**EXTERNSHIP PROJECT**

# Detecting Building Defects Using VGG16 & IBM WATSON

**Guided by:**

**Mr. HARI PRABHU**

**Mrs. KAMYA PALIWAL**

**Mr. PRINCE**

**Submitted by**

**MAKAM DHEERAJ**

**KUNKU SUJAN PRAJAY KUMAR**

**M SHOAIB KHAN**

**KANKATALA S S ANIRUDH GUPTA**

## Contents

# 1. INTRODUCTION

## 1.1 Overview

Detecting building defects is the process of identifying and assessing any damage or deterioration that has occurred to a building. It can be used to know whether a building or a wall needs to be restored or constructed again. We will be classifying a building into two types. Which are cracked and not cracked. To implement this, we will be using VGG16 as transfer learning technique so that we can achieve accuracy, Interpretability, Scalability and Robustness which are the most important factors in this project. The CNN (convolutional neural network) model VGG16(Visual Geometry Group) is strong and adaptable, making it suitable for a range of image recognition tasks. It is an excellent option for applications where accuracy and interpretability are crucial. VGG16 is a 16-layer CNN model that consists of 13 convolutional layers and 3 fully connected layers. The convolutional layers use 3x3 filters with a stride of 1, and the fully connected layers use 4096 neurons each. VGG16 was trained on the ImageNet dataset, which is a dataset of over 14 million images belonging to 1000 classes.

## 1.2 Purpose

The purpose of detecting building defects because it plays an important role in our day-to-day life because as we can see that a lot of buildings and bridges are collapsing due to improper construction or negligence by the contractors and as well builders as a lot of innocent people are losing their lives. We believe that this project is just a start to classify whether a wall is cracked or not but in future we have a concept of making which is more evolved towards this like we can preventing future defects, improving safety, causes of the defects, making recommendations and also, we don't need any guidance of builders where we can just add how much will it cost a person to reconstruct it. So, we think that this project just the first step for a brighter future.

# 2. LITERATURE REVIEW

## 2.1 Existing Problem

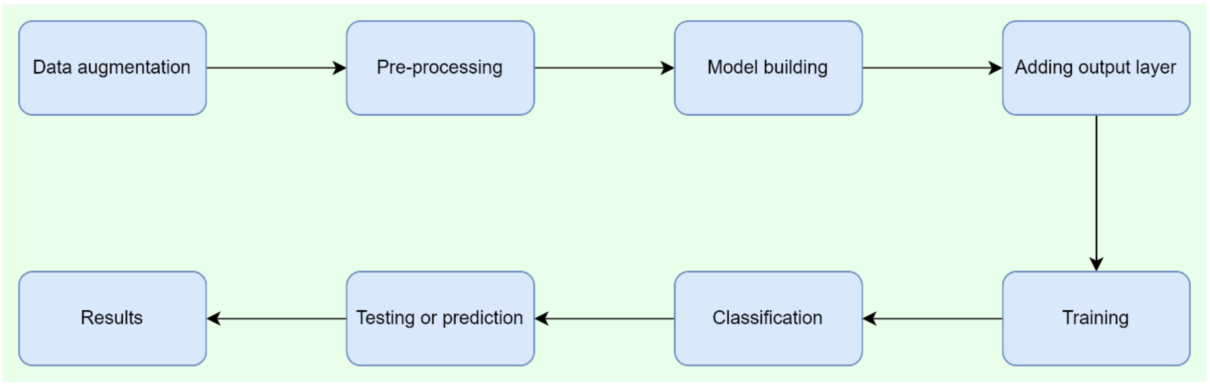
To present a comprehensive, organised, and analytical overview of the current, widely applied object detection models that can be used for defect detection, including Region based CNNs (Convolutional neural networks), YOLO (You only look once), SSD (single shot detectors), and cascaded architectures (stating both advantages and disadvantages). The methods of model acceleration and compression that made it possible for deep learning detection models to be portable are also briefly explained.

## 2.2 Proposed solution

The goal of this research is to create a model that categorises dampness-related faults as "normal" or "mould," "stain," or "deterioration," depending on whether they present in a given image. In this work, we also investigate how much of a role ConvNets plays in dealing with issues brought on by the nature of the flaws under inquiry and the surrounding environment. For instance, one study found that mould in homes can be any colour, including black, brown, green, olive-green, grey, blue, white, yellow, or even pink [66]. The location (walls, ceilings, corners, etc.), background (paint colour, wallpaper patterns, fabric, etc.), and intensity (intensity) of the surrounding environment all have a significant impact on the physical characteristics of stains and paint deterioration. Furthermore, stains and paint deterioration lack a defined shape, size, or colour by the brightness of the light at the time these faults' photographs were captured. When trying to gather a sufficient, sizable dataset to train a model to categorise all of these situations, the irregular character of the faults provides a significant barrier. The proposed solution for the project is we are going to use the transfer learning model which is VGg16. The reason behind using that is the VGG16 model is pretrained with a large data set called ImageNet. By using this pretrained model we can train our model easily and more accurately so that the defects can be traced or predicted correctly. Based on that prediction the further measures can be taken so the building defects do not prolong. The accuracy and prediction results will be discussed in later sections as we go on.

## 3. THEORITICAL ANALYSIS

### 3.1 Block diagram



### 3.2 Hardware / Software designing

The key requirements hardware and software resources are:

**Hardware Requirements:**

1. Computer**:** A computer system with sufficient processing power and memory is necessary for running the project. The specifications will depend on the complexity and size of the dataset, as well as the training and evaluation processes.
2. Storage**:** Sufficient storage capacity is required to store the dataset, pre-processed images, trained models, and other relevant files. The storage capacity should accommodate the size of the dataset and any augmented data generated during preprocessing.
3. GPU (Graphics Processing Unit): Training deep learning models like VGG16 can be computationally intensive. To expedite the training process, it is beneficial to have access to a GPU. GPUs can significantly accelerate the training of neural networks by parallelizing computations.

**Software Requirements:**

1. Python: The project will be implemented using Python programming language, which provides a rich ecosystem of libraries and frameworks for deep learning tasks. Ensure that Python is installed on the computer system.
2. Deep Learning Libraries: Install the necessary deep learning libraries, such as TensorFlow, Keras, or PyTorch, which offer high-level APIs for building and training deep neural networks.
3. Image Processing Libraries: Install libraries like PIL (Python Imaging Library) for image processing tasks such as resizing, normalization, and data augmentation.

4. Data Science Libraries: Utilize data science libraries like NumPy and pandas for handling and manipulating the dataset, as well as performing various data preprocessing tasks.
5. Integrated Development Environment (IDE): Visual Studio Code, to write and execute the Python code.
6. Version Control: It is recommended to use a version control system like Git to manage the project's source code and track changes throughout development.

## 4. EXPERIMENTAL INVESTIGATIONS

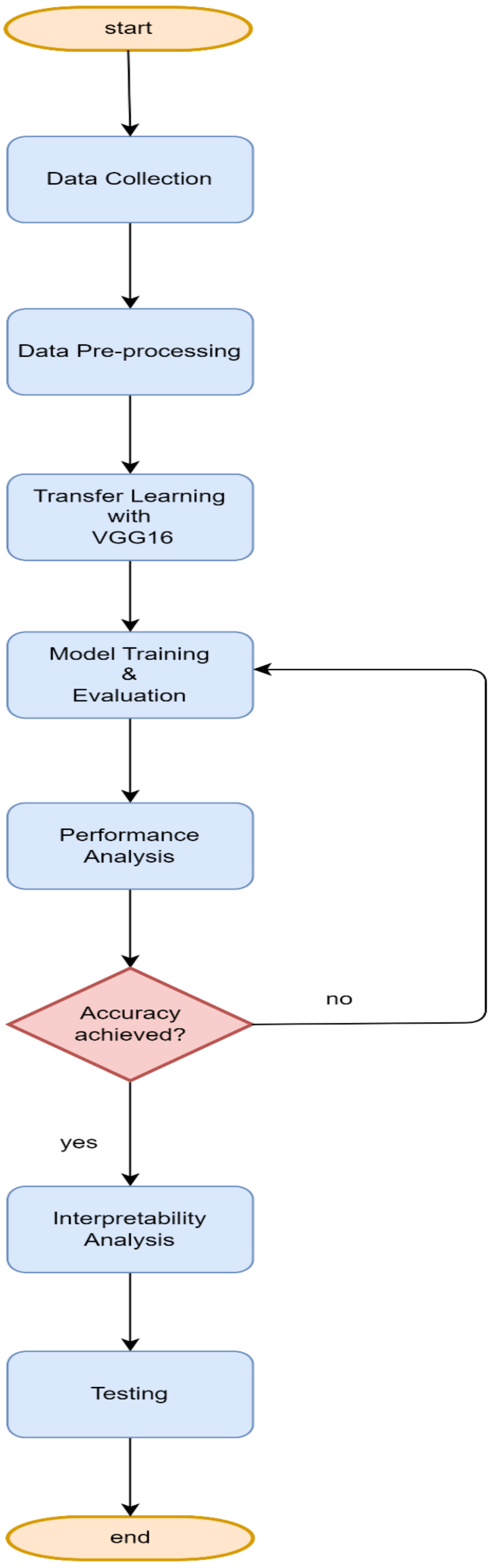
During the course of the project, several experimental investigations were conducted to develop and optimize the building defect detection system. Here is an overview of the key investigations:

1. Dataset Analysis: A thorough analysis of the collected dataset was performed. This involved assessing the size of the dataset, the distribution of cracked and non-cracked samples, and potential class imbalances. Exploratory data analysis techniques were employed to gain insights into the dataset's characteristics and identify any challenges or biases.
2. Preprocessing Techniques: Various preprocessing techniques were experimented with to prepare the dataset for training. These techniques included resizing the images to a consistent resolution, normalizing pixel values, and applying data augmentation methods such as rotation, scaling, and flipping. The impact of these preprocessing techniques on the model's performance and generalization ability was evaluated.
3. Model Architecture Exploration: In addition to using the VGG16 model as a transfer learning technique, alternative architectures were explored. This involved experimenting with different CNN architectures, varying the number and size of layers, and adjusting hyperparameters such as learning rates and dropout rates. The goal was to identify the most suitable architecture for accurately classifying building defects.
4. Training and Validation Strategies: Different training and validation strategies were tested to optimize model performance. This included investigating the effect of batch sizes, learning rate schedules, and early stopping techniques. The aim was to find the optimal training approach that resulted in convergence and prevented overfitting.
5. Hyperparameter Tuning: Systematic hyperparameter tuning was performed to fine-tune the model's performance. Techniques such as grid search or random search were employed to explore different combinations of hyperparameters and identify the configuration that yielded the best results in terms of accuracy, precision, recall, and F1 score.
6. Cross-Validation: Cross-validation techniques, such as k-fold cross-validation, were employed to assess the model's performance across multiple subsets of the dataset. This helped estimate the model's generalization ability and mitigate the impact of any specific biases present in the dataset.
7. Performance Evaluation: The trained model was evaluated using the testing set to measure its performance metrics, including accuracy, precision, recall, and F1 score. The model's ability to correctly classify cracked and non-cracked buildings was assessed, and any limitations or challenges encountered during evaluation were documented.

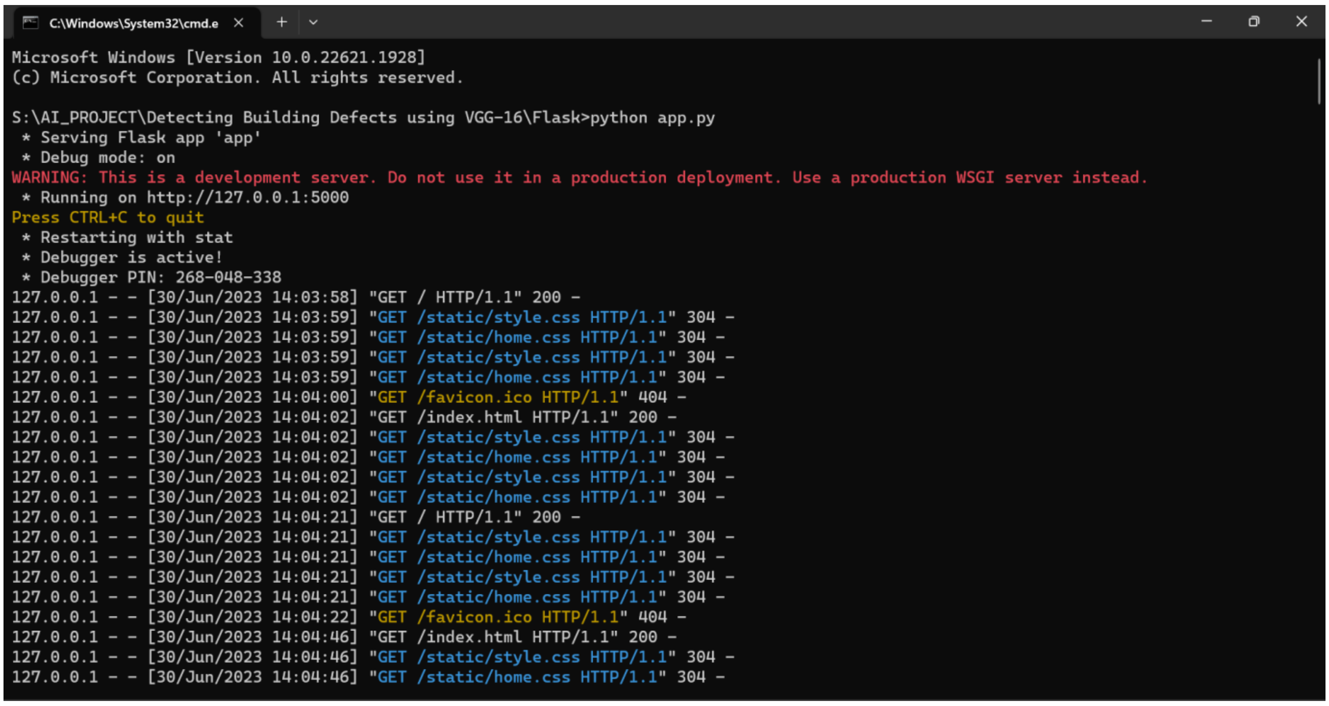
Throughout these experimental investigations, iterative refinement and analysis were performed to enhance the system's performance and address any limitations or issues. The results of these investigations formed the basis for decision-making and guided the development of an optimized building defect detection system.

By conducting these experimental investigations, the project aimed to fine-tune the model, optimize its performance, and ensure its effectiveness in accurately classifying building defects, thereby contributing to the overall success of the project.

## 5. FLOWCHART


start
Data Collection
Data Pre-processing
Transfer Learning with VGG16
Model Training & Evaluation
Performance Analysis
Accuracy achieved?
no
yes
Interpretability Analysis
Testing
end

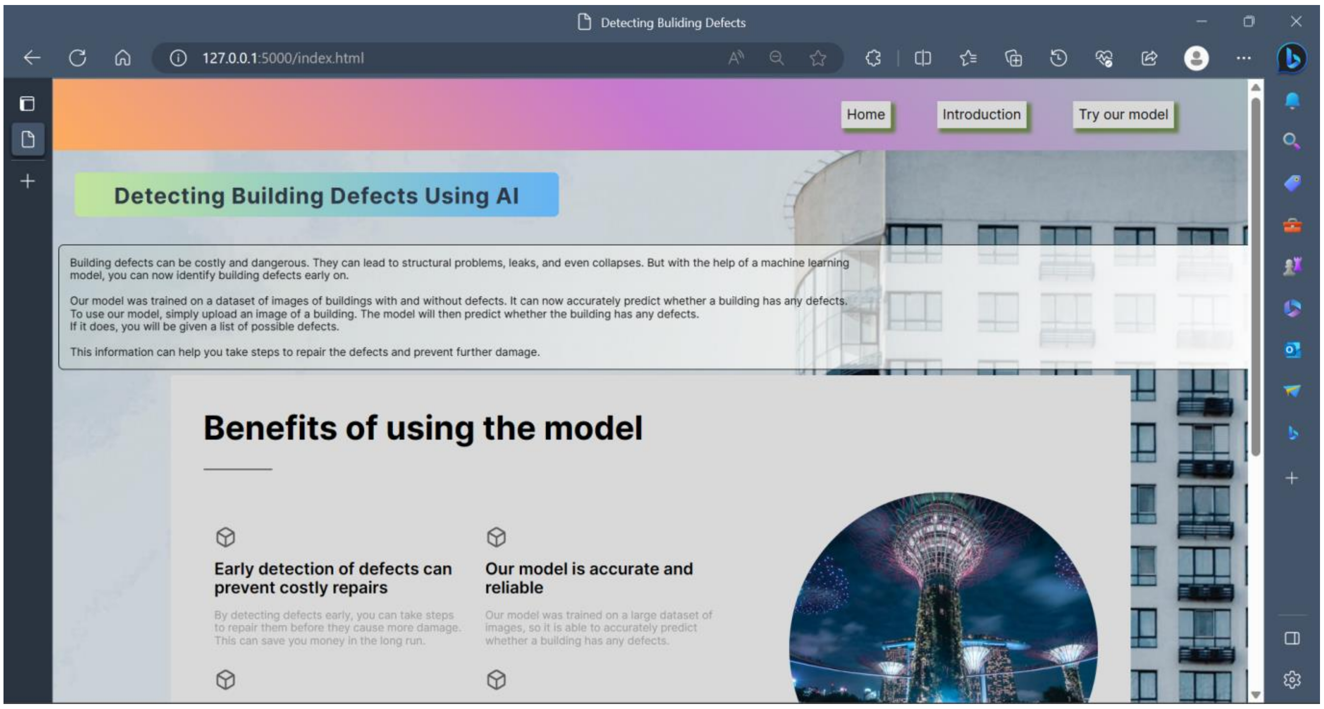
## 6. RESULT



```
Microsoft Windows [Version 10.0.22621.1928]
(c) Microsoft Corporation. All rights reserved.

S:\AI_PROJECT\Detecting Building Defects using VGG-16\Flask>python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 268-048-338
127.0.0.1 - - [30/Jun/2023 14:03:58] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2023 14:03:59] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:03:59] "GET /static/home.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:03:59] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:03:59] "GET /static/home.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:00] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [30/Jun/2023 14:04:02] "GET /index.html HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2023 14:04:02] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:02] "GET /static/home.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:02] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:02] "GET /static/home.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:21] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2023 14:04:21] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:21] "GET /static/home.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:21] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:21] "GET /static/home.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:22] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [30/Jun/2023 14:04:46] "GET /index.html HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2023 14:04:46] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [30/Jun/2023 14:04:46] "GET /static/home.css HTTP/1.1" 304 -
```

### Home page:



Detecting Building Defects Using AI

Building defects can be costly and dangerous. They can lead to structural problems, leaks, and even collapses. But with the help of a machine learning model, you can now identify building defects early on.

Our model was trained on a dataset of images of buildings with and without defects. It can now accurately predict whether a building has any defects.
To use our model, simply upload an image of a building. The model will then predict whether the building has any defects.
If it does, you will be given a list of possible defects.

This information can help you take steps to repair the defects and prevent further damage.

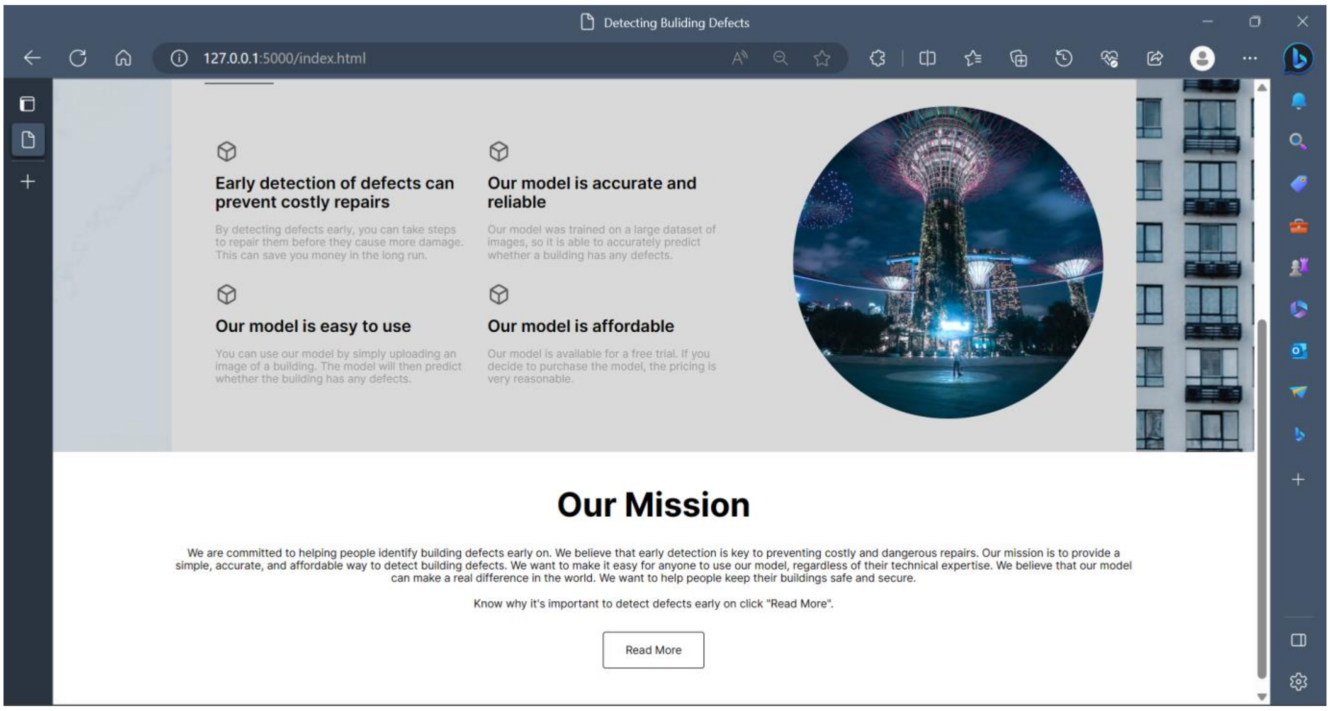
Benefits of using the model

Early detection of defects can prevent costly repairs

By detecting defects early, you can take steps to repair them before they cause more damage. This can save you money in the long run.

Our model is accurate and reliable

Our model was trained on a large dataset of images, so it is able to accurately predict whether a building has any defects.



Early detection of defects can prevent costly repairs

By detecting defects early, you can take steps to repair them before they cause more damage. This can save you money in the long run.

Our model is accurate and reliable

Our model was trained on a large dataset of images, so it is able to accurately predict whether a building has any defects.

Our model is easy to use

You can use our model by simply uploading an image of a building. The model will then predict whether the building has any defects.

Our model is affordable

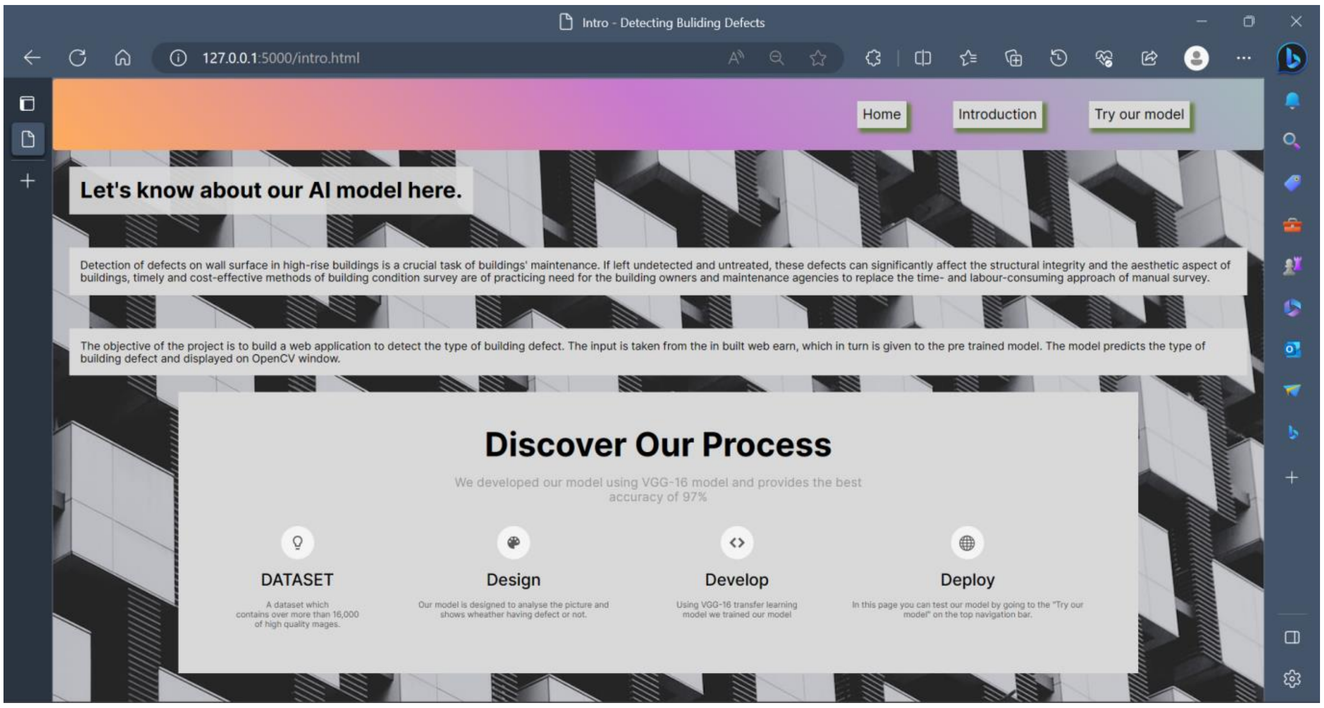
Our model is available for a free trial. If you decide to purchase the model, the pricing is very reasonable.

Our Mission

We are committed to helping people identify building defects early on. We believe that early detection is key to preventing costly and dangerous repairs. Our mission is to provide a simple, accurate, and affordable way to detect building defects. We want to make it easy for anyone to use our model, regardless of their technical expertise. We believe that our model can make a real difference in the world. We want to help people keep their buildings safe and secure.

Know why it's important to detect defects early on click "Read More".

## Intro page:



Let's know about our AI model here.

Detection of defects on wall surface in high-rise buildings is a crucial task of buildings' maintenance. If left undetected and untreated, these defects can significantly affect the structural integrity and the aesthetic aspect of buildings, timely and cost-effective methods of building condition survey are of practicing need for the building owners and maintenance agencies to replace the time- and labour-consuming approach of manual survey.

The objective of the project is to build a web application to detect the type of building defect. The input is taken from the in built web earn, which in turn is given to the pre trained model. The model predicts the type of building defect and displayed on OpenCV window.

Discover Our Process

We developed our model using VGG-16 model and provides the best accuracy of 97%

DATASET

A dataset which contains over more than 16,000 of high quality mages.

Design

Our model is designed to analyse the picture and shows wheather having defect or not.

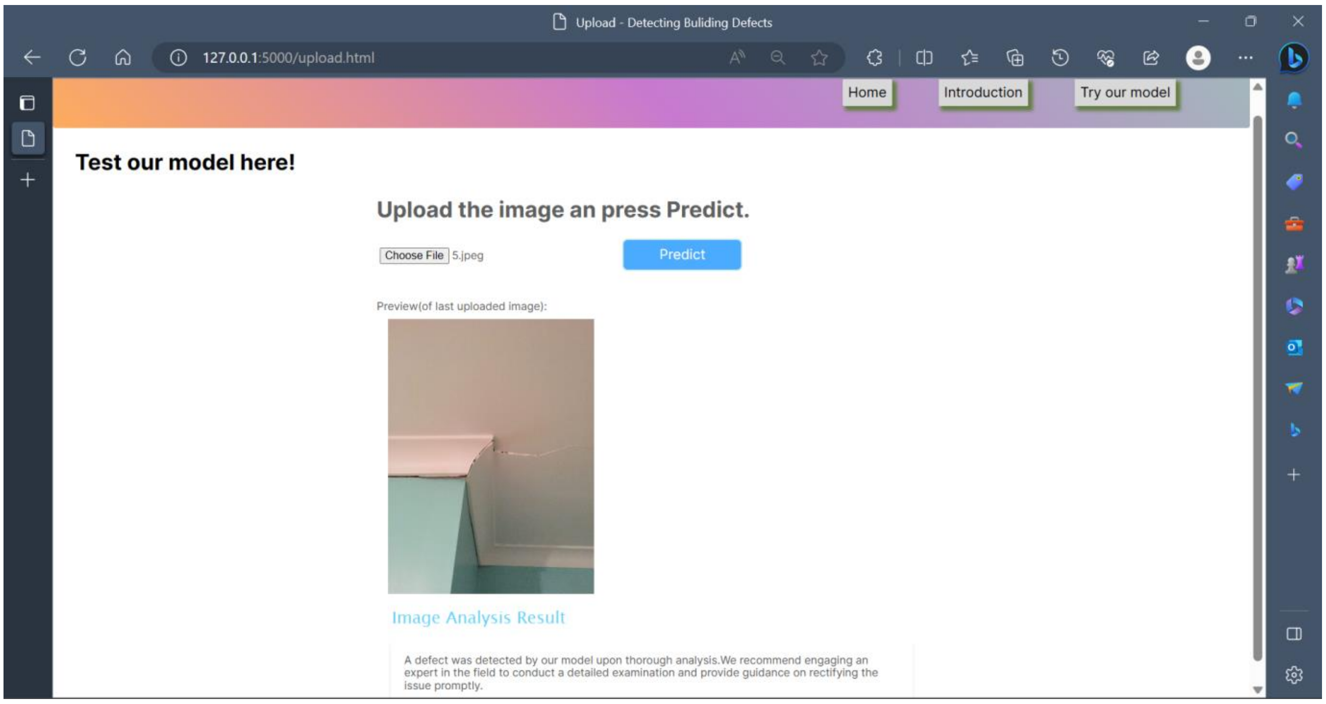
Develop

Using VGG-16 transfer learning model we trained our model

Deploy

In this page you can test our model by going to the "Try our model" on the top navigation bar.

## Upload page:



Upload - Detecting Buliding Defects

127.0.0.1:5000/upload.html

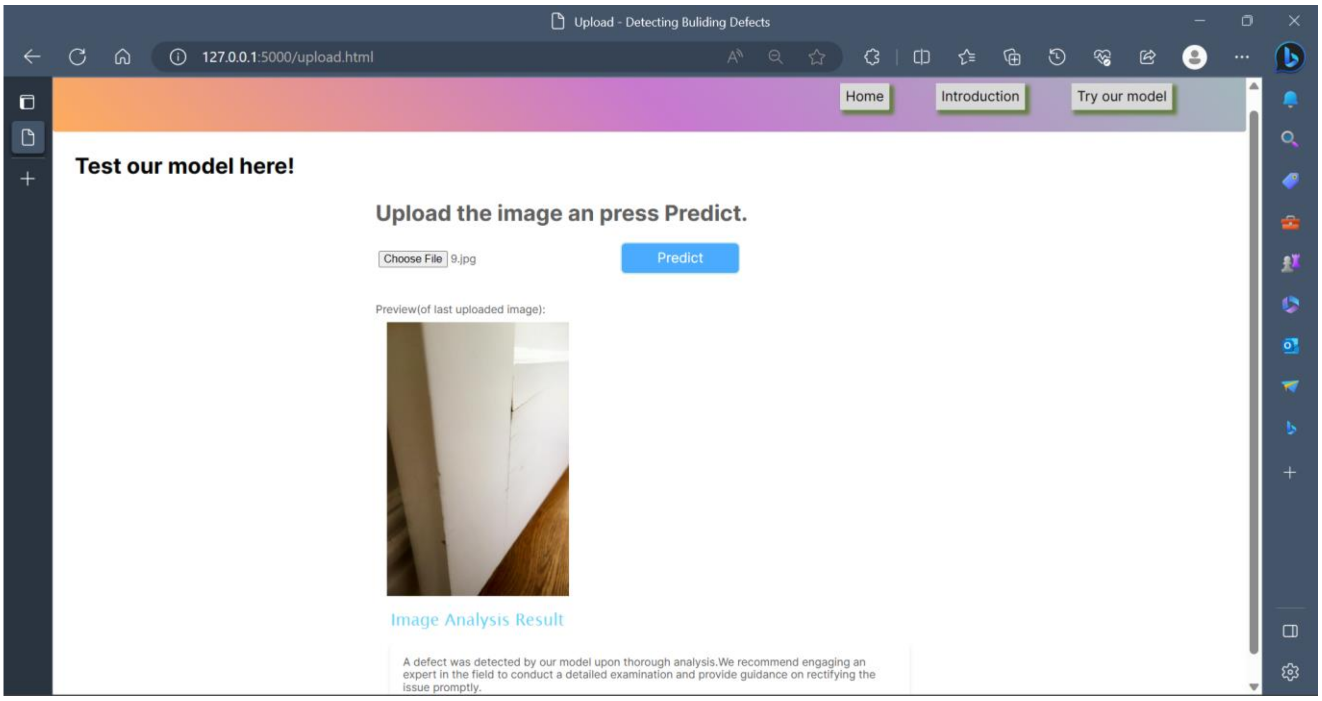
**Test our model here!**

**Upload the image an press Predict.**

Preview(of last uploaded image):

Image Analysis Result

A defect was detected by our model upon thorough analysis.We recommend engaging an expert in the field to conduct a detailed examination and provide guidance on rectifying the issue promptly.



Upload - Detecting Buliding Defects

127.0.0.1:5000/upload.html

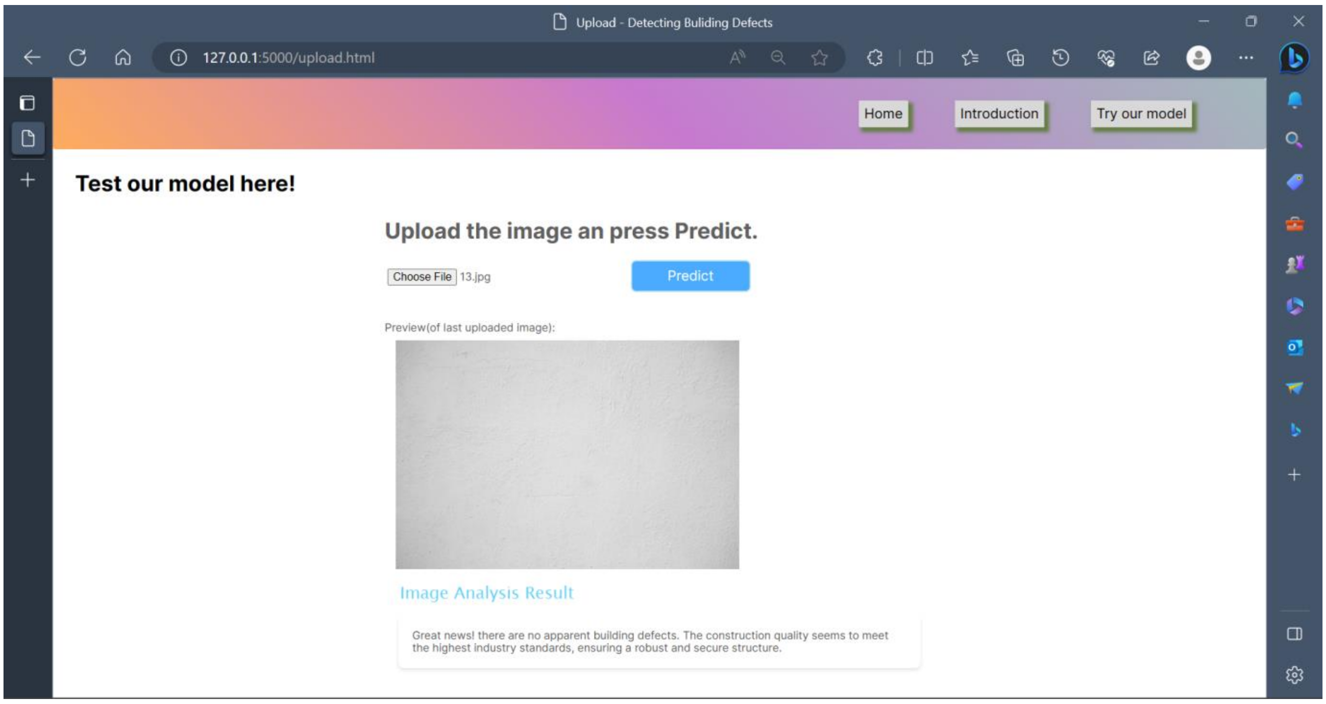
**Test our model here!**

**Upload the image an press Predict.**

Preview(of last uploaded image):

Image Analysis Result

A defect was detected by our model upon thorough analysis.We recommend engaging an expert in the field to conduct a detailed examination and provide guidance on rectifying the issue promptly.


Upload - Detecting Buliding Defects
127.0.0.1:5000/upload.html
Home
Introduction
Try our model
Choose File
13.jpg
Predict

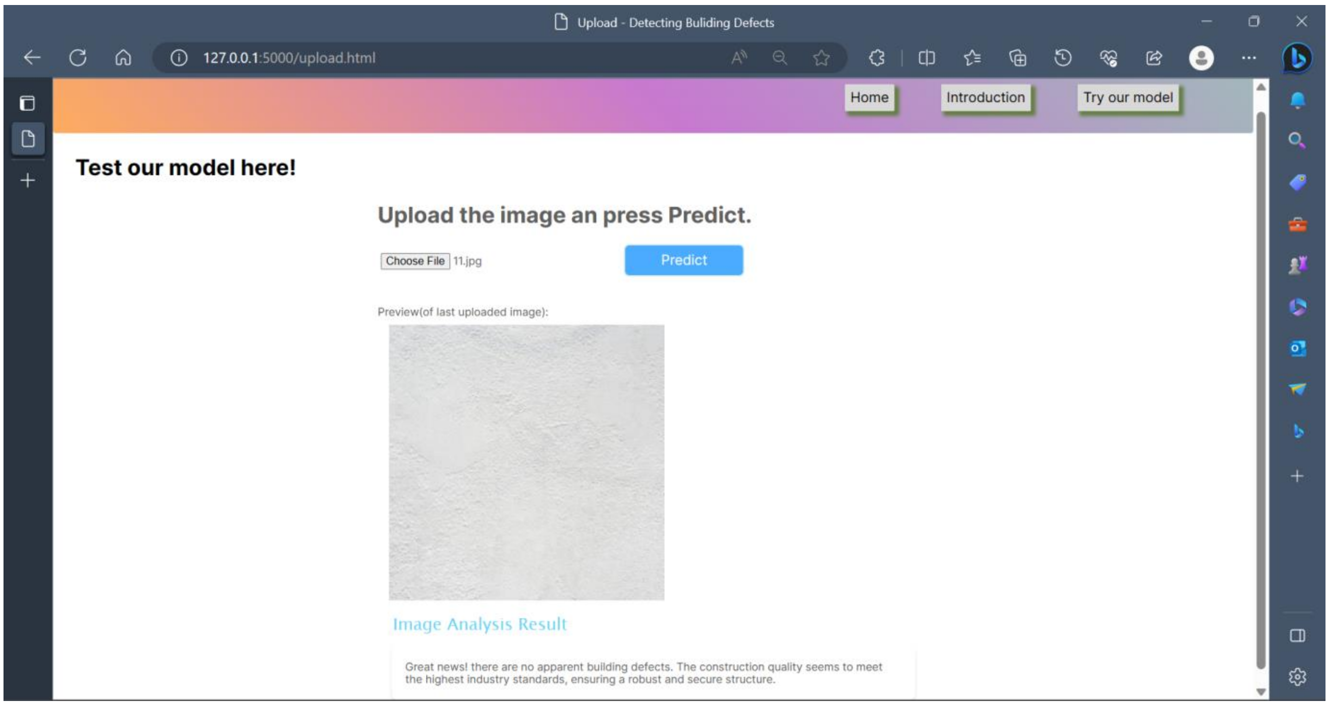
Test our model here!

Upload the image an press Predict.

Preview(of last uploaded image):

Image Analysis Result

Great news! there are no apparent building defects. The construction quality seems to meet the highest industry standards, ensuring a robust and secure structure.


Upload - Detecting Buliding Defects
127.0.0.1:5000/upload.html
Home
Introduction
Try our model
Choose File
11.jpg
Predict

Test our model here!

Upload the image an press Predict.

Preview(of last uploaded image):

Image Analysis Result

Great news! there are no apparent building defects. The construction quality seems to meet the highest industry standards, ensuring a robust and secure structure.

## 7. ADVANTAGES & DISADVANTAGES

**Advantages**:

- Prevention: Early detection of defects can prevent further damage to the building. This can save money in the long run, as it will reduce the cost of repairs.
- Safety: Defects can pose a safety hazard to the occupants of the building. Early detection of defects can help to prevent accidents and injuries.

- Value: A building with no defects is more valuable than a building with defects. Early detection of defects can help to increase the value of the property.
- Risk: Defects can increase the risk of a building failing. Early detection of defects can help to manage this risk and reduce the likelihood of a failure.
- Compliance: Defects can lead to compliance issues. Early detection of defects can help to ensure that the building complies with all applicable regulations.

**Disadvantages**:

- Cost: Detecting building defects can be expensive. The cost of the inspection will vary depending on the size and complexity of the building.
- Time: Detecting building defects can take time. The inspection may need to be carried out over several days or weeks, depending on the size of the building and the severity of the defects.
- Inconvenience: Detecting building defects can be inconvenient for the occupants of the building. The inspection may require access to areas of the building that are normally off-limits, and it may also cause some disruption to normal activities.
- Uncertainty: The results of the inspection may not be definitive. There is always a degree of uncertainty involved in detecting building defects, and it is possible that some defects may not be detected.

Overall, the advantages of detecting building defects outweigh the disadvantages. By detecting defects early, it is possible to prevent further damage, improve safety, increase the value of the property, manage risk, and enhance compliance. However, it is important to be aware of the costs and inconvenience involved in detecting building defects.

## 8. APPLICATIONS

There are many applications of detecting building defects. Here are some of the most common:

- Preventing further damage: Early detection of defects can prevent further damage to the building. This can save money in the long run, as it will reduce the cost of repairs.
- Improving safety: Defects can pose a safety hazard to the occupants of the building. Early detection of defects can help to prevent accidents and injuries.
- Increasing the value of the property: A building with no defects is more valuable than a building with defects. Early detection of defects can help to increase the value of the property.
- Managing risk: Defects can increase the risk of a building failing. Early detection of defects can help to manage this risk and reduce the likelihood of a failure.
- Enhancing compliance: Defects can lead to compliance issues. Early detection of defects can help to ensure that the building complies with all applicable regulations.

In addition to these general applications, there are also some specific applications of detecting building defects in particular industries. For example, in the construction industry, detecting defects can help to ensure that the building is built to code and that it meets the expectations of the owner.

In the insurance industry, detecting defects can help to assess the risk of a building and to determine the cost of repairs.

## 9. CONCLUSION

The project aims to develop a system for detecting building defects using VGG16 as a transfer learning technique. By classifying buildings into cracked and non-cracked categories, it can help identify areas that require restoration or reconstruction, ultimately improving safety and preventing potential accidents.

The use of VGG16 as the underlying model provides accuracy, interpretability, scalability, and robustness, making it a suitable choice for this project. The trained model can be further enhanced and expanded in the future to include additional features such as cost estimation for reconstruction and recommendations for defect prevention.

Overall, this project serves as a crucial step towards a brighter future in building defect detection and aims to contribute to the improvement of construction practices, safety standards, and the preservation of human lives.

## 10. FUTURE SCOPE

The building defect detection project lays the foundation for further enhancements and advancements. Here are some potential areas for future scope and improvements:

1. Multi-class Classification: Currently, the project focuses on binary classification, distinguishing between cracked and non-cracked buildings. In the future, the system can be expanded to support multi-class classification, identifying various types of building defects such as structural damage, leakage, corrosion, etc. This would provide a more comprehensive and detailed assessment of building conditions.
2. Real-time Defect Detection: The current implementation operates on static images of buildings. Enhancements can be made to enable real-time defect detection using video streams or live camera feeds. This would allow for continuous monitoring and prompt identification of defects as they occur, facilitating timely remedial actions.
3. Deployment on Edge Devices: To extend the applicability of the system, optimizing the model for deployment on edge devices such as smartphones, IoT devices, or drones can be explored. This would enable on-site defect detection and empower users to assess building conditions conveniently and in remote locations.
4. Integration with Building Information Modeling (BIM): Building Information Modeling (BIM) systems are extensively used in the construction industry for design, planning, and management. Integrating the defect detection system with BIM software would enable automatic defect identification, generating informative reports, and facilitating seamless collaboration between stakeholders for efficient defect resolution.
5. Defect Severity Assessment: Rather than solely classifying defects, future enhancements could include a severity assessment component. The system could assign a severity level to identified defects, providing insights into the urgency and criticality of remedial actions required.

6. Data Expansion and Generalization: Expanding the dataset to include a more diverse range of building images, encompassing different architectural styles, materials, and environmental conditions, would enhance the model's generalization capabilities. This would improve the system's performance and reliability across a wider variety of real-world scenarios.
7. Active Learning and Feedback Mechanisms: Implementing active learning techniques and incorporating user feedback mechanisms would allow the system to continuously improve and adapt. Users could provide feedback on detected defects, helping refine the model's predictions and addressing false positives or false negatives
8. Localization of Defects: Going beyond the binary classification, enhancing the system to detect and localize the specific regions of defects within building images would be valuable. This would enable targeted repair or reconstruction efforts, resulting in more efficient and cost-effective maintenance processes.
9. Integration with Cost Estimation: Incorporating cost estimation algorithms and construction material pricing data would enable the system to estimate the financial implications of repairing or reconstructing buildings with detected defects. This would assist stakeholders in making informed decisions regarding remedial actions and budget planning.
10. Collaboration with Building Professionals: Collaborating with architects, engineers, and building professionals would provide valuable domain expertise. Their insights could help refine the system's algorithms, improve defect detection accuracy, and develop comprehensive recommendations for defect resolution.

By pursuing these future enhancements, the building defect detection project can evolve into a more advanced, versatile, and intelligent system that contributes to proactive defect prevention, improved safety standards, and cost-effective building maintenance and management.

## 11 BIBILOGRAPHY

Deep Learning for Detecting Building Defects Using Convolutional Neural Networks

https://www.mdpi.com/1424-8220/19/16/3556

https://www.researchgate.net/publication/335175972_Deep_Learning_for_Detecting_Building_Defects_Using_Convolutional_Neural_Networks

## 12 APPENDIX

```
from tensorflow.keras.layers import Dense,Flatten,Input
from tensorflow.keras.models import Model
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img
import numpy as np
train_path = "/content/DATASET/train"
test_path = "/content/DATASET/test"
train_gen = ImageDataGenerator(rescale=1./255,
                               shear_range=0.2,
                               zoom_range=0.2,
                               horizontal_flip=True)
```

```
test_gen = ImageDataGenerator(rescale=1./255)
train = train_gen.flow_from_directory(train_path,
                                      target_size=(224,224),
                                      batch_size=22,
                                      class_mode='categorical')

test = test_gen.flow_from_directory(test_path,
                                      target_size=(224,224),
                                      batch_size=22,
                                      class_mode='categorical')
from tensorflow.keras.applications.vgg16 import VGG16, preprocess_input
vgg = VGG16(include_top=False,weights='imagenet',input_shape=(224,224,3))
for layer in vgg.layers:
  print(layer)
# Train model with existing weights

for layer in vgg.layers:
  layer.trainable=False
x = Flatten()(vgg.output)
# output layer

prediction = Dense(2,activation='softmax')(x)
# Create Vgg16 model

model_vgg16 = Model(inputs=vgg.input,outputs=prediction)
model_vgg16.compile(loss='categorical_crossentropy',optimizer='adam',metrics=[
'accuracy'])
model_vgg16.fit_generator(train,validation_data=test,epochs=10,steps_per_epoch
=len(train),validation_steps=len(test))
model_vgg16.save('BuildingWeights_vgg16.h5')
# Testing 1
img1 =
image.load_img('/content/DATASET/test/crack/1486.jpg',target_size=(224,224))
img1 = image.img_to_array(img1)
img1 = np.expand_dims(img1,axis=0)
pred = np.argmax(model_vgg16.predict(img1))
print(pred)
output = ['Cracked','Non-cracked']
print(output[pred])

if output[pred]=='Cracked':
  print("Building Has a Defect!!! Better repair it!")
else:
  print("Building has no Defect!!!")
```