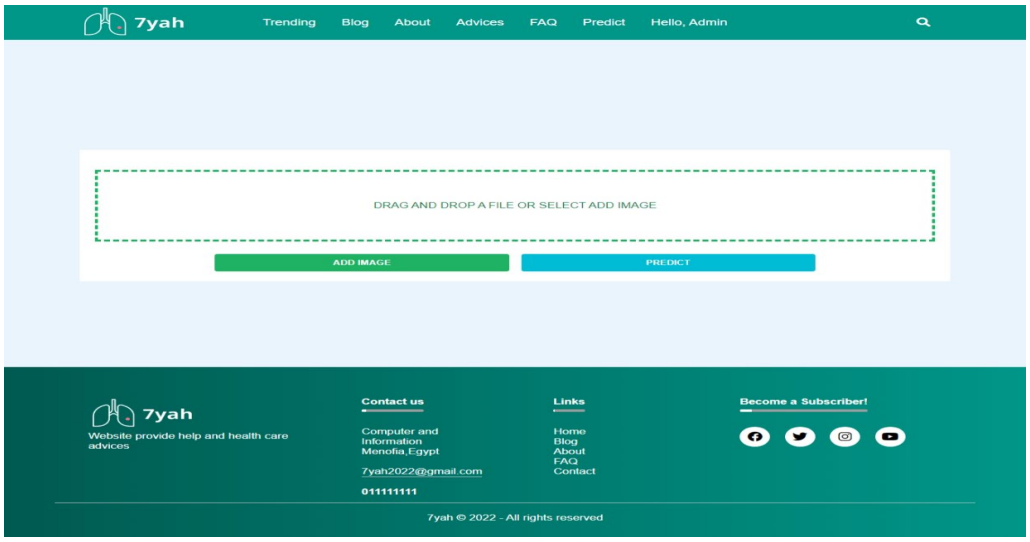
Login:

This page contains the form which gives access for logged users.



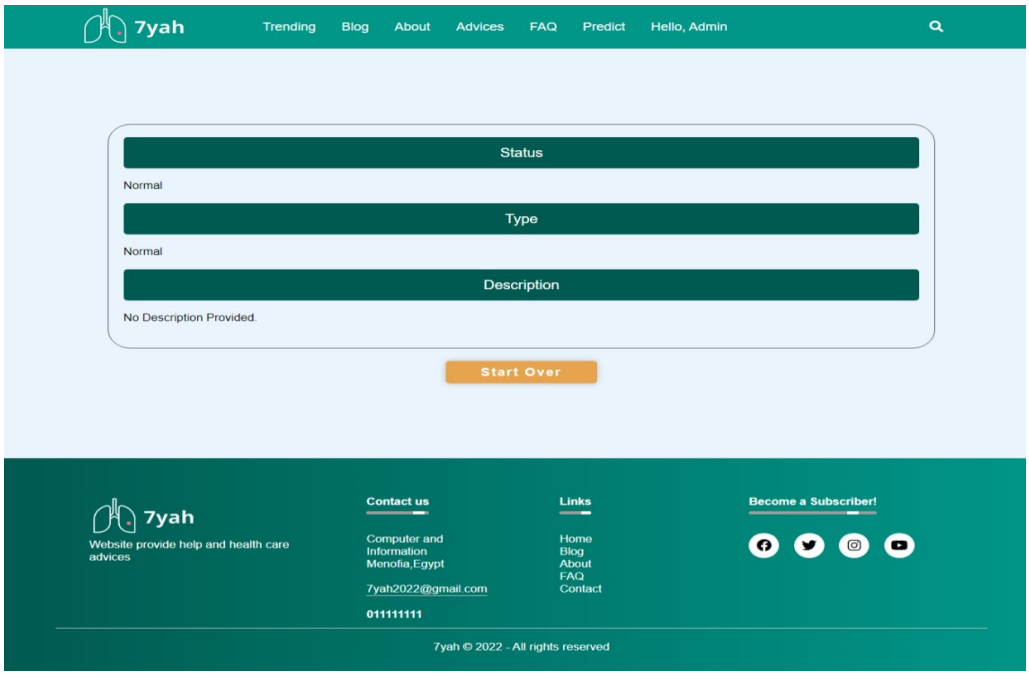
Predict:

This page contain uploader so that user can discover if he/she cancerous or not.



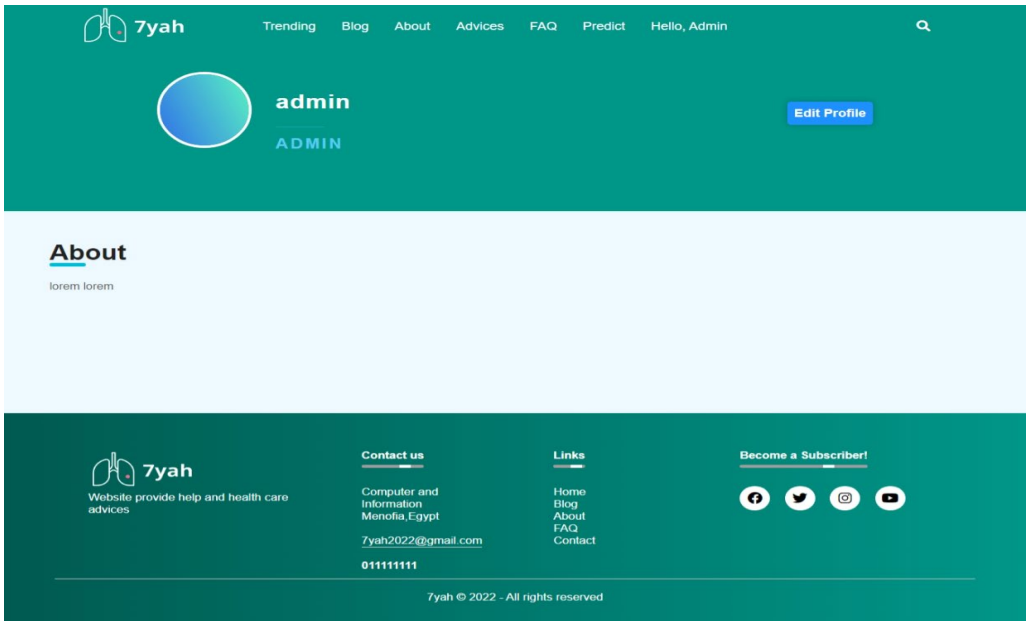
Result:

This page contain the result of analyze the image user uploaded.



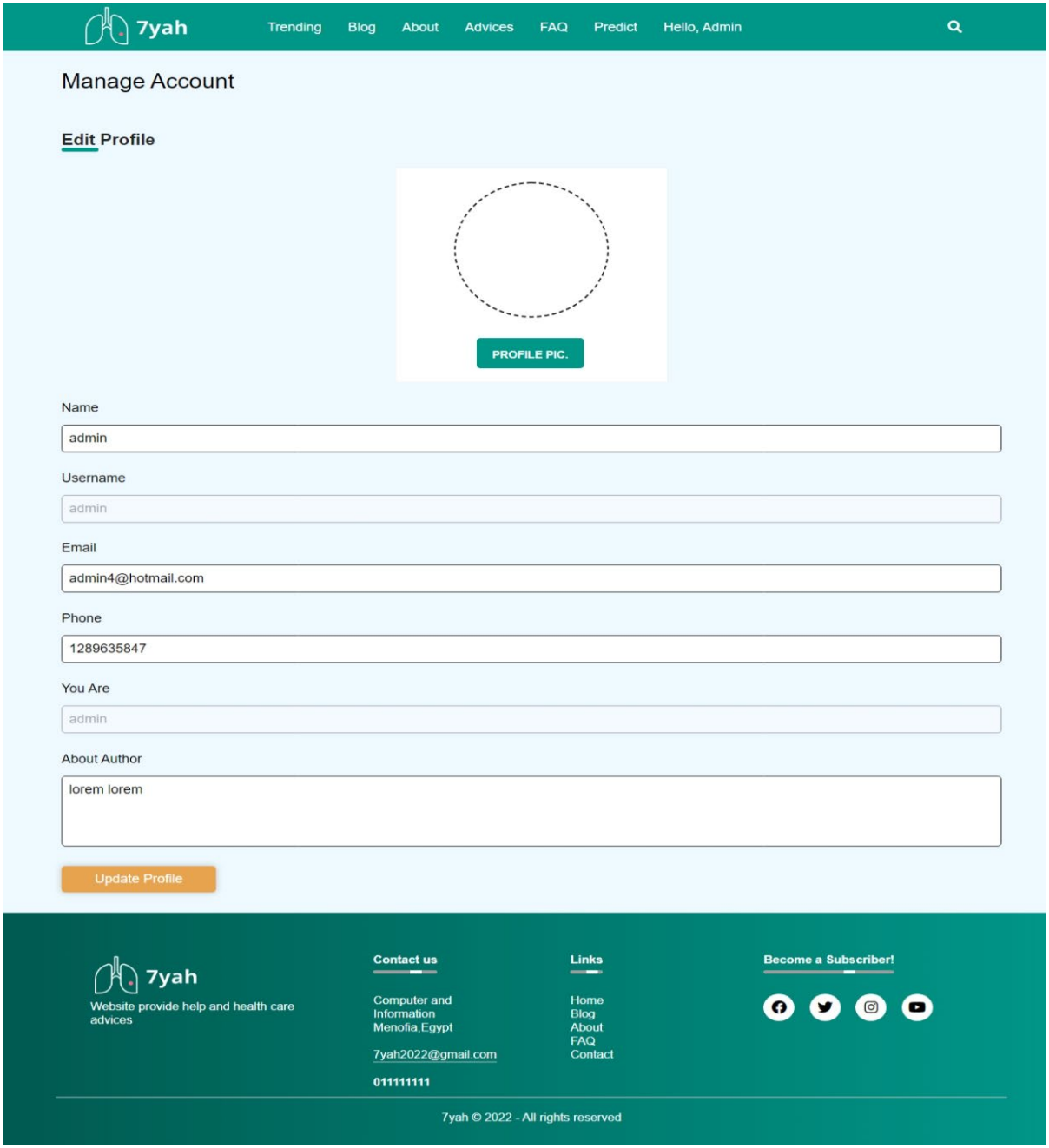
Profile:

This page contains user information and if its profile it gives him ability to edit his information.



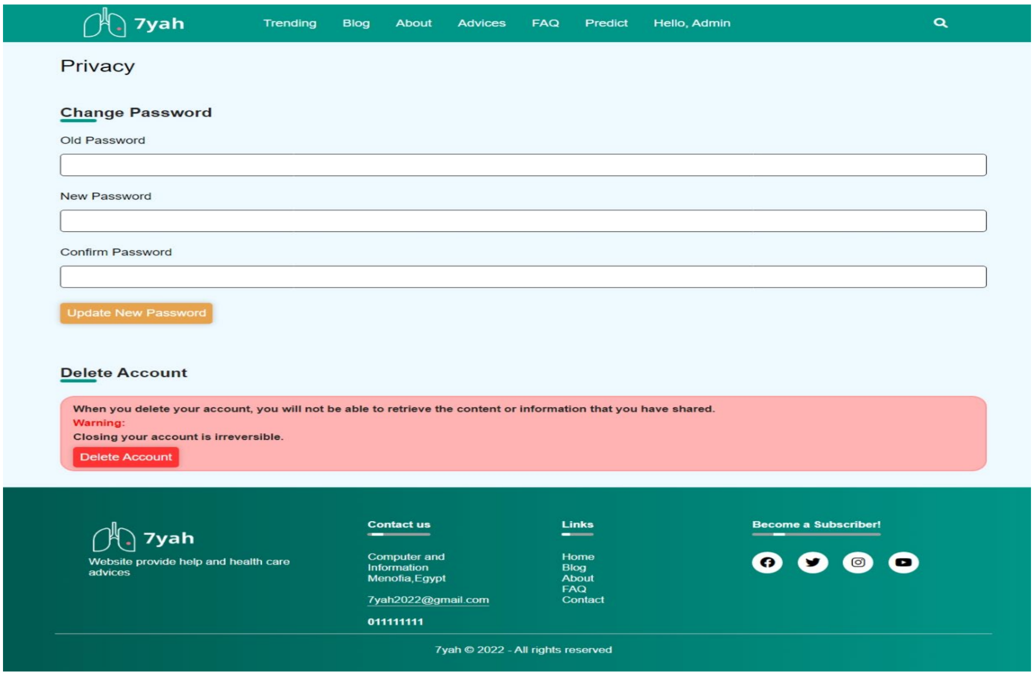
Manage Account – Edit Profile:

This page contain the fields to update his profile . he can also update a picture for his/her profile.



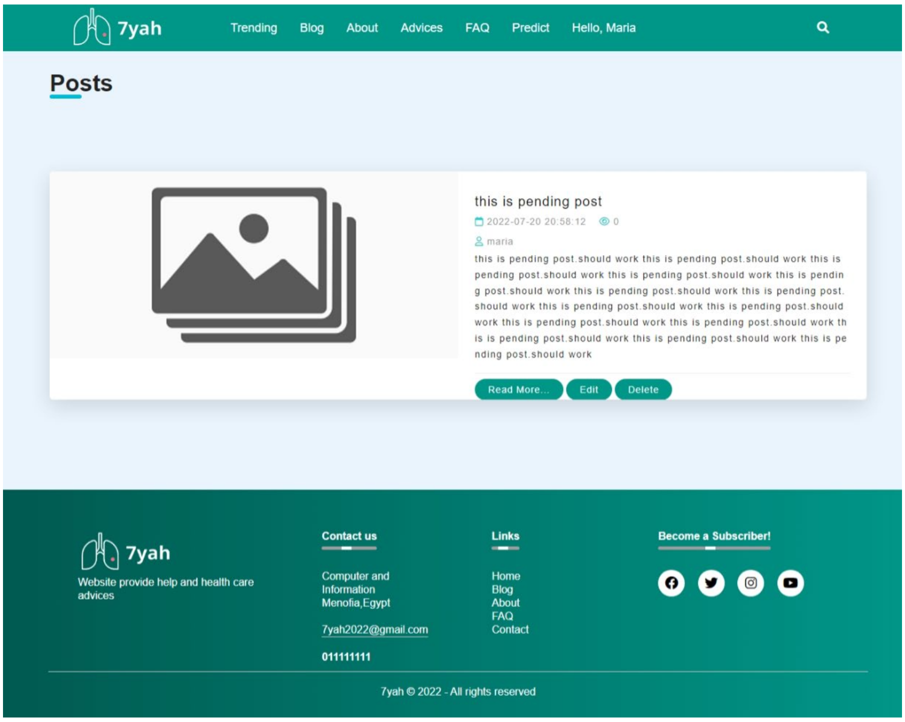
Manage Account -Privacy:

This page gives the ability to change password, delete account permanently.



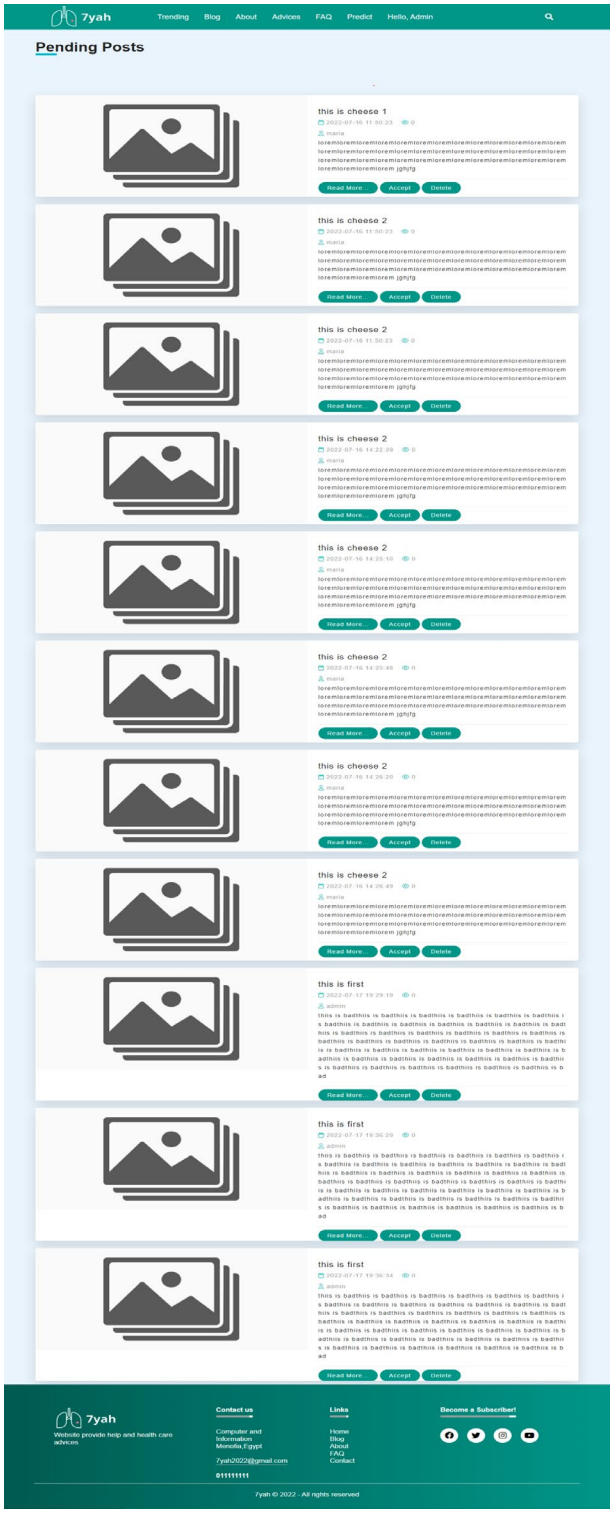
User Pending Posts:

This feature enables the doctor to see the status of his/her posts.



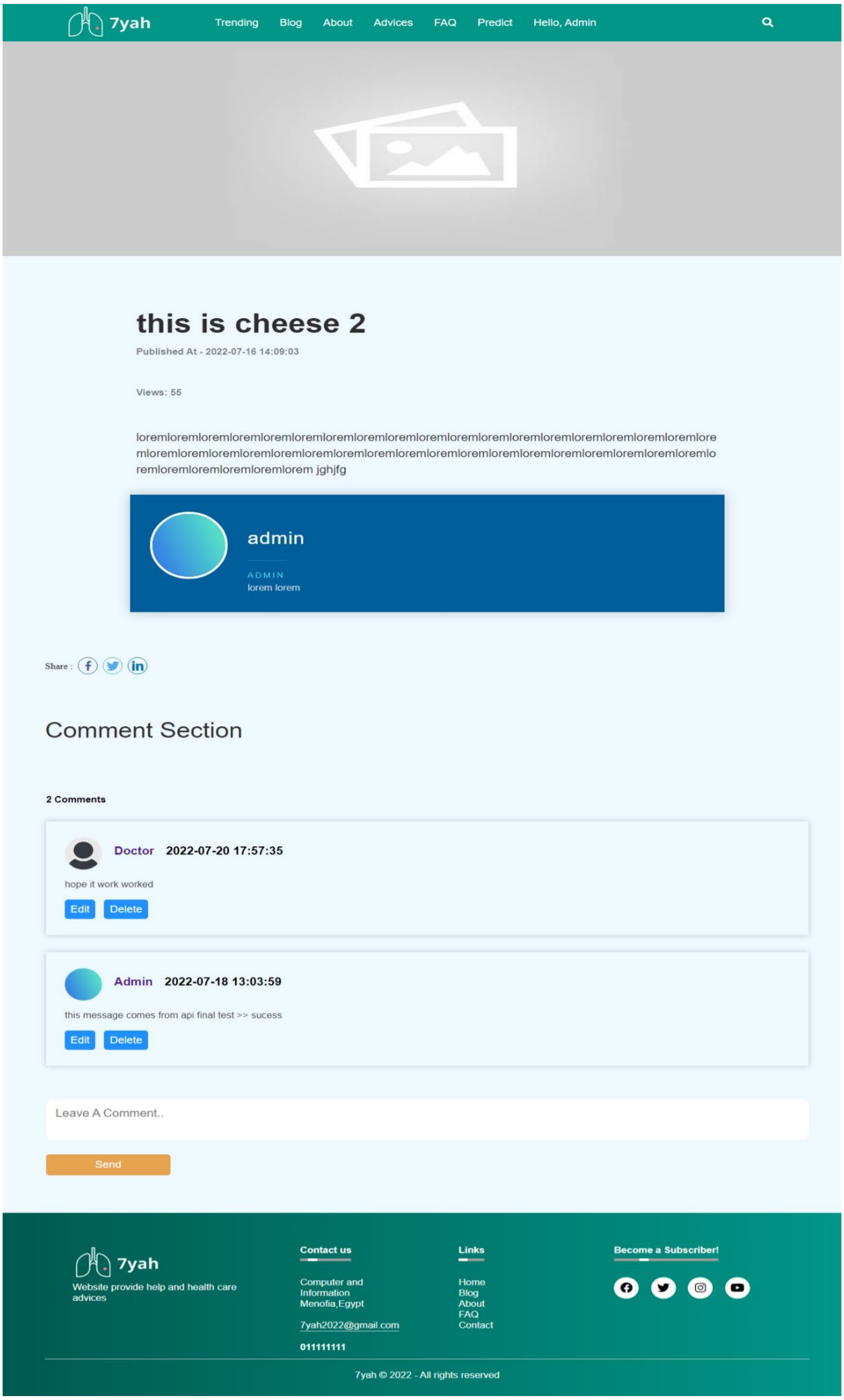
Pending Posts:

This page contains the pending posts that haven't published yet and waiting for acceptance from admin to make them appear in the blog website.



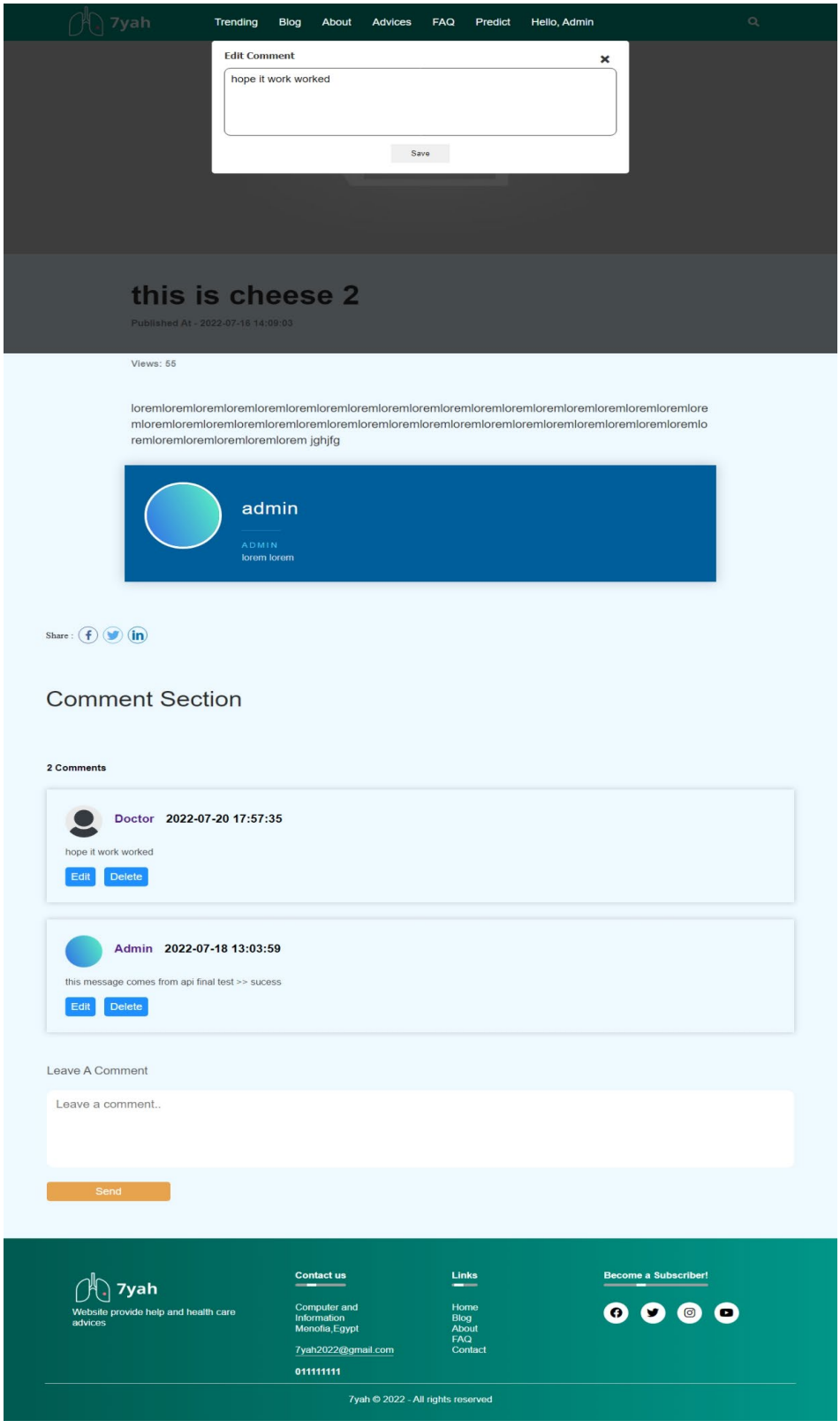
Show post:

This page contain the banner image (if exist) , blog title ,and content of post .it is also gives the logged user to interactive with the post by a comment ,edit, or delete . It is also have share post in social media section.



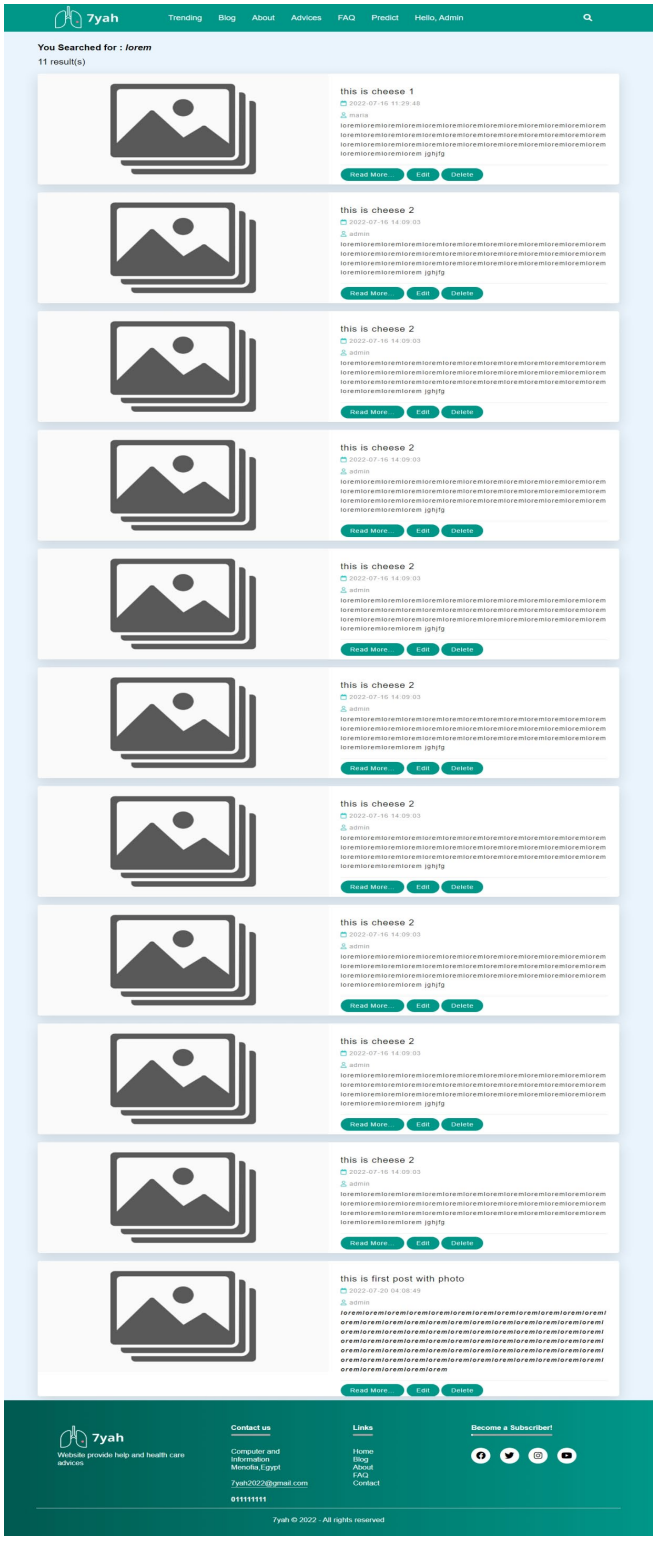
Edit comment:

This modal contains comment to be updated in the post.



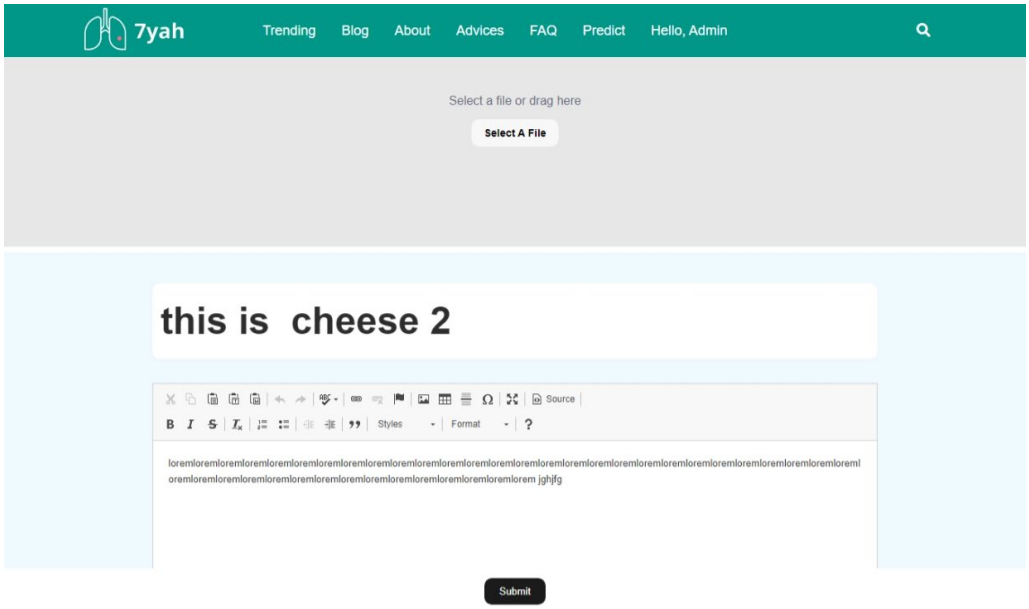
Search:

This page containing the search result for the word guest typed in whether it containing in title or content of the post.



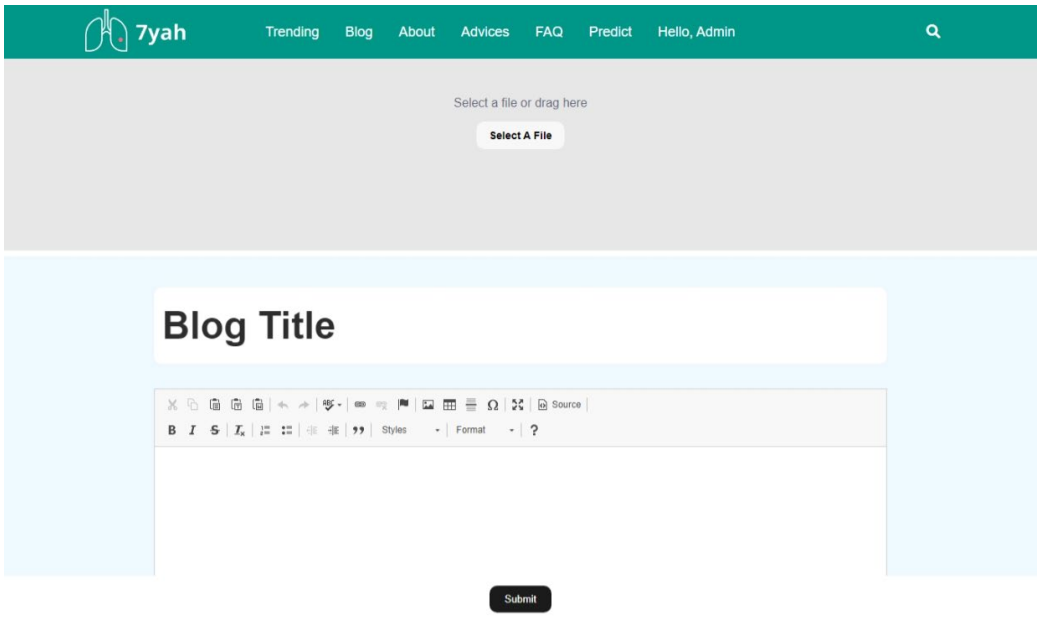
Edit Post:

This page enables the logged user to edit his post.



Add Post:

This page contain a form that gives ability , for doctor and admin, to publish a post in the website.



Logout:

This page enables the user to logout from the website and interact with as a guest.

8.3 FLUTTER

8.3.1 Why Flutter?

Flutter is an open-source software development kit which enables smooth and easy cross-platform mobile app development. You can build high quality natively compiled apps for iOS and Android quickly, without having to write the code for the two apps separately. All you need is one codebase for both platforms.

Flutter has numerous advantages over its competitors. These advantages are inherent in the programming language and in the set of development tools that allow Flutter to solve issues that other languages cannot cope with.

1. One codebase for all platforms

Gone are the days of having to write a code for Android and another codebase for iOS devices. Flutter's code reusability allows you to write just one codebase and use it on not only for mobile Android and iOS but even for web, desktop and more. This cuts development time significantly, removes cost and enables you launch your app that much faster.

2. "It is all Widgets" principle offers countless possibilities

Flutter's custom widgets are an absolute delight when it comes to creating great visuals for your app. At the same time, you don't have to worry about the UI on different devices.

3. Rich libraries

Flutter uses the Sika Graphics Library which is a fast and mature open-source graphics library. It redraws the UI every time a view changes. The result? A quick loading and smooth app experience.

4. Fast testing with hot reloads

The hot reload feature make the app development much quicker. With Flutter there is no need to reload the app to see every single change you make in the code. You can easily make changes in your app in real time, so you have more opportunity to experiment with the code and fix bugs on the go.

8.3.3 Some Libraries

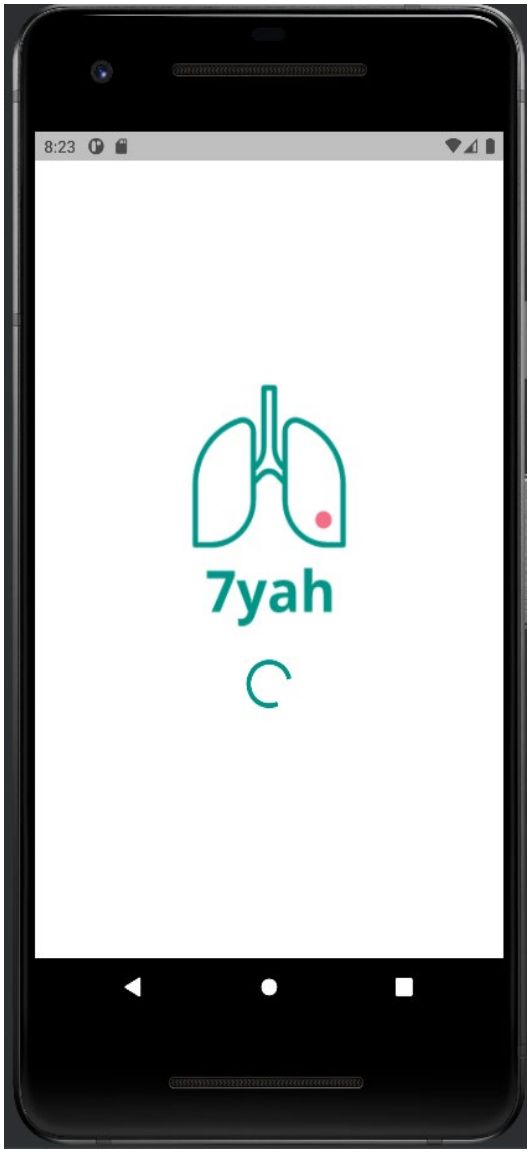
```
cupertino_icons: ^1.0.2
smooth_page_indicator: ^1.0.0+2
http: ^0.13.4
image_picker: ^0.8.5
path_provider: ^2.0.11
bloc: ^8.0.3
flutter_bloc: ^8.0.1
hexcolor: ^2.0.7
shared_preferences: ^2.0.15
```

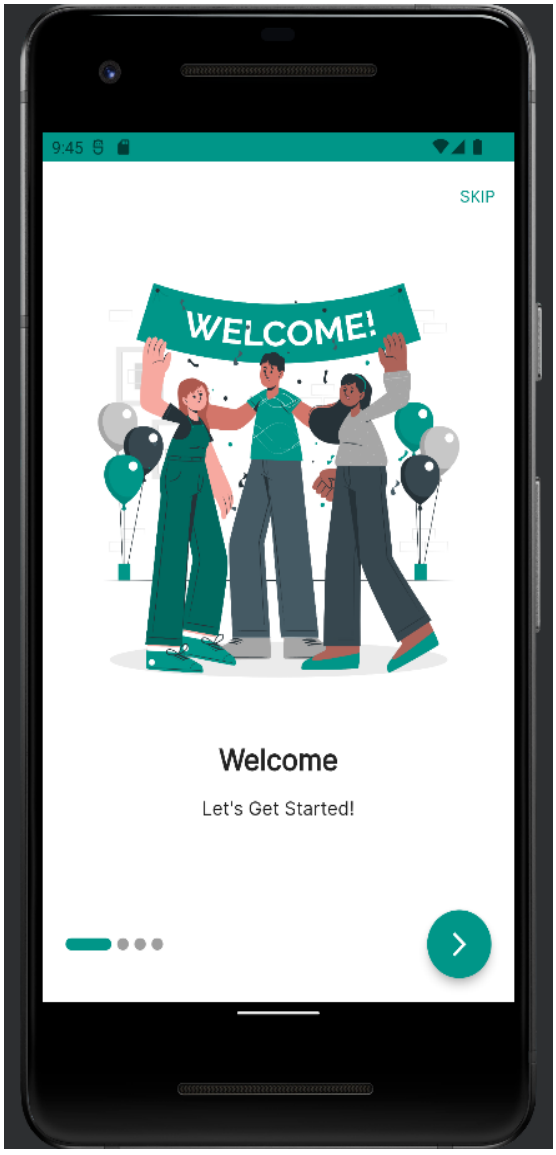
8.3.2 App Features

- Login
- Register
- Home
- Upload
- Advices
- Symptoms
- Settings
- Logout
- Dark Mode
- Shared Preference
- Dio Package
- Bloc State Management

1. Splash Screen

This page contains the program's logo.



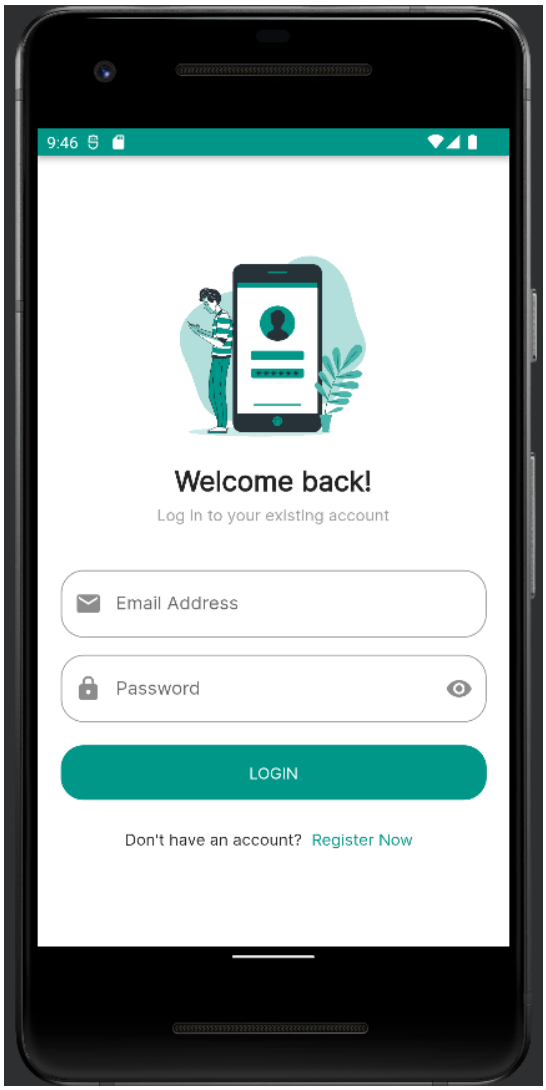


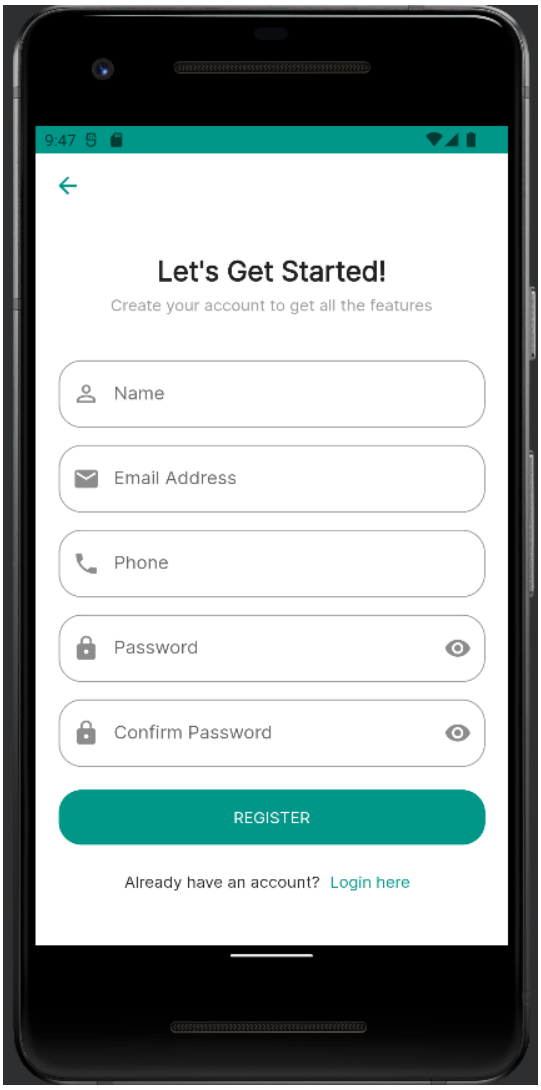
2. On-Boarding Screen

This page contains a definition of what the application offers and a welcome to the user with a button to skip this page and go to the login page directly.

3. Login Screen

This page is intended for entering a user name and password to access his own account within the program.



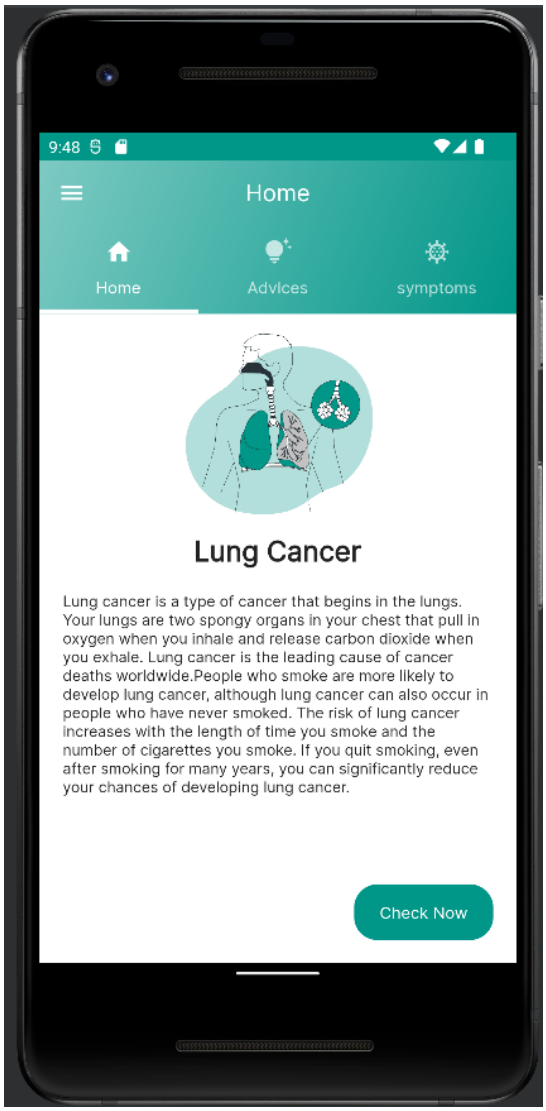


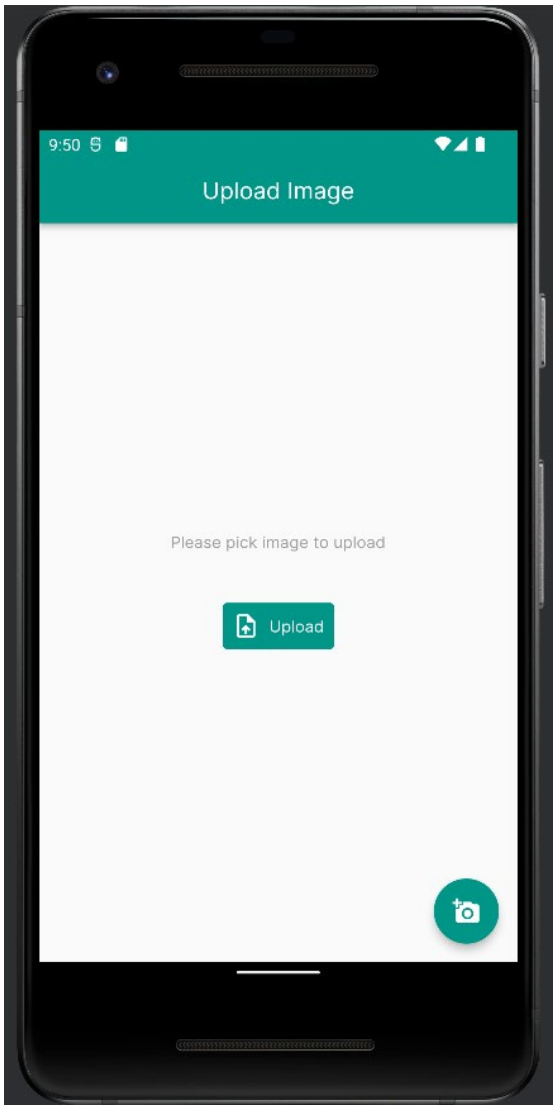
4. Register Screen

This page is intended for entering the data of new users to register a new account for them within the program.

5. Home Screen

This page contains a tab bar that consists of more than one interface such as the hints interface, the symptoms interface, the main interface, there is a button to go to the program's image upload page, and there is also a navigation bar with the user's name, email and other interfaces.



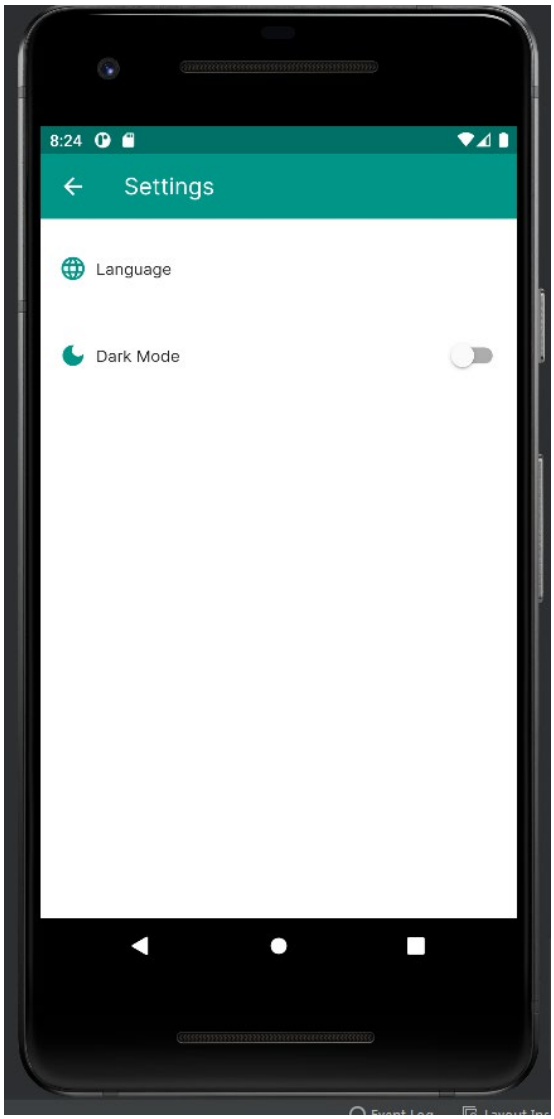


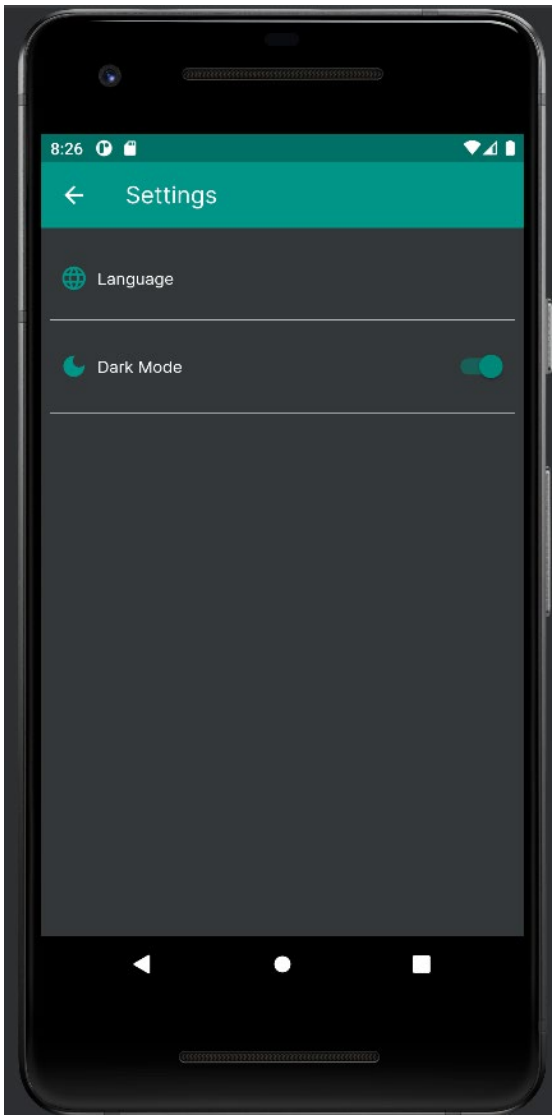
6. Upload Screen

This page was created to enable the user to upload x-ray images and check them to see if the x-ray patient has lung cancer or not.

7. Setting Screen

This page was created to enable the user to change App Theme.





9 CHAPTER NINE: CONCLUSION

The basic idea behind lung cancer classification is to use algorithms in deep neural networks. The web application implemented in this project helps in the early detection of lung cancer by inputting some medical analysis of patients and inputting it as new data to the used algorithm, then getting the prediction of the level of danger of the disease. We use a dataset for some patients with lung cancer to train the algorithm and test the accuracy and time consuming until we get the prediction.

10 APPENDIX

10.1 RESENT MODELBLOCKS

model.summary():

```
Layer (type)                 Output Shape              Param #   Connected to
==================================================================================================
input_1 (InputLayer)         [(None, 224, 224, 3  0           []
)]

conv1_pad (ZeroPadding2D)      (None, 230, 230, 3)  0          ['input_1[0][0]']

conv1_conv (Conv2D)            (None, 112, 112, 64  9472       ['conv1_pad[0][0]']
)

pool1_pad (ZeroPadding2D)      (None, 114, 114, 64  0          ['conv1_conv[0][0]']
)

pool1_pool (MaxPooling2D)      (None, 56, 56, 64)   0          ['pool1_pad[0][0]']

conv2_block1_preact_bn (BatchN  (None, 56, 56, 64)  256        ['pool1_pool[0][0]']
ormalization)

conv2_block1_preact_relu (Acti  (None, 56, 56, 64)  0          ['conv2_block1_preact_bn[0][0]']
vation)

conv2_block1_1_conv (Conv2D)   (None, 56, 56, 64)   4096       ['conv2_block1_preact_relu[0
                                                               ][0]'
                                                               ]

conv2_block1_1_bn (BatchNormal  (None, 56, 56, 64)  256        ['conv2_block1_1_conv[0][0]'
                                                               ]
ization)

conv2_block1_1_relu (Activatio  (None, 56, 56, 64)  0          ['conv2_block1_1_bn[0][0]']
n)

conv2_block1_2_pad (ZeroPaddin  (None, 58, 58, 64)  0          ['conv2_block1_1_relu[0][0]']
g2D)

conv2_block1_2_conv (Conv2D)   (None, 56, 56, 64)   36864      ['conv2_block1_2_pad[0][0]']
```

conv2_block1_2_bn (BatchNormal (None, 56, 56, 64) 256 ['conv2_block1_2_conv[0][0]'
]
ization)

conv2_block1_2_relu (Activatio (None, 56, 56, 64) 0 ['conv2_block1_2_bn[0][0]']
n)

conv2_block1_0_conv (Conv2D) (None, 56, 56, 256) 16640 ['conv2_block1_preact_relu[
0][0]'
]

conv2_block1_3_conv (Conv2D) (None, 56, 56, 256) 16640 ['conv2_block1_2_relu[0][0]
']

conv2_block1_out (Add) (None, 56, 56, 256) 0 ['conv2_block1_0_conv[0][0]',
'conv2_block1_3_conv[0][0]']

conv2_block2_preact_bn (BatchN (None, 56, 56, 256) 1024 ['conv2_block1_out[0][0]']
ormalization)

conv2_block2_preact_relu (Acti (None, 56, 56, 256) 0 ['conv2_block2_preact_bn[0][0]']
vation)

conv2_block2_1_conv (Conv2D) (None, 56, 56, 64) 16384 ['conv2_block2_preact_relu[
0][0]'
]

conv2_block2_1_bn (BatchNormal (None, 56, 56, 64) 256 ['conv2_block2_1_conv[0][0]'
]
ization)

conv2_block2_1_relu (Activatio (None, 56, 56, 64) 0 ['conv2_block2_1_bn[0][0]']
n)

conv2_block2_2_pad (ZeroPaddin (None, 58, 58, 64) 0 ['conv2_block2_1_relu[0][0]']
g2D)

conv2_block2_2_conv (Conv2D) (None, 56, 56, 64) 36864 ['conv2_block2_2_pad[0][0]']

conv2_block2_2_bn (BatchNormal (None, 56, 56, 64) 256 ['conv2_block2_2_conv[0][0]'
]
ization)

conv2_block2_2_relu (Activatio (None, 56, 56, 64) 0 ['conv2_block2_2_bn[0][0]']
n)

conv2_block2_3_conv (Conv2D) (None, 56, 56, 256) 16640 ['conv2_block2_2_relu[0][0]
']

conv2_block2_out (Add) (None, 56, 56, 256) 0 ['conv2_block1_out[0][0]',

'conv2_block2_3_conv[0][0]']

conv2_block3_preact_bn (BatchN ormalization) (None, 56, 56, 256) 1024 ['conv2_block2_out[0][0]']

conv2_block3_preact_relu (Acti vation) (None, 56, 56, 256) 0 ['conv2_block3_preact_bn[0][0]']

conv2_block3_1_conv (Conv2D) (None, 56, 56, 64) 16384 ['conv2_block3_preact_relu[0][0]']

conv2_block3_1_bn (BatchNormal ization) (None, 56, 56, 64) 256 ['conv2_block3_1_conv[0][0]']

conv2_block3_1_relu (Activatio n) (None, 56, 56, 64) 0 ['conv2_block3_1_bn[0][0]']

conv2_block3_2_pad (ZeroPaddin g2D) (None, 58, 58, 64) 0 ['conv2_block3_1_relu[0][0]']

conv2_block3_2_conv (Conv2D) (None, 28, 28, 64) 36864 ['conv2_block3_2_pad[0][0]']

conv2_block3_2_bn (BatchNormal ization) (None, 28, 28, 64) 256 ['conv2_block3_2_conv[0][0]']

conv2_block3_2_relu (Activatio n) (None, 28, 28, 64) 0 ['conv2_block3_2_bn[0][0]']

max_pooling2d (MaxPooling2D) (None, 28, 28, 256) 0 ['conv2_block2_out[0][0]']

conv2_block3_3_conv (Conv2D) (None, 28, 28, 256) 16640 ['conv2_block3_2_relu[0][0]']

conv2_block3_out (Add) (None, 28, 28, 256) 0 ['max_pooling2d[0][0]', 'conv2_block3_3_conv[0][0]']

conv3_block1_preact_bn (BatchN ormalization) (None, 28, 28, 256) 1024 ['conv2_block3_out[0][0]']

conv3_block1_preact_relu (Acti vation) (None, 28, 28, 256) 0 ['conv3_block1_preact_bn[0][0]']

conv3_block1_1_conv (Conv2D) (None, 28, 28, 128) 32768 ['conv3_block1_preact_relu[0][0]']

conv3_block1_1_bn (BatchNormal (None, 28, 28, 128) 512 ['conv3_block1_1_conv[0][0]
']
ization)

conv3_block1_1_relu (Activatio (None, 28, 28, 128) 0 ['conv3_block1_1_bn[0][0]']
n)

conv3_block1_2_pad (ZeroPaddin (None, 30, 30, 128) 0 ['conv3_block1_1_relu[0][0]']
g2D)

conv3_block1_2_conv (Conv2D) (None, 28, 28, 128) 147456 ['conv3_block1_2_pad[0][0]
']

conv3_block1_2_bn (BatchNormal (None, 28, 28, 128) 512 ['conv3_block1_2_conv[0][0]
']
ization)

conv3_block1_2_relu (Activatio (None, 28, 28, 128) 0 ['conv3_block1_2_bn[0][0]']
n)

conv3_block1_0_conv (Conv2D) (None, 28, 28, 512) 131584 ['conv3_block1_preact_relu
[0][0]'
]

conv3_block1_3_conv (Conv2D) (None, 28, 28, 512) 66048 ['conv3_block1_2_relu[0][0]
']

conv3_block1_out (Add) (None, 28, 28, 512) 0 ['conv3_block1_0_conv[0][0]',
'conv3_block1_3_conv[0][0]']

conv3_block2_preact_bn (BatchN (None, 28, 28, 512) 2048 ['conv3_block1_out[0][0]']
ormalization)

conv3_block2_preact_relu (Acti (None, 28, 28, 512) 0 ['conv3_block2_preact_bn[0][0]']
vation)

conv3_block2_1_conv (Conv2D) (None, 28, 28, 128) 65536 ['conv3_block2_preact_relu[
0][0]'
]

conv3_block2_1_bn (BatchNormal (None, 28, 28, 128) 512 ['conv3_block2_1_conv[0][0]
']
ization)

conv3_block2_1_relu (Activatio (None, 28, 28, 128) 0 ['conv3_block2_1_bn[0][0]']
n)

conv3_block2_2_pad (ZeroPaddin (None, 30, 30, 128) 0 ['conv3_block2_1_relu[0][0]']
g2D)

conv3_block2_2_conv (Conv2D) (None, 28, 28, 128) 147456 ['conv3_block2_2_pad[0][0]']

conv3_block2_2_bn (BatchNormalization) (None, 28, 28, 128) 512 ['conv3_block2_2_conv[0][0]']

conv3_block2_2_relu (Activation) (None, 28, 28, 128) 0 ['conv3_block2_2_bn[0][0]']

conv3_block2_3_conv (Conv2D) (None, 28, 28, 512) 66048 ['conv3_block2_2_relu[0][0]']

conv3_block2_out (Add) (None, 28, 28, 512) 0 ['conv3_block1_out[0][0]', 'conv3_block2_3_conv[0][0]']

conv3_block3_preact_bn (BatchNormalization) (None, 28, 28, 512) 2048 ['conv3_block2_out[0][0]']

conv3_block3_preact_relu (Activation) (None, 28, 28, 512) 0 ['conv3_block3_preact_bn[0][0]']

conv3_block3_1_conv (Conv2D) (None, 28, 28, 128) 65536 ['conv3_block3_preact_relu[0][0]']

conv3_block3_1_bn (BatchNormalization) (None, 28, 28, 128) 512 ['conv3_block3_1_conv[0][0]']

conv3_block3_1_relu (Activation) (None, 28, 28, 128) 0 ['conv3_block3_1_bn[0][0]']

conv3_block3_2_pad (ZeroPadding2D) (None, 30, 30, 128) 0 ['conv3_block3_1_relu[0][0]']

conv3_block3_2_conv (Conv2D) (None, 28, 28, 128) 147456 ['conv3_block3_2_pad[0][0]']

conv3_block3_2_bn (BatchNormalization) (None, 28, 28, 128) 512 ['conv3_block3_2_conv[0][0]']

conv3_block3_2_relu (Activation) (None, 28, 28, 128) 0 ['conv3_block3_2_bn[0][0]']

conv3_block3_3_conv (Conv2D) (None, 28, 28, 512) 66048 ['conv3_block3_2_relu[0][0]']

```
conv3_block3_out (Add)        (None, 28, 28, 512)  0          ['conv3_block2_out[0][0]',
                                                               'conv3_block3_3_conv[0][0]']

conv3_block4_preact_bn (BatchN  (None, 28, 28, 512)  2048      ['conv3_block3_out[0][0]']

ormalization)

conv3_block4_preact_relu (Acti  (None, 28, 28, 512)  0         ['conv3_block4_preact_bn[0][0]'
]
vation)

conv3_block4_1_conv (Conv2D)   (None, 28, 28, 128)  65536      ['conv3_block4_preact_relu
[0][0]'
                                                               ]

conv3_block4_1_bn (BatchNormal  (None, 28, 28, 128)  512       ['conv3_block4_1_conv[0][0
]']
ization)

conv3_block4_1_relu (Activatio  (None, 28, 28, 128)  0         ['conv3_block4_1_bn[0][0]']
n)

conv3_block4_2_pad (ZeroPaddin  (None, 30, 30, 128)  0         ['conv3_block4_1_relu[0][0]']

g2D)

conv3_block4_2_conv (Conv2D)   (None, 14, 14, 128)  147456     ['conv3_block4_2_pad[0][0
]']

conv3_block4_2_bn (BatchNormal  (None, 14, 14, 128)  512       ['conv3_block4_2_conv[0][0
]']
ization)

conv3_block4_2_relu (Activatio  (None, 14, 14, 128)  0         ['conv3_block4_2_bn[0][0]']
n)

max_pooling2d_1 (MaxPooling2D)  (None, 14, 14, 512)  0         ['conv3_block3_out[0][0]']


conv3_block4_3_conv (Conv2D)   (None, 14, 14, 512)  66048      ['conv3_block4_2_relu[0][0
]']

conv3_block4_out (Add)        (None, 14, 14, 512)  0          ['max_pooling2d_1[0][0]',
                                                               'conv3_block4_3_conv[0][0]']

conv4_block1_preact_bn (BatchN  (None, 14, 14, 512)  2048      ['conv3_block4_out[0][0]']

ormalization)
```

conv4_block1_preact_relu (Acti (None, 14, 14, 512) 0 ['conv4_block1_preact_bn[0][0]'
]
vation)

conv4_block1_1_conv (Conv2D) (None, 14, 14, 256) 131072 ['conv4_block1_preact_rel
u[0][0]'
]

conv4_block1_1_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block1_1_conv[0][
0]']
ization)

conv4_block1_1_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block1_1_bn[0][0]']
n)

conv4_block1_2_pad (ZeroPaddin (None, 16, 16, 256) 0 ['conv4_block1_1_relu[0][0]']

g2D)

conv4_block1_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block1_2_pad[0][0
]']

conv4_block1_2_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block1_2_conv[0][
0]']
ization)

conv4_block1_2_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block1_2_bn[0][0]']
n)

conv4_block1_0_conv (Conv2D) (None, 14, 14, 1024 525312 ['conv4_block1_preact_rel
u[0][0]'
)]

conv4_block1_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block1_2_relu[0][
0]']
)

conv4_block1_out (Add) (None, 14, 14, 1024 0 ['conv4_block1_0_conv[0][0]',
) 'conv4_block1_3_conv[0][0]']

conv4_block2_preact_bn (BatchN (None, 14, 14, 1024 4096 ['conv4_block1_out[0][0]']

ormalization))

conv4_block2_preact_relu (Acti (None, 14, 14, 1024 0 ['conv4_block2_preact_bn[0][0]'
]
vation))

conv4_block2_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block2_preact_rel
u[0][0]'

]

conv4_block2_1_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block2_1_conv[0][0]']
ization)

conv4_block2_1_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block2_1_bn[0][0]']
n)

conv4_block2_2_pad (ZeroPaddin (None, 16, 16, 256) 0 ['conv4_block2_1_relu[0][0]']

g2D)

conv4_block2_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block2_2_pad[0][0]']

conv4_block2_2_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block2_2_conv[0][0]']
ization)

conv4_block2_2_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block2_2_bn[0][0]']
n)

conv4_block2_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block2_2_relu[0][0]']
)

conv4_block2_out (Add) (None, 14, 14, 1024 0 ['conv4_block1_out[0][0]',
) 'conv4_block2_3_conv[0][0]']

conv4_block3_preact_bn (BatchN (None, 14, 14, 1024 4096 ['conv4_block2_out[0][0]']

ormalization))

conv4_block3_preact_relu (Acti (None, 14, 14, 1024 0 ['conv4_block3_preact_bn[0][0]']
vation))

conv4_block3_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block3_preact_relu[0][0]'
]

conv4_block3_1_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block3_1_conv[0][0]']
ization)

conv4_block3_1_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block3_1_bn[0][0]']
n)

conv4_block3_2_pad (ZeroPaddin (None, 16, 16, 256) 0 ['conv4_block3_1_relu[0][0']

g2D)

conv4_block3_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block3_2_pad[0][0
]']

conv4_block3_2_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block3_2_conv[0][
0]']
ization)

conv4_block3_2_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block3_2_bn[0][0]']
n)

conv4_block3_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block3_2_relu[0][
0]']
)

conv4_block3_out (Add) (None, 14, 14, 1024 0 ['conv4_block2_out[0][0]',
) 'conv4_block3_3_conv[0][0]']

conv4_block4_preact_bn (BatchN (None, 14, 14, 1024 4096 ['conv4_block3_out[0][0]']

ormalization))

conv4_block4_preact_relu (Acti (None, 14, 14, 1024 0 ['conv4_block4_preact_bn[0][0]'
]
vation))

conv4_block4_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block4_preact_rel
u[0][0]'
]

conv4_block4_1_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block4_1_conv[0][
0]']
ization)

conv4_block4_1_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block4_1_bn[0][0]']
n)

conv4_block4_2_pad (ZeroPaddin (None, 16, 16, 256) 0 ['conv4_block4_1_relu[0][0']

g2D)

conv4_block4_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block4_2_pad[0][0
]']

conv4_block4_2_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block4_2_conv[0][
0]']
ization)

```
conv4_block4_2_relu (Activatio  (None, 14, 14, 256)  0         ['conv4_block4_2_bn[0][0]']
n)

conv4_block4_3_conv (Conv2D)   (None, 14, 14, 1024  263168     ['conv4_block4_2_relu[0][
0]']
                               )

conv4_block4_out (Add)         (None, 14, 14, 1024  0          ['conv4_block3_out[0][0]',
                               )                                'conv4_block4_3_conv[0][0]']

conv4_block5_preact_bn (BatchN  (None, 14, 14, 1024  4096      ['conv4_block4_out[0][0]']

ormalization)                  )

conv4_block5_preact_relu (Acti  (None, 14, 14, 1024  0         ['conv4_block5_preact_bn[0][0]'
]
vation)                        )

conv4_block5_1_conv (Conv2D)   (None, 14, 14, 256)  262144     ['conv4_block5_preact_rel
u[0][0]'
                                                               ]

conv4_block5_1_bn (BatchNormal  (None, 14, 14, 256)  1024      ['conv4_block5_1_conv[0][
0]']
ization)

conv4_block5_1_relu (Activatio  (None, 14, 14, 256)  0         ['conv4_block5_1_bn[0][0]']
n)

conv4_block5_2_pad (ZeroPaddin  (None, 16, 16, 256)  0         ['conv4_block5_1_relu[0][0]']

g2D)

conv4_block5_2_conv (Conv2D)   (None, 14, 14, 256)  589824     ['conv4_block5_2_pad[0][0
]']

conv4_block5_2_bn (BatchNormal  (None, 14, 14, 256)  1024      ['conv4_block5_2_conv[0][
0]']
ization)

conv4_block5_2_relu (Activatio  (None, 14, 14, 256)  0         ['conv4_block5_2_bn[0][0]']
n)

conv4_block5_3_conv (Conv2D)   (None, 14, 14, 1024  263168     ['conv4_block5_2_relu[0][
0]']
                               )

conv4_block5_out (Add)         (None, 14, 14, 1024  0          ['conv4_block4_out[0][0]',
                               )                                'conv4_block5_3_conv[0][0]']
```

```
conv4_block6_preact_bn (BatchN  (None, 14, 14, 1024  4096      ['conv4_block5_out[0][0]']
ormalization)                )

conv4_block6_preact_relu (Acti  (None, 14, 14, 1024  0         ['conv4_block6_preact_bn[0][0]'
]
vation)                      )

conv4_block6_1_conv (Conv2D)   (None, 14, 14, 256)  262144     ['conv4_block6_preact_rel
u[0][0]'
                                                                ]

conv4_block6_1_bn (BatchNormal  (None, 14, 14, 256)  1024      ['conv4_block6_1_conv[0][[
0]']
ization)

conv4_block6_1_relu (Activatio  (None, 14, 14, 256)  0         ['conv4_block6_1_bn[0][0]']
n)

conv4_block6_2_pad (ZeroPaddin  (None, 16, 16, 256)  0         ['conv4_block6_1_relu[0][0]']
g2D)

conv4_block6_2_conv (Conv2D)   (None, 14, 14, 256)  589824     ['conv4_block6_2_pad[0][0
]']

conv4_block6_2_bn (BatchNormal  (None, 14, 14, 256)  1024      ['conv4_block6_2_conv[0][[
0]']
ization)

conv4_block6_2_relu (Activatio  (None, 14, 14, 256)  0         ['conv4_block6_2_bn[0][0]']
n)

conv4_block6_3_conv (Conv2D)   (None, 14, 14, 1024  263168     ['conv4_block6_2_relu[0][[
0]']
                             )

conv4_block6_out (Add)         (None, 14, 14, 1024  0          ['conv4_block5_out[0][0]',
                             )                                  'conv4_block6_3_conv[0][0]']

conv4_block7_preact_bn (BatchN  (None, 14, 14, 1024  4096      ['conv4_block6_out[0][0]']
ormalization)                )

conv4_block7_preact_relu (Acti  (None, 14, 14, 1024  0         ['conv4_block7_preact_bn[0][0]'
]
vation)                      )
```

conv4_block7_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block7_preact_rel
u[0][0]'
                                ]

conv4_block7_1_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block7_1_conv[0][
0]']
ization)

conv4_block7_1_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block7_1_bn[0][0]']
n)

conv4_block7_2_pad (ZeroPaddin (None, 16, 16, 256) 0 ['conv4_block7_1_relu[0][0]']

g2D)

conv4_block7_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block7_2_pad[0][0
]']

conv4_block7_2_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block7_2_conv[0][
0]']
ization)

conv4_block7_2_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block7_2_bn[0][0]']
n)

conv4_block7_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block7_2_relu[0][
0]']
)

conv4_block7_out (Add) (None, 14, 14, 1024 0 ['conv4_block6_out[0][0]',
)   'conv4_block7_3_conv[0][0]']

conv4_block8_preact_bn (BatchN (None, 14, 14, 1024 4096 ['conv4_block7_out[0][0]']

ormalization)   )

conv4_block8_preact_relu (Acti (None, 14, 14, 1024 0 ['conv4_block8_preact_bn[0][0]'
]
vation)   )

conv4_block8_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block8_preact_rel
u[0][0]'
                                ]

conv4_block8_1_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block8_1_conv[0][
0]']
ization)

conv4_block8_1_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block8_1_bn[0][0]']
n)

conv4_block8_2_pad (ZeroPaddin (None, 16, 16, 256) 0 ['conv4_block8_1_relu[0][0]']
g2D)

conv4_block8_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block8_2_pad[0][0]']

conv4_block8_2_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block8_2_conv[0][0]']
ization)

conv4_block8_2_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block8_2_bn[0][0]']
n)

conv4_block8_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block8_2_relu[0][0]']
)

conv4_block8_out (Add) (None, 14, 14, 1024 0 ['conv4_block7_out[0][0]',
) 'conv4_block8_3_conv[0][0]']

conv4_block9_preact_bn (BatchN (None, 14, 14, 1024 4096 ['conv4_block8_out[0][0]']
ormalization))

conv4_block9_preact_relu (Acti (None, 14, 14, 1024 0 ['conv4_block9_preact_bn[0][0]']
vation))

conv4_block9_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block9_preact_relu[0][0]'
]

conv4_block9_1_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block9_1_conv[0][0]']
ization)

conv4_block9_1_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block9_1_bn[0][0]']
n)

conv4_block9_2_pad (ZeroPaddin (None, 16, 16, 256) 0 ['conv4_block9_1_relu[0][0]']
g2D)

conv4_block9_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block9_2_pad[0][0]']

conv4_block9_2_bn (BatchNormal (None, 14, 14, 256) 1024 ['conv4_block9_2_conv[0][0]']

ization)

conv4_block9_2_relu (Activatio (None, 14, 14, 256) 0 ['conv4_block9_2_bn[0][0]']
n)

conv4_block9_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block9_2_relu[0][0]']
)

conv4_block9_out (Add) (None, 14, 14, 1024 0 ['conv4_block8_out[0][0]',
) 'conv4_block9_3_conv[0][0]']

conv4_block10_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block9_out[0][0]']

Normalization))

conv4_block10_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block10_preact_bn[0][0]']
ivation))

conv4_block10_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block10_preact_relu[0][0]
']

conv4_block10_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block10_1_conv[0][0]']
lization)

conv4_block10_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block10_1_bn[0][0]']
on)

conv4_block10_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block10_1_relu[0][0]']
ng2D)

conv4_block10_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block10_2_pad[0][0]']

conv4_block10_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block10_2_conv[0][0]']
lization)

conv4_block10_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block10_2_bn[0][0]']
on)

conv4_block10_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block10_2_relu[0][0]']
)

conv4_block10_out (Add) (None, 14, 14, 1024 0 ['conv4_block9_out[0][0]',

) 'conv4_block10_3_conv[0][0]']

conv4_block11_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block10_out[0][0]']

Normalization))

conv4_block11_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block11_preact_bn[0][0]']
ivation))

conv4_block11_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block11_preact_relu[0][0]']

conv4_block11_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block11_1_conv[0][0]']
lization)

conv4_block11_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block11_1_bn[0][0]']
on)

conv4_block11_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block11_1_relu[0][0]']
ng2D)

conv4_block11_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block11_2_pad[0][0]']

conv4_block11_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block11_2_conv[0][0]']
lization)

conv4_block11_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block11_2_bn[0][0]']
on)

conv4_block11_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block11_2_relu[0][0]']
)

conv4_block11_out (Add) (None, 14, 14, 1024 0 ['conv4_block10_out[0][0]',
) 'conv4_block11_3_conv[0][0]']

conv4_block12_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block11_out[0][0]']

Normalization))

conv4_block12_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block12_preact_bn[0][0]']
ivation))

conv4_block12_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block12_preact_relu[0][0]']

conv4_block12_1_bn (BatchNormalization) (None, 14, 14, 256) 1024 ['conv4_block12_1_conv[0][0]']

conv4_block12_1_relu (Activation) (None, 14, 14, 256) 0 ['conv4_block12_1_bn[0][0]']

conv4_block12_2_pad (ZeroPadding2D) (None, 16, 16, 256) 0 ['conv4_block12_1_relu[0][0]']

conv4_block12_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block12_2_pad[0][0]']

conv4_block12_2_bn (BatchNormalization) (None, 14, 14, 256) 1024 ['conv4_block12_2_conv[0][0]']

conv4_block12_2_relu (Activation) (None, 14, 14, 256) 0 ['conv4_block12_2_bn[0][0]']

conv4_block12_3_conv (Conv2D) (None, 14, 14, 1024) 263168 ['conv4_block12_2_relu[0][0]']

conv4_block12_out (Add) (None, 14, 14, 1024) 0 ['conv4_block11_out[0][0]', 'conv4_block12_3_conv[0][0]']

conv4_block13_preact_bn (BatchNormalization) (None, 14, 14, 1024) 4096 ['conv4_block12_out[0][0]']

conv4_block13_preact_relu (Activation) (None, 14, 14, 1024) 0 ['conv4_block13_preact_bn[0][0]']

conv4_block13_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block13_preact_relu[0][0]']

conv4_block13_1_bn (BatchNormalization) (None, 14, 14, 256) 1024 ['conv4_block13_1_conv[0][0]']

conv4_block13_1_relu (Activation) (None, 14, 14, 256) 0 ['conv4_block13_1_bn[0][0]']

on)

conv4_block13_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block13_1_relu[0][0]'
]
ng2D)

conv4_block13_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block13_2_pad[0]
[0]']

conv4_block13_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block13_2_conv[0
][0]']
lization)

conv4_block13_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block13_2_bn[0][0]']
on)

conv4_block13_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block13_2_relu[0
][0]']
)

conv4_block13_out (Add) (None, 14, 14, 1024 0 ['conv4_block12_out[0][0]',
) 'conv4_block13_3_conv[0][0]']

conv4_block14_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block13_out[0][0]']

Normalization))

conv4_block14_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block14_preact_bn[0][
0]']
ivation))

conv4_block14_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block14_preact_r
elu[0][0]
']

conv4_block14_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block14_1_conv[0
][0]']
lization)

conv4_block14_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block14_1_bn[0][0]']
on)

conv4_block14_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block14_1_relu[0][0]'
]
ng2D)

conv4_block14_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block14_2_pad[0]
[0]']

conv4_block14_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block14_2_conv[0
lization)][0]']

conv4_block14_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block14_2_bn[0][0]']
on)

conv4_block14_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block14_2_relu[0
)][0]']

conv4_block14_out (Add) (None, 14, 14, 1024 0 ['conv4_block13_out[0][0]',
) 'conv4_block14_3_conv[0][0]']

conv4_block15_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block14_out[0][0]']
Normalization))

conv4_block15_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block15_preact_bn[0][
ivation)) 0]']

conv4_block15_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block15_preact_r
elu[0][0]
']

conv4_block15_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block15_1_conv[0
lization)][0]']

conv4_block15_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block15_1_bn[0][0]']
on)

conv4_block15_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block15_1_relu[0][0]'
ng2D)]

conv4_block15_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block15_2_pad[0]
[0]']

conv4_block15_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block15_2_conv[0
lization)][0]']

conv4_block15_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block15_2_bn[0][0]']
on)

conv4_block15_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block15_2_relu[0
)][0]']

conv4_block15_out (Add) (None, 14, 14, 1024 0 ['conv4_block14_out[0][0]',
) 'conv4_block15_3_conv[0][0]']

conv4_block16_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block15_out[0][0]']

Normalization))

conv4_block16_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block16_preact_bn[0][
0]']
ivation))

conv4_block16_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block16_preact_r
elu[0][0]
']

conv4_block16_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block16_1_conv[0
][0]']
lization)

conv4_block16_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block16_1_bn[0][0]']
on)

conv4_block16_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block16_1_relu[0][0]'
]
ng2D)

conv4_block16_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block16_2_pad[0]
[0]']

conv4_block16_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block16_2_conv[0
][0]']
lization)

conv4_block16_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block16_2_bn[0][0]']
on)

conv4_block16_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block16_2_relu[0
][0]']
)

conv4_block16_out (Add) (None, 14, 14, 1024 0 ['conv4_block15_out[0][0]',
) 'conv4_block16_3_conv[0][0]']

conv4_block17_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block16_out[0][0]']

Normalization))

conv4_block17_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block17_preact_bn[0][
0]']

ivation)   )

conv4_block17_1_conv (Conv2D) (None, 14, 14, 256) 262144   ['conv4_block17_preact_r
elu[0][0]
                                                                 ']

conv4_block17_1_bn (BatchNorma (None, 14, 14, 256) 1024   ['conv4_block17_1_conv[0
][0]']
lization)

conv4_block17_1_relu (Activati (None, 14, 14, 256) 0   ['conv4_block17_1_bn[0][0]']
on)

conv4_block17_2_pad (ZeroPaddi (None, 16, 16, 256) 0   ['conv4_block17_1_relu[0][0]'
]
ng2D)

conv4_block17_2_conv (Conv2D) (None, 14, 14, 256) 589824   ['conv4_block17_2_pad[0]
[0]']

conv4_block17_2_bn (BatchNorma (None, 14, 14, 256) 1024   ['conv4_block17_2_conv[0
][0]']
lization)

conv4_block17_2_relu (Activati (None, 14, 14, 256) 0   ['conv4_block17_2_bn[0][0]']
on)

conv4_block17_3_conv (Conv2D) (None, 14, 14, 1024 263168   ['conv4_block17_2_relu[0
][0]']
)

conv4_block17_out (Add)   (None, 14, 14, 1024 0   ['conv4_block16_out[0][0]',
)   'conv4_block17_3_conv[0][0]']

conv4_block18_preact_bn (Batch (None, 14, 14, 1024 4096   ['conv4_block17_out[0][0]']

Normalization)   )

conv4_block18_preact_relu (Act (None, 14, 14, 1024 0   ['conv4_block18_preact_bn[0][
0]']
ivation)   )

conv4_block18_1_conv (Conv2D) (None, 14, 14, 256) 262144   ['conv4_block18_preact_r
elu[0][0]
                                                                 ']

conv4_block18_1_bn (BatchNorma (None, 14, 14, 256) 1024   ['conv4_block18_1_conv[0
][0]']
lization)

```
conv4_block18_1_relu (Activati  (None, 14, 14, 256)  0         ['conv4_block18_1_bn[0][0]']
on)

conv4_block18_2_pad (ZeroPaddi  (None, 16, 16, 256)  0         ['conv4_block18_1_relu[0][0]'
]
ng2D)

conv4_block18_2_conv (Conv2D)  (None, 14, 14, 256)  589824     ['conv4_block18_2_pad[0]
[0]']

conv4_block18_2_bn (BatchNorma  (None, 14, 14, 256)  1024      ['conv4_block18_2_conv[0
][0]']
lization)

conv4_block18_2_relu (Activati  (None, 14, 14, 256)  0         ['conv4_block18_2_bn[0][0]']
on)

conv4_block18_3_conv (Conv2D)  (None, 14, 14, 1024  263168     ['conv4_block18_2_relu[0
][0]']
                               )

conv4_block18_out (Add)        (None, 14, 14, 1024  0          ['conv4_block17_out[0][0]',
                               )                                'conv4_block18_3_conv[0][0]']

conv4_block19_preact_bn (Batch  (None, 14, 14, 1024  4096      ['conv4_block18_out[0][0]']

Normalization)                 )

conv4_block19_preact_relu (Act  (None, 14, 14, 1024  0         ['conv4_block19_preact_bn[0][
0]']
ivation)                       )

conv4_block19_1_conv (Conv2D)  (None, 14, 14, 256)  262144     ['conv4_block19_preact_r
elu[0][0]
                                                               ']

conv4_block19_1_bn (BatchNorma  (None, 14, 14, 256)  1024      ['conv4_block19_1_conv[0
][0]']
lization)

conv4_block19_1_relu (Activati  (None, 14, 14, 256)  0         ['conv4_block19_1_bn[0][0]']
on)

conv4_block19_2_pad (ZeroPaddi  (None, 16, 16, 256)  0         ['conv4_block19_1_relu[0][0]'
]
ng2D)

conv4_block19_2_conv (Conv2D)  (None, 14, 14, 256)  589824     ['conv4_block19_2_pad[0]
[0]']
```

conv4_block19_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block19_2_conv[0
][0]']
lization)

conv4_block19_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block19_2_bn[0][0]']
on)

conv4_block19_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block19_2_relu[0
][0]']
)

conv4_block19_out (Add) (None, 14, 14, 1024 0 ['conv4_block18_out[0][0]',
) 'conv4_block19_3_conv[0][0]']

conv4_block20_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block19_out[0][0]']

Normalization))

conv4_block20_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block20_preact_bn[0][
0]']
ivation))

conv4_block20_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block20_preact_r
elu[0][0]
']

conv4_block20_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block20_1_conv[0
][0]']
lization)

conv4_block20_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block20_1_bn[0][0]']
on)

conv4_block20_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block20_1_relu[0][0]'
]
ng2D)

conv4_block20_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block20_2_pad[0]
[0]']

conv4_block20_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block20_2_conv[0
][0]']
lization)

conv4_block20_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block20_2_bn[0][0]']
on)

conv4_block20_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block20_2_relu[0
][0]']
)

conv4_block20_out (Add) (None, 14, 14, 1024 0 ['conv4_block19_out[0][0]',
) 'conv4_block20_3_conv[0][0]']

conv4_block21_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block20_out[0][0]']

Normalization))

conv4_block21_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block21_preact_bn[0][
0]']
ivation))

conv4_block21_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block21_preact_r
elu[0][0]
']

conv4_block21_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block21_1_conv[0
][0]']
lization)

conv4_block21_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block21_1_bn[0][0]']
on)

conv4_block21_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block21_1_relu[0][0]'
]
ng2D)

conv4_block21_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block21_2_pad[0]
[0]']

conv4_block21_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block21_2_conv[0
][0]']
lization)

conv4_block21_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block21_2_bn[0][0]']
on)

conv4_block21_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block21_2_relu[0
][0]']
)

conv4_block21_out (Add) (None, 14, 14, 1024 0 ['conv4_block20_out[0][0]',
) 'conv4_block21_3_conv[0][0]']

conv4_block22_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block21_out[0][0]']

Normalization))

conv4_block22_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block22_preact_bn[0][0]']
ivation))

conv4_block22_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block22_preact_relu[0][0]']

conv4_block22_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block22_1_conv[0][0]']
lization)

conv4_block22_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block22_1_bn[0][0]']
on)

conv4_block22_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block22_1_relu[0][0]']
ng2D)

conv4_block22_2_conv (Conv2D) (None, 14, 14, 256) 589824 ['conv4_block22_2_pad[0][0]']

conv4_block22_2_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block22_2_conv[0][0]']
lization)

conv4_block22_2_relu (Activati (None, 14, 14, 256) 0 ['conv4_block22_2_bn[0][0]']
on)

conv4_block22_3_conv (Conv2D) (None, 14, 14, 1024 263168 ['conv4_block22_2_relu[0][0]']
)

conv4_block22_out (Add) (None, 14, 14, 1024 0 ['conv4_block21_out[0][0]',
) 'conv4_block22_3_conv[0][0]']

conv4_block23_preact_bn (Batch (None, 14, 14, 1024 4096 ['conv4_block22_out[0][0]']

Normalization))

conv4_block23_preact_relu (Act (None, 14, 14, 1024 0 ['conv4_block23_preact_bn[0][0]']
ivation))

conv4_block23_1_conv (Conv2D) (None, 14, 14, 256) 262144 ['conv4_block23_preact_relu[0][0]']

conv4_block23_1_bn (BatchNorma (None, 14, 14, 256) 1024 ['conv4_block23_1_conv[0][0]']

lization)

conv4_block23_1_relu (Activati (None, 14, 14, 256) 0 ['conv4_block23_1_bn[0][0]']
on)

conv4_block23_2_pad (ZeroPaddi (None, 16, 16, 256) 0 ['conv4_block23_1_relu[0][0]'
]
ng2D)

conv4_block23_2_conv (Conv2D) (None, 7, 7, 256) 589824 ['conv4_block23_2_pad[0][
0]']

conv4_block23_2_bn (BatchNorma (None, 7, 7, 256) 1024 ['conv4_block23_2_conv[0][
0]']
lization)

conv4_block23_2_relu (Activati (None, 7, 7, 256) 0 ['conv4_block23_2_bn[0][0]']
on)

max_pooling2d_2 (MaxPooling2D) (None, 7, 7, 1024) 0 ['conv4_block22_out[0][0]']

conv4_block23_3_conv (Conv2D) (None, 7, 7, 1024) 263168 ['conv4_block23_2_relu[0][
0]']

conv4_block23_out (Add) (None, 7, 7, 1024) 0 ['max_pooling2d_2[0][0]',
'conv4_block23_3_conv[0][0]']

conv5_block1_preact_bn (BatchN (None, 7, 7, 1024) 4096 ['conv4_block23_out[0][0]']

ormalization)

conv5_block1_preact_relu (Acti (None, 7, 7, 1024) 0 ['conv5_block1_preact_bn[0][0]']

vation)

conv5_block1_1_conv (Conv2D) (None, 7, 7, 512) 524288 ['conv5_block1_preact_relu[
0][0]'
]

conv5_block1_1_bn (BatchNormal (None, 7, 7, 512) 2048 ['conv5_block1_1_conv[0][0]
']
ization)

conv5_block1_1_relu (Activatio (None, 7, 7, 512) 0 ['conv5_block1_1_bn[0][0]']
n)

conv5_block1_2_pad (ZeroPaddin (None, 9, 9, 512) 0 ['conv5_block1_1_relu[0][0]']
g2D)

```
conv5_block1_2_conv (Conv2D)   (None, 7, 7, 512)    2359296    ['conv5_block1_2_pad[0][0]
']

conv5_block1_2_bn (BatchNormal  (None, 7, 7, 512)   2048       ['conv5_block1_2_conv[0][0]
']
ization)

conv5_block1_2_relu (Activatio  (None, 7, 7, 512)   0          ['conv5_block1_2_bn[0][0]']
n)

conv5_block1_0_conv (Conv2D)   (None, 7, 7, 2048)   2099200    ['conv5_block1_preact_rel
u[0][0]'
                                ]

conv5_block1_3_conv (Conv2D)   (None, 7, 7, 2048)   1050624    ['conv5_block1_2_relu[0][0
]']

conv5_block1_out (Add)         (None, 7, 7, 2048)   0          ['conv5_block1_0_conv[0][0]',
                                                                'conv5_block1_3_conv[0][0]']

conv5_block2_preact_bn (BatchN  (None, 7, 7, 2048)  8192       ['conv5_block1_out[0][0]']

ormalization)

conv5_block2_preact_relu (Acti  (None, 7, 7, 2048)  0          ['conv5_block2_preact_bn[0][0]']

vation)

conv5_block2_1_conv (Conv2D)   (None, 7, 7, 512)    1048576    ['conv5_block2_preact_relu
[0][0]'
                                ]

conv5_block2_1_bn (BatchNormal  (None, 7, 7, 512)   2048       ['conv5_block2_1_conv[0][0]
']
ization)

conv5_block2_1_relu (Activatio  (None, 7, 7, 512)   0          ['conv5_block2_1_bn[0][0]']
n)

conv5_block2_2_pad (ZeroPaddin  (None, 9, 9, 512)   0          ['conv5_block2_1_relu[0][0]']
g2D)

conv5_block2_2_conv (Conv2D)   (None, 7, 7, 512)    2359296    ['conv5_block2_2_pad[0][0]
']

conv5_block2_2_bn (BatchNormal  (None, 7, 7, 512)   2048       ['conv5_block2_2_conv[0][0]
']
ization)

conv5_block2_2_relu (Activatio  (None, 7, 7, 512)   0          ['conv5_block2_2_bn[0][0]']
```

n)

conv5_block2_3_conv (Conv2D) (None, 7, 7, 2048) 1050624 ['conv5_block2_2_relu[0][0]']

conv5_block2_out (Add) (None, 7, 7, 2048) 0 ['conv5_block1_out[0][0]', 'conv5_block2_3_conv[0][0]']

conv5_block3_preact_bn (BatchN ormalization) (None, 7, 7, 2048) 8192 ['conv5_block2_out[0][0]']

conv5_block3_preact_relu (Acti vation) (None, 7, 7, 2048) 0 ['conv5_block3_preact_bn[0][0]']

conv5_block3_1_conv (Conv2D) (None, 7, 7, 512) 1048576 ['conv5_block3_preact_relu[0][0]']

conv5_block3_1_bn (BatchNormal ization) (None, 7, 7, 512) 2048 ['conv5_block3_1_conv[0][0]']

conv5_block3_1_relu (Activatio n) (None, 7, 7, 512) 0 ['conv5_block3_1_bn[0][0]']

conv5_block3_2_pad (ZeroPaddin g2D) (None, 9, 9, 512) 0 ['conv5_block3_1_relu[0][0]']

conv5_block3_2_conv (Conv2D) (None, 7, 7, 512) 2359296 ['conv5_block3_2_pad[0][0]']

conv5_block3_2_bn (BatchNormal ization) (None, 7, 7, 512) 2048 ['conv5_block3_2_conv[0][0]']

conv5_block3_2_relu (Activatio n) (None, 7, 7, 512) 0 ['conv5_block3_2_bn[0][0]']

conv5_block3_3_conv (Conv2D) (None, 7, 7, 2048) 1050624 ['conv5_block3_2_relu[0][0]']

conv5_block3_out (Add) (None, 7, 7, 2048) 0 ['conv5_block2_out[0][0]', 'conv5_block3_3_conv[0][0]']

post_bn (BatchNormalization) (None, 7, 7, 2048) 8192 ['conv5_block3_out[0][0]']

post_relu (Activation) (None, 7, 7, 2048) 0 ['post_bn[0][0]']

batch_normalization (BatchNorm alization) (None, 7, 7, 2048) 8192 ['post_relu[0][0]']

conv2d (Conv2D) (None, 5, 5, 32) 589856 ['batch_normalization[0][0]']

max_pooling2d_3 (MaxPooling2D) (None, 2, 2, 32) 0 ['conv2d[0][0]']

conv2d_1 (Conv2D) (None, 2, 2, 32) 9248 ['max_pooling2d_3[0][0]']

max_pooling2d_4 (MaxPooling2D) (None, 1, 1, 32) 0 ['conv2d_1[0][0]']

conv2d_2 (Conv2D) (None, 1, 1, 64) 18496 ['max_pooling2d_4[0][0]']

max_pooling2d_5 (MaxPooling2D) (None, 1, 1, 64) 0 ['conv2d_2[0][0]']

dropout (Dropout) (None, 1, 1, 64) 0 ['max_pooling2d_5[0][0]']

flatten (Flatten) (None, 64) 0 ['dropout[0][0]']

dense (Dense) (None, 1024) 66560 ['flatten[0][0]']

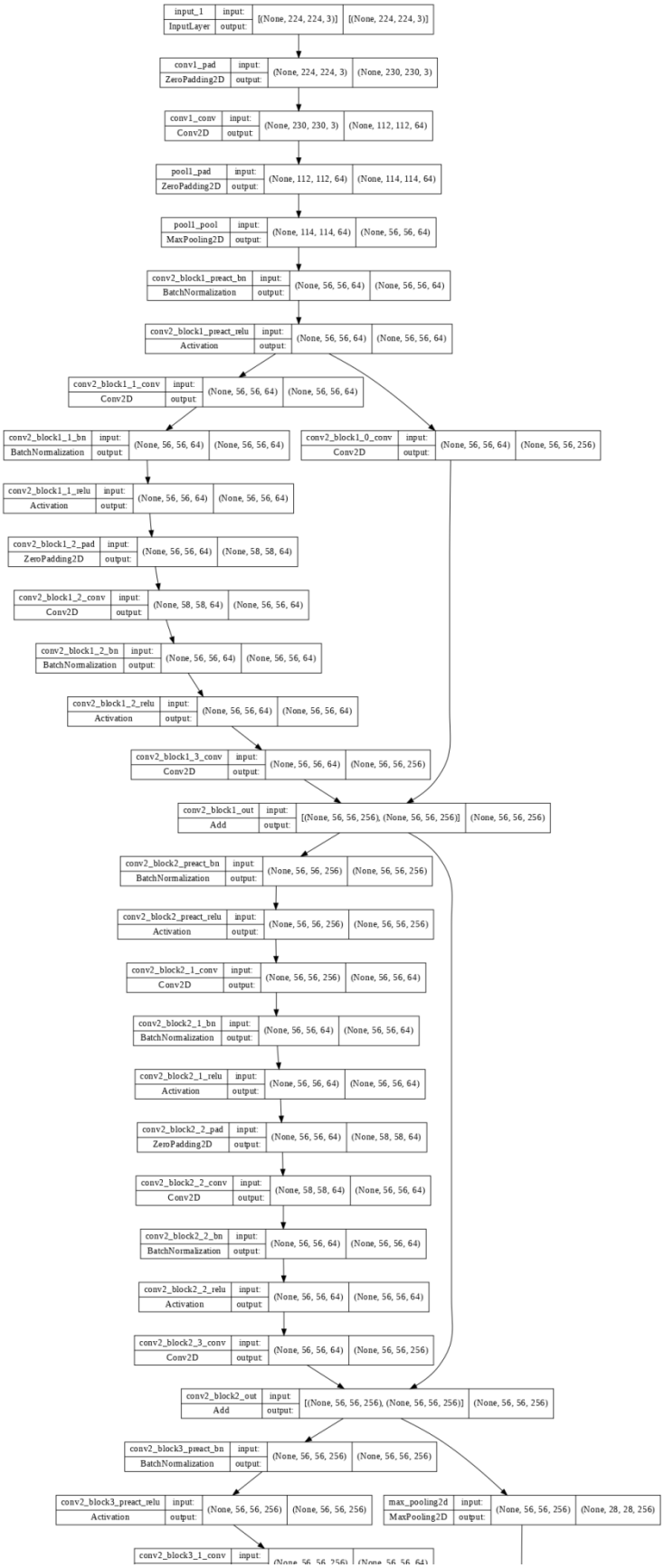
dropout_1 (Dropout) (None, 1024) 0 ['dense[0][0]']

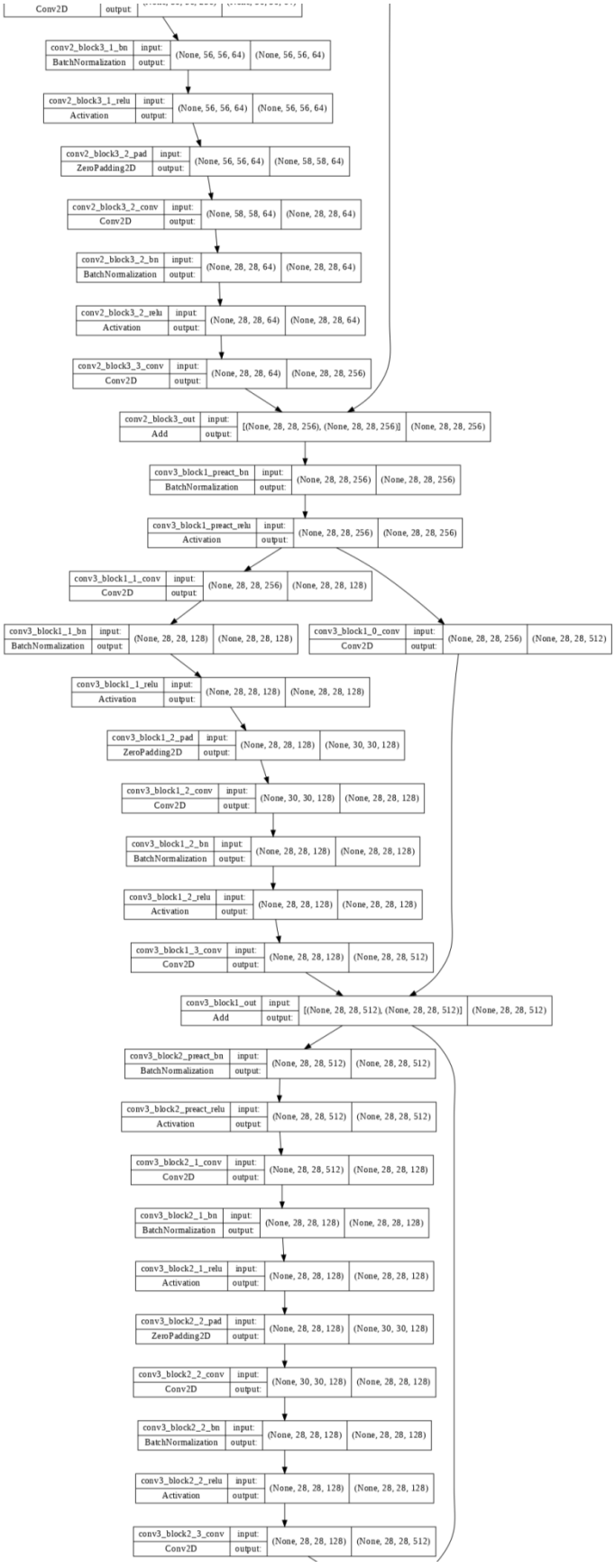
dense_1 (Dense) (None, 4) 4100 ['dropout_1[0][0]']

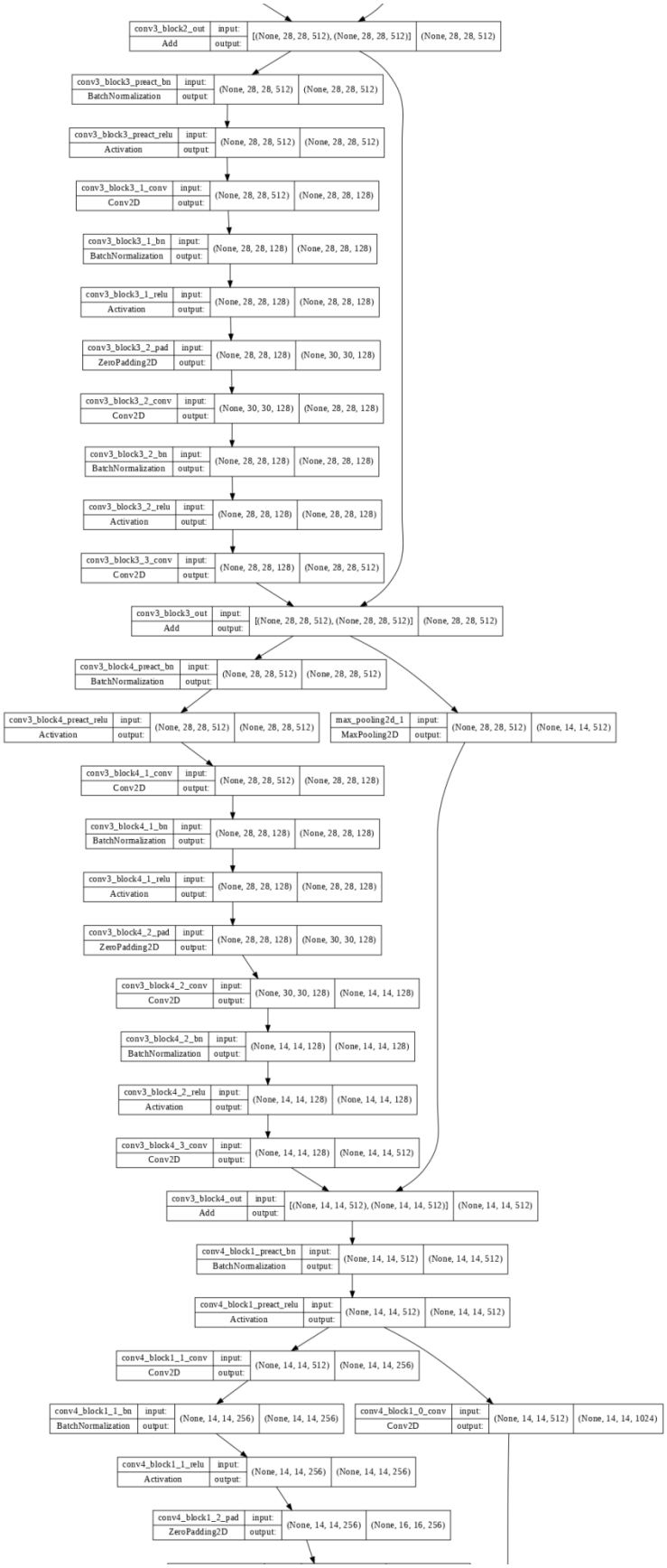
==========

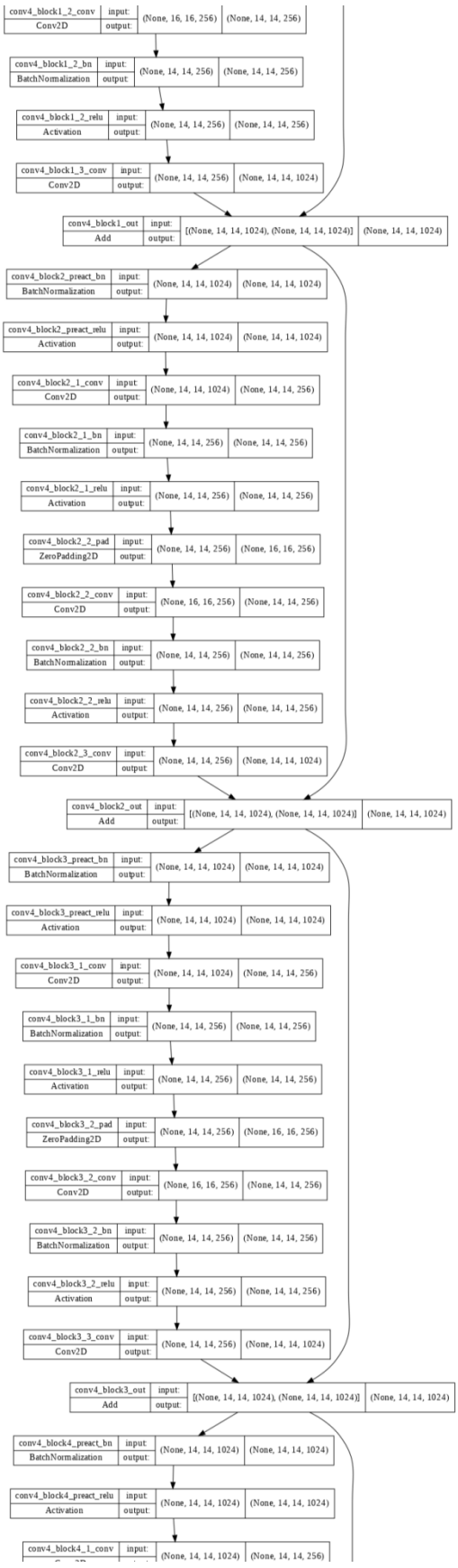
Total params: 43,323,012
Trainable params: 692,356
Non-trainable params: 42,630,656

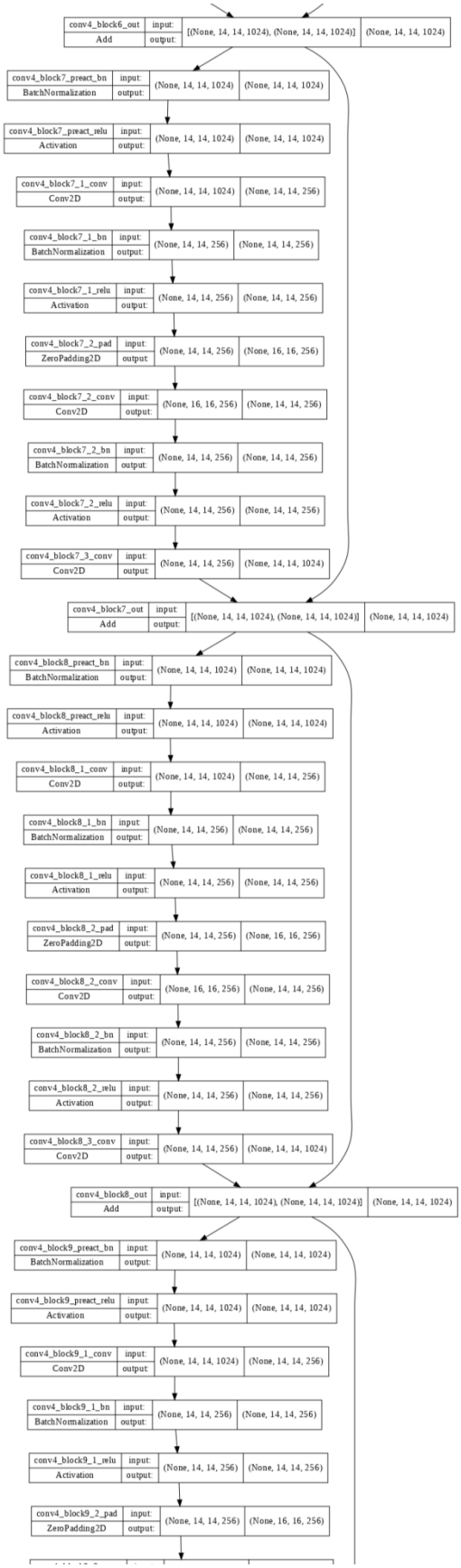
_____

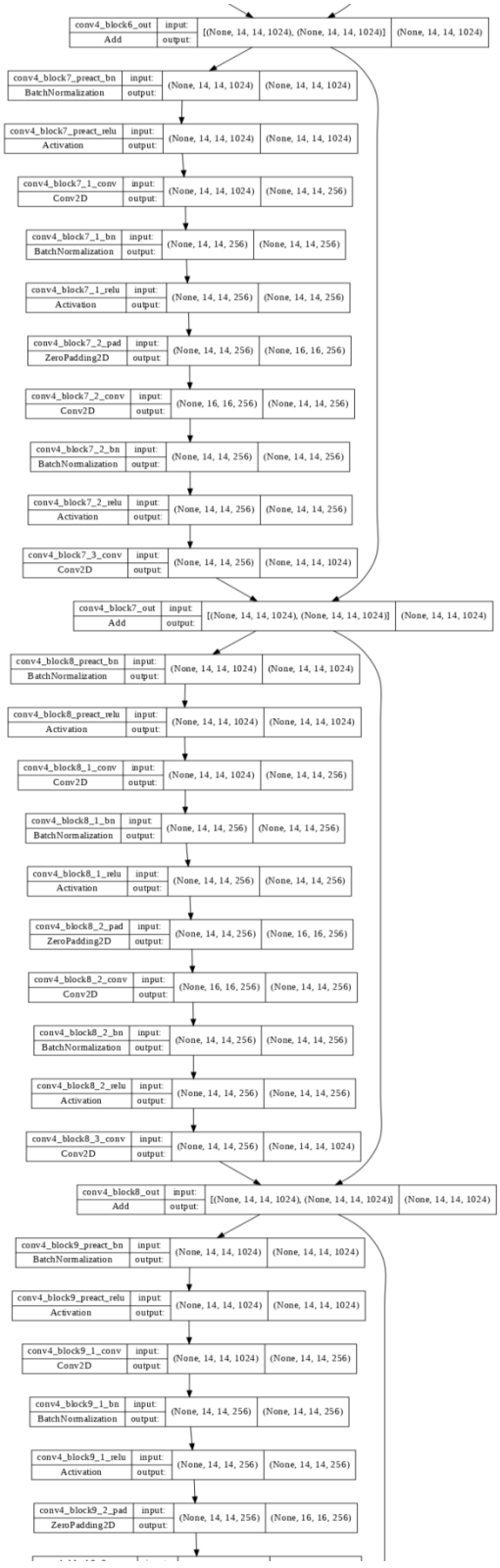

input_1 | InputLayer | input: | output: | [(None, 224, 224, 3)] | [(None, 224, 224, 3)]
conv1_pad | ZeroPadding2D | input: | output: | (None, 224, 224, 3) | (None, 230, 230, 3)
conv1_conv | Conv2D | input: | output: | (None, 230, 230, 3) | (None, 112, 112, 64)
pool1_pad | ZeroPadding2D | input: | output: | (None, 112, 112, 64) | (None, 114, 114, 64)
pool1_pool | MaxPooling2D | input: | output: | (None, 114, 114, 64) | (None, 56, 56, 64)
conv2_block1_preact_bn | BatchNormalization | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block1_preact_relu | Activation | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block1_1_conv | Conv2D | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block1_1_bn | BatchNormalization | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block1_0_conv | Conv2D | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 256)
conv2_block1_1_relu | Activation | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block1_2_pad | ZeroPadding2D | input: | output: | (None, 56, 56, 64) | (None, 58, 58, 64)
conv2_block1_2_conv | Conv2D | input: | output: | (None, 58, 58, 64) | (None, 56, 56, 64)
conv2_block1_2_bn | BatchNormalization | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block1_2_relu | Activation | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block1_3_conv | Conv2D | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 256)
conv2_block1_out | Add | input: | output: | [(None, 56, 56, 256), (None, 56, 56, 256)] | (None, 56, 56, 256)
conv2_block2_preact_bn | BatchNormalization | input: | output: | (None, 56, 56, 256) | (None, 56, 56, 256)
conv2_block2_preact_relu | Activation | input: | output: | (None, 56, 56, 256) | (None, 56, 56, 256)
conv2_block2_1_conv | Conv2D | input: | output: | (None, 56, 56, 256) | (None, 56, 56, 64)
conv2_block2_1_bn | BatchNormalization | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block2_1_relu | Activation | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block2_2_pad | ZeroPadding2D | input: | output: | (None, 56, 56, 64) | (None, 58, 58, 64)
conv2_block2_2_conv | Conv2D | input: | output: | (None, 58, 58, 64) | (None, 56, 56, 64)
conv2_block2_2_bn | BatchNormalization | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block2_2_relu | Activation | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 64)
conv2_block2_3_conv | Conv2D | input: | output: | (None, 56, 56, 64) | (None, 56, 56, 256)
conv2_block2_out | Add | input: | output: | [(None, 56, 56, 256), (None, 56, 56, 256)] | (None, 56, 56, 256)
conv2_block3_preact_bn | BatchNormalization | input: | output: | (None, 56, 56, 256) | (None, 56, 56, 256)
conv2_block3_preact_relu | Activation | input: | output: | (None, 56, 56, 256) | (None, 56, 56, 256)
max_pooling2d | MaxPooling2D | input: | output: | (None, 56, 56, 256) | (None, 28, 28, 256)
conv2_block3_1_conv | input:

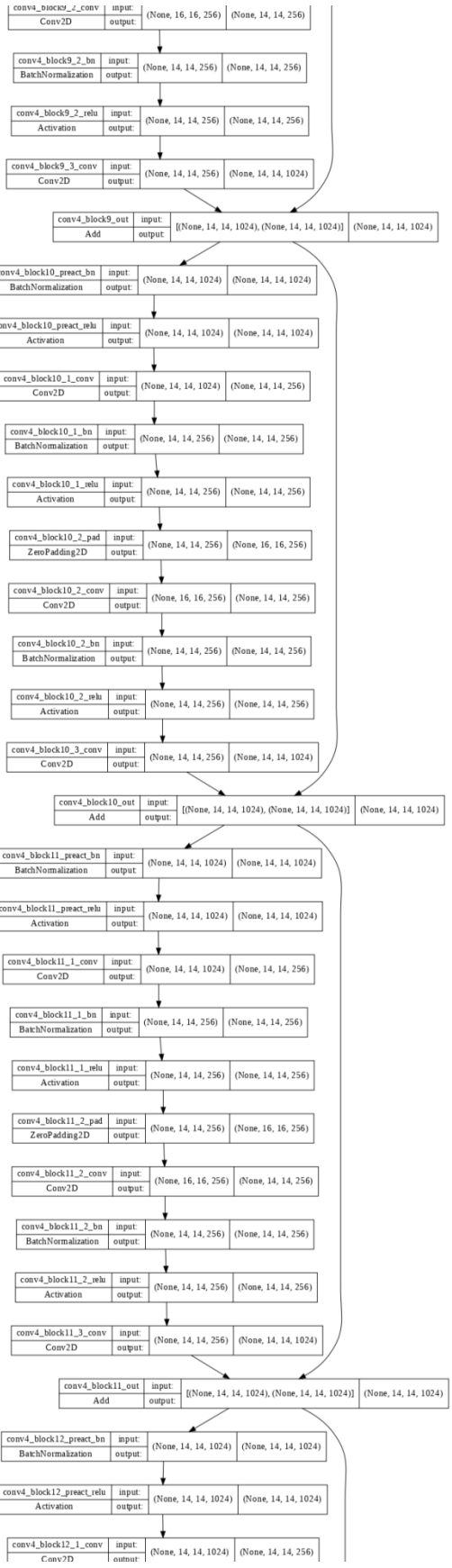

Conv2D output:
conv2_block3_1_bn input: BatchNormalization output: (None, 56, 56, 64) (None, 56, 56, 64)
conv2_block3_1_relu input: Activation output: (None, 56, 56, 64) (None, 56, 56, 64)
conv2_block3_2_pad input: ZeroPadding2D output: (None, 56, 56, 64) (None, 58, 58, 64)
conv2_block3_2_conv input: Conv2D output: (None, 58, 58, 64) (None, 28, 28, 64)
conv2_block3_2_bn input: BatchNormalization output: (None, 28, 28, 64) (None, 28, 28, 64)
conv2_block3_2_relu input: Activation output: (None, 28, 28, 64) (None, 28, 28, 64)
conv2_block3_3_conv input: Conv2D output: (None, 28, 28, 64) (None, 28, 28, 256)
conv2_block3_out input: Add output: [(None, 28, 28, 256), (None, 28, 28, 256)] (None, 28, 28, 256)
conv3_block1_preact_bn input: BatchNormalization output: (None, 28, 28, 256) (None, 28, 28, 256)
conv3_block1_preact_relu input: Activation output: (None, 28, 28, 256) (None, 28, 28, 256)
conv3_block1_1_conv input: Conv2D output: (None, 28, 28, 256) (None, 28, 28, 128)
conv3_block1_1_bn input: BatchNormalization output: (None, 28, 28, 128) (None, 28, 28, 128)
conv3_block1_0_conv input: Conv2D output: (None, 28, 28, 256) (None, 28, 28, 512)
conv3_block1_1_relu input: Activation output: (None, 28, 28, 128) (None, 28, 28, 128)
conv3_block1_2_pad input: ZeroPadding2D output: (None, 28, 28, 128) (None, 30, 30, 128)
conv3_block1_2_conv input: Conv2D output: (None, 30, 30, 128) (None, 28, 28, 128)
conv3_block1_2_bn input: BatchNormalization output: (None, 28, 28, 128) (None, 28, 28, 128)
conv3_block1_2_relu input: Activation output: (None, 28, 28, 128) (None, 28, 28, 128)
conv3_block1_3_conv input: Conv2D output: (None, 28, 28, 128) (None, 28, 28, 512)
conv3_block1_out input: Add output: [(None, 28, 28, 512), (None, 28, 28, 512)] (None, 28, 28, 512)
conv3_block2_preact_bn input: BatchNormalization output: (None, 28, 28, 512) (None, 28, 28, 512)
conv3_block2_preact_relu input: Activation output: (None, 28, 28, 512) (None, 28, 28, 512)
conv3_block2_1_conv input: Conv2D output: (None, 28, 28, 512) (None, 28, 28, 128)
conv3_block2_1_bn input: BatchNormalization output: (None, 28, 28, 128) (None, 28, 28, 128)
conv3_block2_1_relu input: Activation output: (None, 28, 28, 128) (None, 28, 28, 128)
conv3_block2_2_pad input: ZeroPadding2D output: (None, 28, 28, 128) (None, 30, 30, 128)
conv3_block2_2_conv input: Conv2D output: (None, 30, 30, 128) (None, 28, 28, 128)
conv3_block2_2_bn input: BatchNormalization output: (None, 28, 28, 128) (None, 28, 28, 128)
conv3_block2_2_relu input: Activation output: (None, 28, 28, 128) (None, 28, 28, 128)
conv3_block2_3_conv input: Conv2D output: (None, 28, 28, 128) (None, 28, 28, 512)

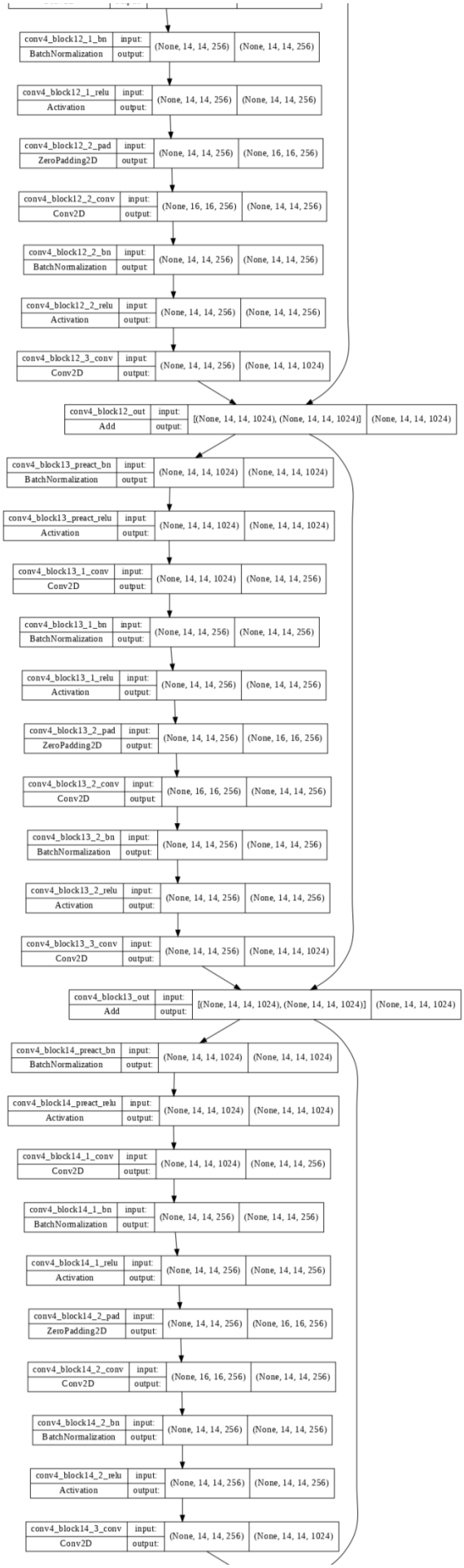

conv3_block2_out | Add | input: | output: | [(None, 28, 28, 512), (None, 28, 28, 512)] | (None, 28, 28, 512)
conv3_block3_preact_bn | BatchNormalization | input: | output: | (None, 28, 28, 512) | (None, 28, 28, 512)
conv3_block3_preact_relu | Activation | input: | output: | (None, 28, 28, 512) | (None, 28, 28, 512)
conv3_block3_1_conv | Conv2D | input: | output: | (None, 28, 28, 512) | (None, 28, 28, 128)
conv3_block3_1_bn | BatchNormalization | input: | output: | (None, 28, 28, 128) | (None, 28, 28, 128)
conv3_block3_1_relu | Activation | input: | output: | (None, 28, 28, 128) | (None, 28, 28, 128)
conv3_block3_2_pad | ZeroPadding2D | input: | output: | (None, 28, 28, 128) | (None, 30, 30, 128)
conv3_block3_2_conv | Conv2D | input: | output: | (None, 30, 30, 128) | (None, 28, 28, 128)
conv3_block3_2_bn | BatchNormalization | input: | output: | (None, 28, 28, 128) | (None, 28, 28, 128)
conv3_block3_2_relu | Activation | input: | output: | (None, 28, 28, 128) | (None, 28, 28, 128)
conv3_block3_3_conv | Conv2D | input: | output: | (None, 28, 28, 128) | (None, 28, 28, 512)
conv3_block3_out | Add | input: | output: | [(None, 28, 28, 512), (None, 28, 28, 512)] | (None, 28, 28, 512)
conv3_block4_preact_bn | BatchNormalization | input: | output: | (None, 28, 28, 512) | (None, 28, 28, 512)
conv3_block4_preact_relu | Activation | input: | output: | (None, 28, 28, 512) | (None, 28, 28, 512)
max_pooling2d_1 | MaxPooling2D | input: | output: | (None, 28, 28, 512) | (None, 14, 14, 512)
conv3_block4_1_conv | Conv2D | input: | output: | (None, 28, 28, 512) | (None, 28, 28, 128)
conv3_block4_1_bn | BatchNormalization | input: | output: | (None, 28, 28, 128) | (None, 28, 28, 128)
conv3_block4_1_relu | Activation | input: | output: | (None, 28, 28, 128) | (None, 28, 28, 128)
conv3_block4_2_pad | ZeroPadding2D | input: | output: | (None, 28, 28, 128) | (None, 30, 30, 128)
conv3_block4_2_conv | Conv2D | input: | output: | (None, 30, 30, 128) | (None, 14, 14, 128)
conv3_block4_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 128) | (None, 14, 14, 128)
conv3_block4_2_relu | Activation | input: | output: | (None, 14, 14, 128) | (None, 14, 14, 128)
conv3_block4_3_conv | Conv2D | input: | output: | (None, 14, 14, 128) | (None, 14, 14, 512)
conv3_block4_out | Add | input: | output: | [(None, 14, 14, 512), (None, 14, 14, 512)] | (None, 14, 14, 512)
conv4_block1_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 512) | (None, 14, 14, 512)
conv4_block1_preact_relu | Activation | input: | output: | (None, 14, 14, 512) | (None, 14, 14, 512)
conv4_block1_1_conv | Conv2D | input: | output: | (None, 14, 14, 512) | (None, 14, 14, 256)
conv4_block1_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block1_0_conv | Conv2D | input: | output: | (None, 14, 14, 512) | (None, 14, 14, 1024)
conv4_block1_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block1_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)

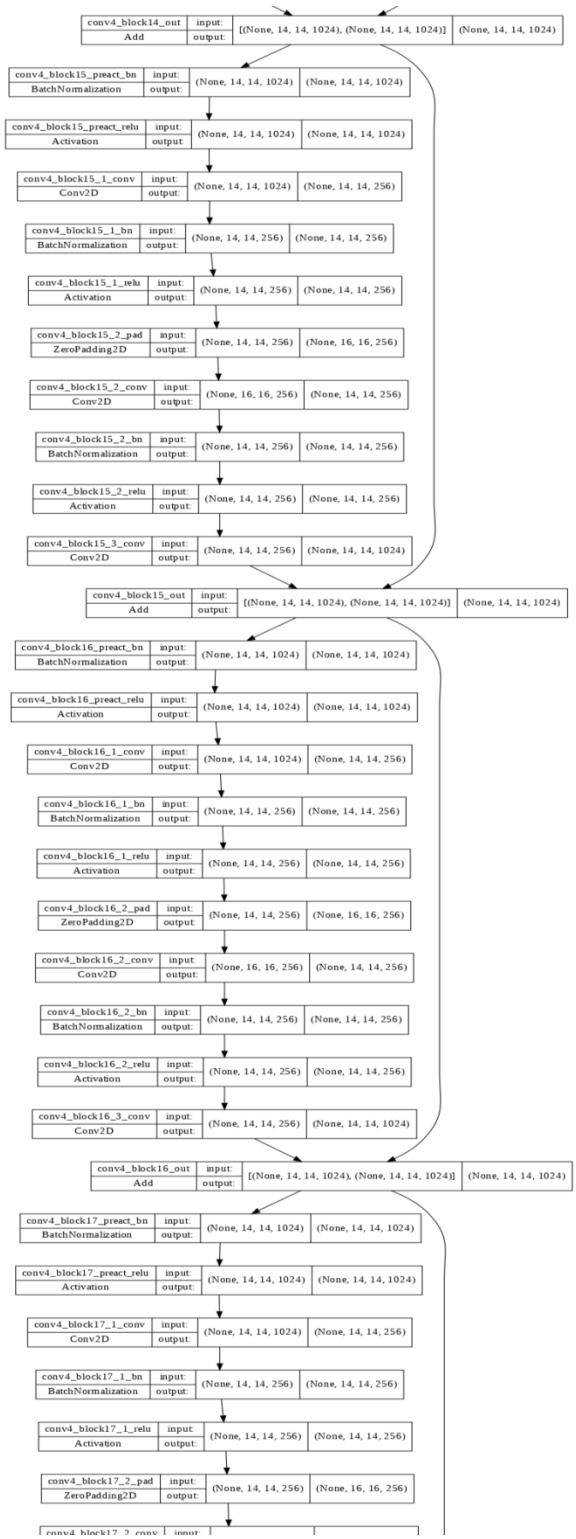

conv4_block1_2_conv | Conv2D | input: (None, 16, 16, 256) | output: (None, 14, 14, 256)
conv4_block1_2_bn | BatchNormalization | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block1_2_relu | Activation | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block1_3_conv | Conv2D | input: (None, 14, 14, 256) | output: (None, 14, 14, 1024)
conv4_block1_out | Add | input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] | output: (None, 14, 14, 1024)
conv4_block2_preact_bn | BatchNormalization | input: (None, 14, 14, 1024) | output: (None, 14, 14, 1024)
conv4_block2_preact_relu | Activation | input: (None, 14, 14, 1024) | output: (None, 14, 14, 1024)
conv4_block2_1_conv | Conv2D | input: (None, 14, 14, 1024) | output: (None, 14, 14, 256)
conv4_block2_1_bn | BatchNormalization | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block2_1_relu | Activation | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block2_2_pad | ZeroPadding2D | input: (None, 14, 14, 256) | output: (None, 16, 16, 256)
conv4_block2_2_conv | Conv2D | input: (None, 16, 16, 256) | output: (None, 14, 14, 256)
conv4_block2_2_bn | BatchNormalization | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block2_2_relu | Activation | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block2_3_conv | Conv2D | input: (None, 14, 14, 256) | output: (None, 14, 14, 1024)
conv4_block2_out | Add | input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] | output: (None, 14, 14, 1024)
conv4_block3_preact_bn | BatchNormalization | input: (None, 14, 14, 1024) | output: (None, 14, 14, 1024)
conv4_block3_preact_relu | Activation | input: (None, 14, 14, 1024) | output: (None, 14, 14, 1024)
conv4_block3_1_conv | Conv2D | input: (None, 14, 14, 1024) | output: (None, 14, 14, 256)
conv4_block3_1_bn | BatchNormalization | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block3_1_relu | Activation | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block3_2_pad | ZeroPadding2D | input: (None, 14, 14, 256) | output: (None, 16, 16, 256)
conv4_block3_2_conv | Conv2D | input: (None, 16, 16, 256) | output: (None, 14, 14, 256)
conv4_block3_2_bn | BatchNormalization | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block3_2_relu | Activation | input: (None, 14, 14, 256) | output: (None, 14, 14, 256)
conv4_block3_3_conv | Conv2D | input: (None, 14, 14, 256) | output: (None, 14, 14, 1024)
conv4_block3_out | Add | input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] | output: (None, 14, 14, 1024)
conv4_block4_preact_bn | BatchNormalization | input: (None, 14, 14, 1024) | output: (None, 14, 14, 1024)
conv4_block4_preact_relu | Activation | input: (None, 14, 14, 1024) | output: (None, 14, 14, 1024)
conv4_block4_1_conv | input: (None, 14, 14, 1024) | (None, 14, 14, 256)

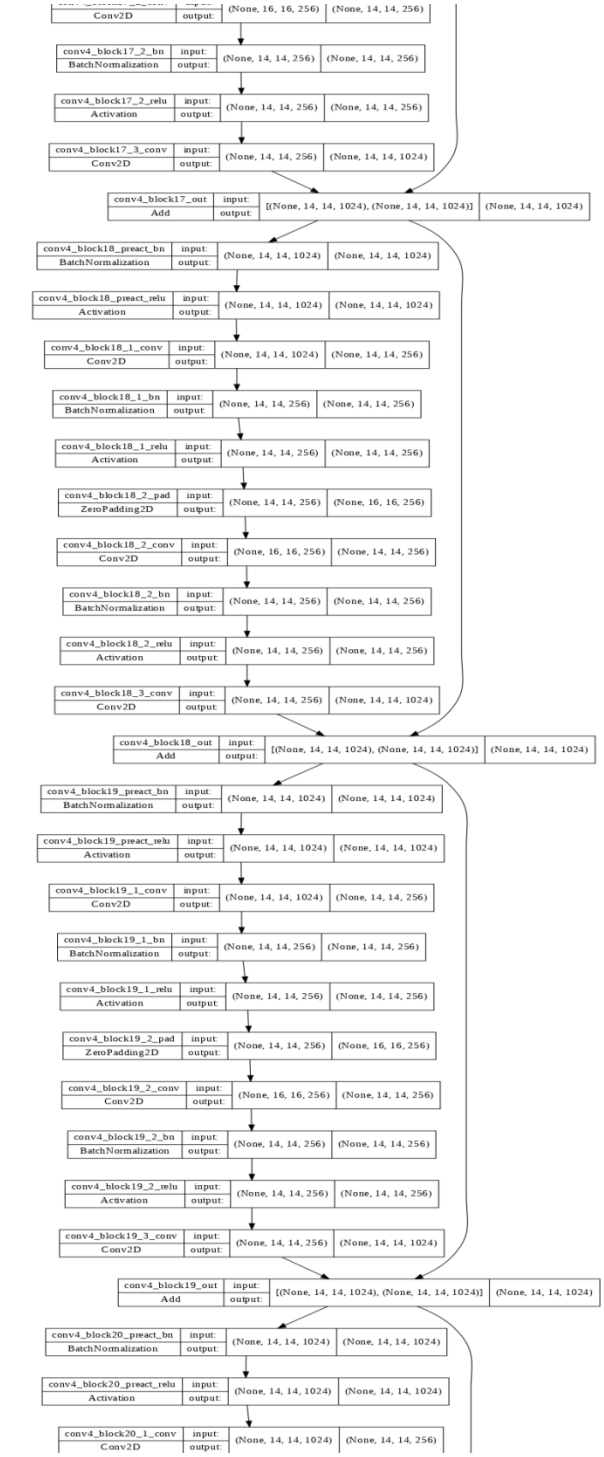

conv4_block6_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block7_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block7_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block7_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block7_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block7_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block7_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block7_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block7_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block7_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block7_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block7_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block8_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block8_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block8_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block8_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block8_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block8_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block8_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block8_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block8_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block8_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block8_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block9_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block9_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block9_1_conv | Conv2D | input: | output | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block9_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block9_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block9_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)

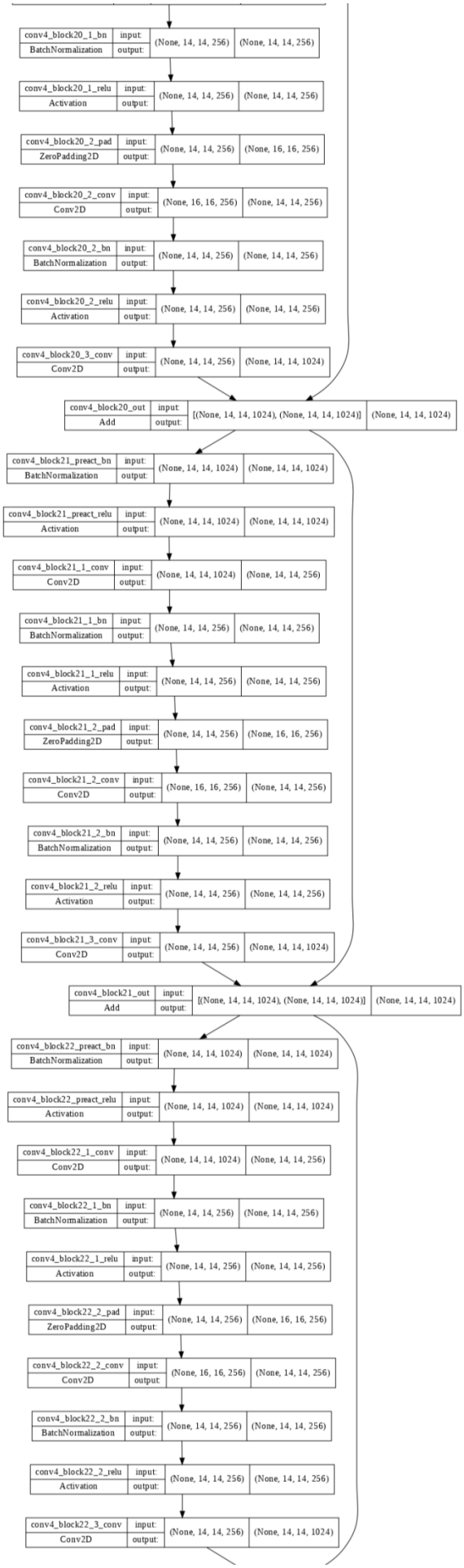

conv4_block6_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block7_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block7_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block7_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block7_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block7_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block7_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block7_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block7_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block7_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block7_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block7_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block8_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block8_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block8_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block8_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block8_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block8_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block8_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block8_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block8_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block8_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block8_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block9_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block9_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block9_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block9_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block9_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block9_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)

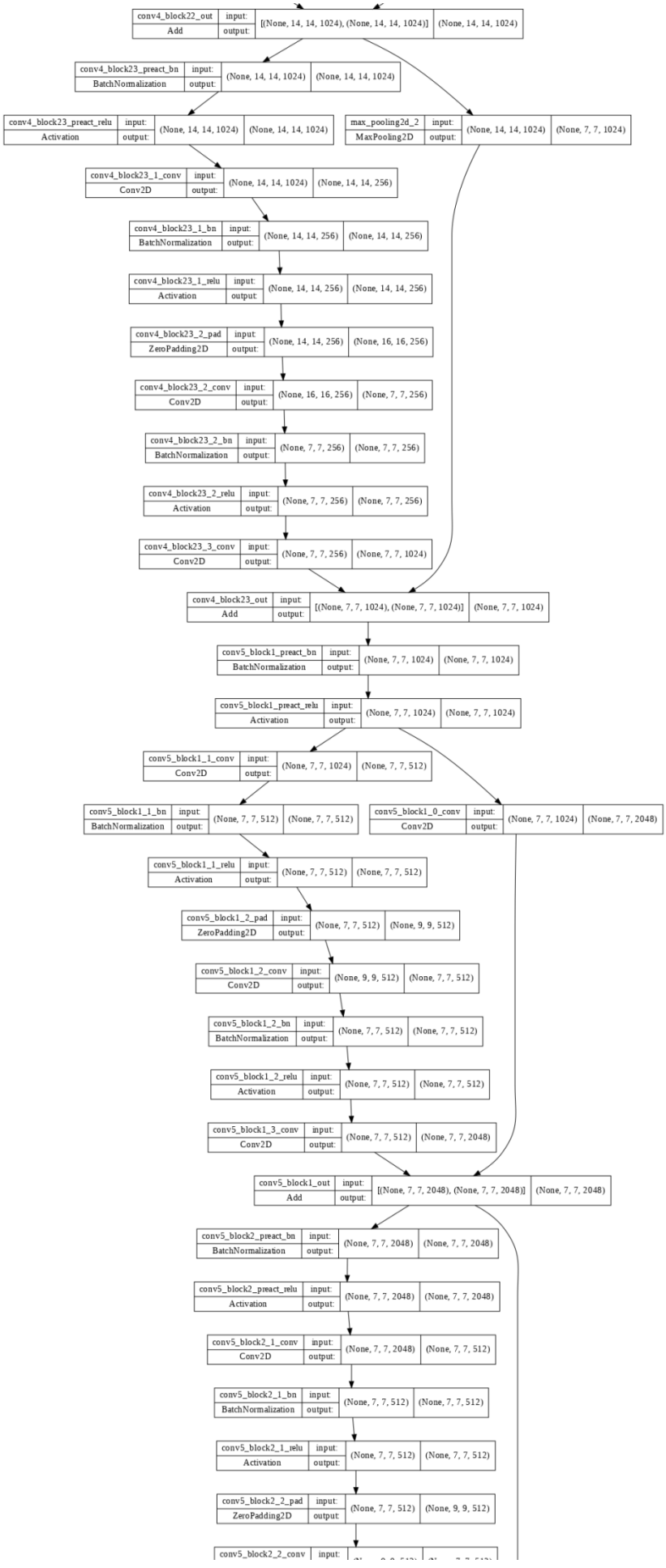

conv4_block9_2_conv | input: (None, 16, 16, 256) | Conv2D | output: (None, 14, 14, 256)
conv4_block9_2_bn | input: (None, 14, 14, 256) | BatchNormalization | output: (None, 14, 14, 256)
conv4_block9_2_relu | input: (None, 14, 14, 256) | Activation | output: (None, 14, 14, 256)
conv4_block9_3_conv | input: (None, 14, 14, 256) | Conv2D | output: (None, 14, 14, 1024)
conv4_block9_out | input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] | Add | output: (None, 14, 14, 1024)
conv4_block10_preact_bn | input: (None, 14, 14, 1024) | BatchNormalization | output: (None, 14, 14, 1024)
conv4_block10_preact_relu | input: (None, 14, 14, 1024) | Activation | output: (None, 14, 14, 1024)
conv4_block10_1_conv | input: (None, 14, 14, 1024) | Conv2D | output: (None, 14, 14, 256)
conv4_block10_1_bn | input: (None, 14, 14, 256) | BatchNormalization | output: (None, 14, 14, 256)
conv4_block10_1_relu | input: (None, 14, 14, 256) | Activation | output: (None, 14, 14, 256)
conv4_block10_2_pad | input: (None, 14, 14, 256) | ZeroPadding2D | output: (None, 16, 16, 256)
conv4_block10_2_conv | input: (None, 16, 16, 256) | Conv2D | output: (None, 14, 14, 256)
conv4_block10_2_bn | input: (None, 14, 14, 256) | BatchNormalization | output: (None, 14, 14, 256)
conv4_block10_2_relu | input: (None, 14, 14, 256) | Activation | output: (None, 14, 14, 256)
conv4_block10_3_conv | input: (None, 14, 14, 256) | Conv2D | output: (None, 14, 14, 1024)
conv4_block10_out | input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] | Add | output: (None, 14, 14, 1024)
conv4_block11_preact_bn | input: (None, 14, 14, 1024) | BatchNormalization | output: (None, 14, 14, 1024)
conv4_block11_preact_relu | input: (None, 14, 14, 1024) | Activation | output: (None, 14, 14, 1024)
conv4_block11_1_conv | input: (None, 14, 14, 1024) | Conv2D | output: (None, 14, 14, 256)
conv4_block11_1_bn | input: (None, 14, 14, 256) | BatchNormalization | output: (None, 14, 14, 256)
conv4_block11_1_relu | input: (None, 14, 14, 256) | Activation | output: (None, 14, 14, 256)
conv4_block11_2_pad | input: (None, 14, 14, 256) | ZeroPadding2D | output: (None, 16, 16, 256)
conv4_block11_2_conv | input: (None, 16, 16, 256) | Conv2D | output: (None, 14, 14, 256)
conv4_block11_2_bn | input: (None, 14, 14, 256) | BatchNormalization | output: (None, 14, 14, 256)
conv4_block11_2_relu | input: (None, 14, 14, 256) | Activation | output: (None, 14, 14, 256)
conv4_block11_3_conv | input: (None, 14, 14, 256) | Conv2D | output: (None, 14, 14, 1024)
conv4_block11_out | input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] | Add | output: (None, 14, 14, 1024)
conv4_block12_preact_bn | input: (None, 14, 14, 1024) | BatchNormalization | output: (None, 14, 14, 1024)
conv4_block12_preact_relu | input: (None, 14, 14, 1024) | Activation | output: (None, 14, 14, 1024)
conv4_block12_1_conv | input: (None, 14, 14, 1024) | Conv2D | output: (None, 14, 14, 256)

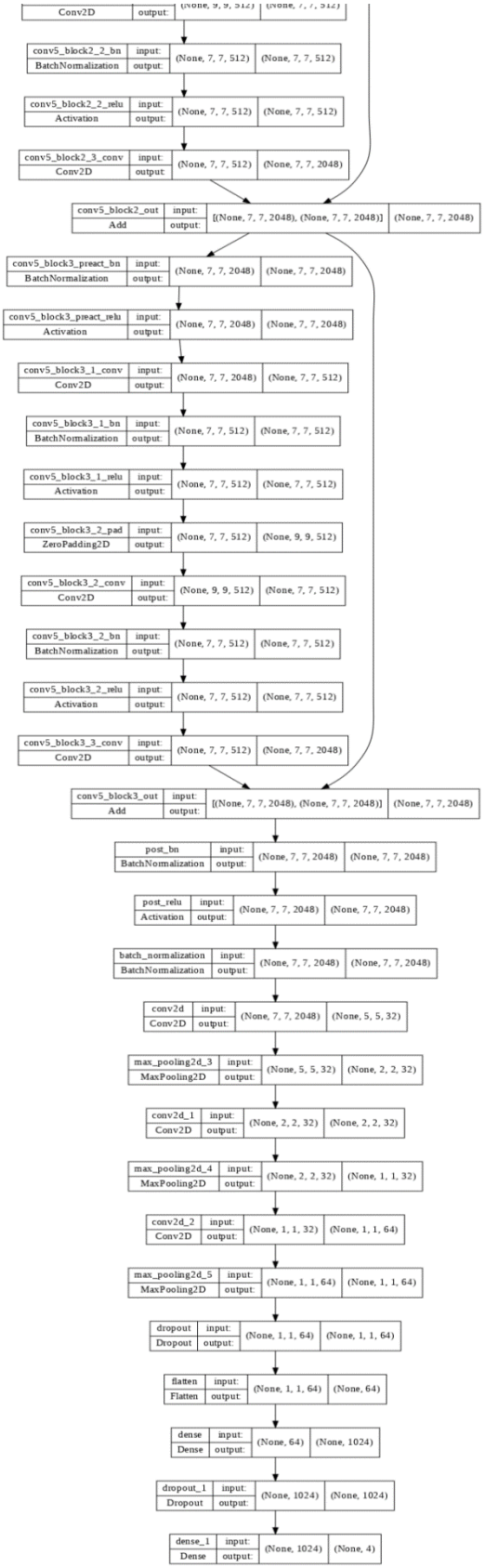

conv4_block12_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block12_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block12_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block12_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block12_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block12_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block12_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block12_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block13_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block13_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block13_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block13_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block13_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block13_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block13_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block13_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block13_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block13_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block13_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block14_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block14_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block14_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block14_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block14_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block14_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block14_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block14_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block14_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block14_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)


conv4_block14_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block15_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block15_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block15_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block15_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block15_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block15_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block15_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block15_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block15_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block15_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block15_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block16_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block16_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block16_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block16_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block16_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block16_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block16_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block16_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block16_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block16_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block16_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block17_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block17_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block17_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block17_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block17_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block17_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)


Conv2D input: (None, 16, 16, 256) output: (None, 14, 14, 256)
conv4_block17_2_bn BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block17_2_relu Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block17_3_conv Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
conv4_block17_out Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
conv4_block18_preact_bn BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
conv4_block18_preact_relu Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
conv4_block18_1_conv Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
conv4_block18_1_bn BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block18_1_relu Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block18_2_pad ZeroPadding2D input: (None, 14, 14, 256) output: (None, 16, 16, 256)
conv4_block18_2_conv Conv2D input: (None, 16, 16, 256) output: (None, 14, 14, 256)
conv4_block18_2_bn BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block18_2_relu Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block18_3_conv Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
conv4_block18_out Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
conv4_block19_preact_bn BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
conv4_block19_preact_relu Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
conv4_block19_1_conv Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
conv4_block19_1_bn BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block19_1_relu Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block19_2_pad ZeroPadding2D input: (None, 14, 14, 256) output: (None, 16, 16, 256)
conv4_block19_2_conv Conv2D input: (None, 16, 16, 256) output: (None, 14, 14, 256)
conv4_block19_2_bn BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block19_2_relu Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
conv4_block19_3_conv Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
conv4_block19_out Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
conv4_block20_preact_bn BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
conv4_block20_preact_relu Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
conv4_block20_1_conv Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)


conv4_block20_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block20_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block20_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block20_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block20_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block20_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block20_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block20_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block21_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block21_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block21_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block21_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block21_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block21_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block21_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block21_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block21_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block21_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)
conv4_block21_out | Add | input: | output: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
conv4_block22_preact_bn | BatchNormalization | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block22_preact_relu | Activation | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
conv4_block22_1_conv | Conv2D | input: | output: | (None, 14, 14, 1024) | (None, 14, 14, 256)
conv4_block22_1_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block22_1_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block22_2_pad | ZeroPadding2D | input: | output: | (None, 14, 14, 256) | (None, 16, 16, 256)
conv4_block22_2_conv | Conv2D | input: | output: | (None, 16, 16, 256) | (None, 14, 14, 256)
conv4_block22_2_bn | BatchNormalization | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block22_2_relu | Activation | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 256)
conv4_block22_3_conv | Conv2D | input: | output: | (None, 14, 14, 256) | (None, 14, 14, 1024)


conv4_block22_out | input: | [(None, 14, 14, 1024), (None, 14, 14, 1024)] | (None, 14, 14, 1024)
Add | output:
conv4_block23_preact_bn | input: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
BatchNormalization | output:
conv4_block23_preact_relu | input: | (None, 14, 14, 1024) | (None, 14, 14, 1024)
Activation | output:
max_pooling2d_2 | input: | (None, 14, 14, 1024) | (None, 7, 7, 1024)
MaxPooling2D | output:
conv4_block23_1_conv | input: | (None, 14, 14, 1024) | (None, 14, 14, 256)
Conv2D | output:
conv4_block23_1_bn | input: | (None, 14, 14, 256) | (None, 14, 14, 256)
BatchNormalization | output:
conv4_block23_1_relu | input: | (None, 14, 14, 256) | (None, 14, 14, 256)
Activation | output:
conv4_block23_2_pad | input: | (None, 14, 14, 256) | (None, 16, 16, 256)
ZeroPadding2D | output:
conv4_block23_2_conv | input: | (None, 16, 16, 256) | (None, 7, 7, 256)
Conv2D | output:
conv4_block23_2_bn | input: | (None, 7, 7, 256) | (None, 7, 7, 256)
BatchNormalization | output:
conv4_block23_2_relu | input: | (None, 7, 7, 256) | (None, 7, 7, 256)
Activation | output:
conv4_block23_3_conv | input: | (None, 7, 7, 256) | (None, 7, 7, 1024)
Conv2D | output:
conv4_block23_out | input: | [(None, 7, 7, 1024), (None, 7, 7, 1024)] | (None, 7, 7, 1024)
Add | output:
conv5_block1_preact_bn | input: | (None, 7, 7, 1024) | (None, 7, 7, 1024)
BatchNormalization | output:
conv5_block1_preact_relu | input: | (None, 7, 7, 1024) | (None, 7, 7, 1024)
Activation | output:
conv5_block1_1_conv | input: | (None, 7, 7, 1024) | (None, 7, 7, 512)
Conv2D | output:
conv5_block1_1_bn | input: | (None, 7, 7, 512) | (None, 7, 7, 512)
BatchNormalization | output:
conv5_block1_0_conv | input: | (None, 7, 7, 1024) | (None, 7, 7, 2048)
Conv2D | output:
conv5_block1_1_relu | input: | (None, 7, 7, 512) | (None, 7, 7, 512)
Activation | output:
conv5_block1_2_pad | input: | (None, 7, 7, 512) | (None, 9, 9, 512)
ZeroPadding2D | output:
conv5_block1_2_conv | input: | (None, 9, 9, 512) | (None, 7, 7, 512)
Conv2D | output:
conv5_block1_2_bn | input: | (None, 7, 7, 512) | (None, 7, 7, 512)
BatchNormalization | output:
conv5_block1_2_relu | input: | (None, 7, 7, 512) | (None, 7, 7, 512)
Activation | output:
conv5_block1_3_conv | input: | (None, 7, 7, 512) | (None, 7, 7, 2048)
Conv2D | output:
conv5_block1_out | input: | [(None, 7, 7, 2048), (None, 7, 7, 2048)] | (None, 7, 7, 2048)
Add | output:
conv5_block2_preact_bn | input: | (None, 7, 7, 2048) | (None, 7, 7, 2048)
BatchNormalization | output:
conv5_block2_preact_relu | input: | (None, 7, 7, 2048) | (None, 7, 7, 2048)
Activation | output:
conv5_block2_1_conv | input: | (None, 7, 7, 2048) | (None, 7, 7, 512)
Conv2D | output:
conv5_block2_1_bn | input: | (None, 7, 7, 512) | (None, 7, 7, 512)
BatchNormalization | output:
conv5_block2_1_relu | input: | (None, 7, 7, 512) | (None, 7, 7, 512)
Activation | output:
conv5_block2_2_pad | input: | (None, 7, 7, 512) | (None, 9, 9, 512)
ZeroPadding2D | output:
conv5_block2_2_conv | input:


Conv2D output: (None, 9, 9, 512) (None, 7, 7, 512)
conv5_block2_2_bn BatchNormalization input: output: (None, 7, 7, 512) (None, 7, 7, 512)
conv5_block2_2_relu Activation input: output: (None, 7, 7, 512) (None, 7, 7, 512)
conv5_block2_3_conv Conv2D input: output: (None, 7, 7, 512) (None, 7, 7, 2048)
conv5_block2_out Add input: output: [(None, 7, 7, 2048), (None, 7, 7, 2048)] (None, 7, 7, 2048)
conv5_block3_preact_bn BatchNormalization input: output: (None, 7, 7, 2048) (None, 7, 7, 2048)
conv5_block3_preact_relu Activation input: output: (None, 7, 7, 2048) (None, 7, 7, 2048)
conv5_block3_1_conv Conv2D input: output: (None, 7, 7, 2048) (None, 7, 7, 512)
conv5_block3_1_bn BatchNormalization input: output: (None, 7, 7, 512) (None, 7, 7, 512)
conv5_block3_1_relu Activation input: output: (None, 7, 7, 512) (None, 7, 7, 512)
conv5_block3_2_pad ZeroPadding2D input: output: (None, 7, 7, 512) (None, 9, 9, 512)
conv5_block3_2_conv Conv2D input: output: (None, 9, 9, 512) (None, 7, 7, 512)
conv5_block3_2_bn BatchNormalization input: output: (None, 7, 7, 512) (None, 7, 7, 512)
conv5_block3_2_relu Activation input: output: (None, 7, 7, 512) (None, 7, 7, 512)
conv5_block3_3_conv Conv2D input: output: (None, 7, 7, 512) (None, 7, 7, 2048)
conv5_block3_out Add input: output: [(None, 7, 7, 2048), (None, 7, 7, 2048)] (None, 7, 7, 2048)
post_bn BatchNormalization input: output: (None, 7, 7, 2048) (None, 7, 7, 2048)
post_relu Activation input: output: (None, 7, 7, 2048) (None, 7, 7, 2048)
batch_normalization BatchNormalization input: output: (None, 7, 7, 2048) (None, 7, 7, 2048)
conv2d Conv2D input: output: (None, 7, 7, 2048) (None, 5, 5, 32)
max_pooling2d_3 MaxPooling2D input: output: (None, 5, 5, 32) (None, 2, 2, 32)
conv2d_1 Conv2D input: output: (None, 2, 2, 32) (None, 2, 2, 32)
max_pooling2d_4 MaxPooling2D input: output: (None, 2, 2, 32) (None, 1, 1, 32)
conv2d_2 Conv2D input: output: (None, 1, 1, 32) (None, 1, 1, 64)
max_pooling2d_5 MaxPooling2D input: output: (None, 1, 1, 64) (None, 1, 1, 64)
dropout Dropout input: output: (None, 1, 1, 64) (None, 1, 1, 64)
flatten Flatten input: output: (None, 1, 1, 64) (None, 64)
dense Dense input: output: (None, 64) (None, 1024)
dropout_1 Dropout input: output: (None, 1024) (None, 1024)
dense_1 Dense input: output: (None, 1024) (None, 4)

11 REFERENCES

1. python - Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX AVX2 - Stack Overflow
2. bugtijamal/flaskblog: this is a news blog (github.com)
3. flatplanet/flasker (github.com)
4. Flutter-Flask-Login/flask_app at master · mohammedhashim44/Flutter-Flask-Login (github.com)
5. bookmarker-api/bookmarks.py at main · CryceTruly/bookmarker-api (github.com)
6. Jinja — Jinja Documentation (3.1.x) (palletsprojects.com)
7. Flask-WTF — Flask-WTF Documentation (1.0.x)
8. WTForms — WTForms Documentation (3.0.x)
9. Flask-SQLAlchemy — Flask-SQLAlchemy Documentation (2.x) (palletsprojects.com)
10. marshmallow: simplified object serialization — marshmallow 3.17.0 documentation
11. Flask-JWT-Extended's Documentation — flask-jwt-extended 4.4.2 documentation
12. Flask-Share — Flask-Share 0.1.1 documentation
13. Flask-Login — Flask-Login 0.6.1 documentation
14. Welcome to Flask — Flask Documentation (2.1.x) (palletsprojects.com)
15. Werkzeug — Werkzeug Documentation (2.1.x) (palletsprojects.com)
16. python - How to handle file upload validations using Flask-Marshmallow? - Stack Overflow

17. _forms.html (github.com)
18. Deploy Machine Learning Models to Heroku with Flask - YouTube
19. Princekrampah/flaskTodolistAPITutorial (github.com)
20. MySQL :: MySQL Connector/Python Developer Guide
21. SQLAlchemy - The Database Toolkit for Python
22. Flask-Migrate · PyPI
23. vimalloc/flask-jwt-extended: An open source Flask extension that provides JWT support (with batteries included)! (github.com)
24. https://www.kaggle.com/datasets/mohamedhanyyy/chest-ctscan-images
25. https://github.com/fawziafaraag/lung-cancer-detectionس
26. https://github.com/fawziafaraag/graduation-_pro
27. https://www.kaggle.com/code/yasserhessein/chest-cancer-classification-with-vvg-16
28. https://www.kaggle.com/code/iqrar99/chest-ct-scan-resnet50-with-transfer-learning
29. https://www.kaggle.com/code/vincentsiow/chest-ct-scan-using-resnet101v2
30. https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/
31. https://www.investopedia.com/terms/n/neuralnetwork.asp
32. https://en.wikipedia.org/wiki/Logistic_regression/
33. https://www.geeksforgeeks.org/k-nearest-neighbor-algorithm-in-python/
34. https://www.geeksforgeeks.org/k-nearest-neighbor-algorithm-in-python/
35. https://scikit-learn.org/stable/modules/tree.html
36. https://www.researchgate.net/publication/348167140_Detection_of_Lung_Cancer_using_Machine_Learning_Techniques_Based_on_Routine_Blood_Indices
37. https://www.researchgate.net/publication/351889513_Lung_cancer_Prediction_and_Classification_based_on_Correlation_Selection_method_Using_Machine_Learning_Techniques?fbclid=IwAR2RxsF_H87GpuHe4BtibucA-7IDGFUTtfRfmPVv9xLdcO_7zAiWPspNO9Y

38. https://www.researchgate.net/publication/322350907_Lung_Cancer_Detection_using_CT_Scan_Images
39. https://drive.google.com/file/d/1lFV82J-clnzlucNRWo8jQ7JxJ1LvywxD/view?usp=sharing
40. https://pub.dev/
41. https://www.udemy.com/course/complete-flutter-arabic/
42. https://www.healthline.com/health/lung-cancer/effects-on-body#Respiratory-system