A PROJECT REPORT

ON

FACE MASK DETECTION USING

MACHINE LEARNING

Submitted in partial fulfillment of the requirements for the award of the degree in

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

BY

**M.JAHNAVI (20NN1A1286)**

**N.GNANA POOJITHA (20NN1A1280)**

**B.HEMALATHA (20NN1A1272)**

**G.REVATHI (20NN1A1282)**

## Under the esteemed guidance of

## Lakshmi Durga

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**VIGNAN’S NIRULA INSTITUTE OF TECHNOLOGY AND SCIENCE FOR WOMEN**

#### (Approved by AICTE, NEW DELHI and Affiliated to JNTUK) PEDAPALAKALURU, GUNTUR-522005 (2020-2024)

**VIGNAN’S NIRULA INSTITUTE OF TECHNOLOGY AND SCIENCE FOR WOMEN**

#### (Approved by AICTE, NEW DELHI and Affiliated to JNTUK)

**PEDAPALAKALURU, GUNTUR-522005**

**(2020-2024)**

## DEPARTMENT OF INFORMATION TECHNOLOGY



**CERTIFICATE**

This is to certify that the project work entitled “FACE MASK DETECTION USING MACHINE LEARNING“is a bonafide work submitted by M.Jahnavi(20NN1A1286), N.Gnana Poojitha(20NN1A1280), B.Hemalatha(20NN1A1272), G.Revathi(20NN1A1282) from the department of Information Technology in the partial fulfillment of the requirements forward of degree of Bachelor of Technology in Information Technology from Vignan’s Nirula Institute of Technology & Science for Women, Guntur.

**Internal Guide Head of the Department**

**Laksmi Durga Dr. Ramakrishna**

**External Examiner**

**DECLARATION**

We hereby declare that the work described in this project work, entitled **“**FACE MASK DETECTION USING MACHINE LEARNING**”** which is submitted by us in partial fulfillment for the award of Bachelor of Technology in the department of Information Technology to the Vignan’s Nirula Institute of Technology & Science for women, affiliated to Jawaharlal Nehru Technological University Kakinada, Andhra Pradesh, is the result of work done by us under the esteemed guidance of Lakshmi Durga, Assistant Professor.

The work is original and has not been submitted for any Degree/Diploma of this or any other university.

**M.Jahnavi (20NN1A1286)**

**N.Gnana Poojitha (20NN1A1280)**

**B.Hemalatha (20NN1A1272)**

**G.Revathi (20NN1A1282)**

**ACKNOWLEDGEMENT**

We profoundly grateful to express our deep sense of gratitude and respect towards our honorable chairman**, LAVU RATHAIAH** sir ,Chairman of Vignan group for his precious support in the college.

We are much thankful to **DR.P.RADHIKA**, Principal VNITSW, Guntur, for her support during and till the completion of the project.

We would like to thank **DR.RAMA KRISHNA**, Associate Professor and Head of the Department of Information Technology, for his extended and continuous support, valuable guidance and timely advices in the completion of this project thesis.

We wish to express our profound sense of sincere gratitude to our Project Guide, **LAKSHMI DURGA**, Assistant Professor of Information Technology, without whose help, guidance and motivation this project thesis could not have been completed the project successfully.

We also thank all the faculty of the Department of Information Technology for their help and guidance of numerous occasions, which has given us the cogency to build-up adamant aspiration over the completion of our project thesis and finally, we thank one and all who directly or indirectly helped us to complete our project thesis successfully.

**M.JAHNAVI (20NN1A1286)**

**Y.v.k.Chaitanya 17NN1A1251 N.Vindhya 17NN1A1235 N.Yamini poojitha 17NN1A1236**

**R.Krishnaveni 17NN1A1240**

**N.GNANA POOJITHA (20NN1A1280)**

**B.HEMALATHA (20NN1A1272)**

**G.REVATHI (20NN1A1282)**

ABSTRACT:

As per the guidance from the World Health Organization (WHO), the global COVID-19 pandemic is causing a widespread health crisis. The most effective preventive measure recommended is the wearing of facial coverings in public spaces. The COVID-19 outbreak has necessitated governments around the world to enforce lockdown measures in order to curb the spread of the virus. Wearing a facial mask while at work, as per various reports, significantly reduces the risk of viral transmission. The employment of artificial intelligence (AI) to establish a secure environment in a manufacturing setting is a practical and economical approach. The concept of face mask detection will be illustrated using a deep learning model. We will employ OpenCV to achieve real-time facial recognition from a live webcam feed, using a dataset that contains images of individuals both with and without masks. With the assistance of Python, OpenCV, and Keras, we will create a COVID-19 facial mask identifier through deep learning. By leveraging deep learning techniques, our aim is to determine whether the person in the image or video stream is adhering to mask-wearing guidelines.

**TABLE OF CONTENTS**

**CONTENTS PAGE NO**

1. INTRODUCTION 1-2
2. SYSTEM ANALYSIS AND DESCRIPTION 3-15
   1. Existing System 4-5
   2. Proposed System 5-11
   3. Software Development Life Cycle 11-13
   4. Study of the System 13-14
   5. System Analysis 14-15
3. REQUIREMENT SPECIFICATION 16-20
   1. Functional Requirements 17-20
4. LANGUAGES OF IMPLEMENTATION 21-33
   1. Introduction Script 22-23
   2. Python 23-33
5. SYSTEM DESIGN 34-60
   1. Introduction 35-36
   2. Data flow Diagrams 36-53
   3. UML diagram overview 54-59
6. IMPLEMENTATION 60-63
   1. Screenshots 61-63
7. SYSTEM TESTING 64-69
   1. White box testing 65-67
   2. Black box testing 68-69
8. CONCLUSION 70-71
9. FUTURE ENHANCEMENT 72-73
10. BIBILIOGRAPHY 74-76

**LIST OF FIGURES**

**FIGURE NAME PAGE NO**

1. Random Forest Tree 4
2. Complexity of SVM 6
3. Multilayer Perception 7
4. Hyperplane Classification 7
5. Software Development Life Cycle 11
6. Study of the System 14
7. UML Diagrams Overview 54
8. Use Case Diagram 55
9. Sequence Diagram 56
10. Activity Diagram 57
11. System Architecture 58
12. Code Screenshots 61-62
13. Output Screenshots 62-63

**CHAPTER-1:**

# 

# INTRODUCTION

**CHAPTER-1**

**INTRODUCTION**

* 1. **INTRODUCTION**

According to the World Health Organization (WHO)’s official Situation Report – 205, corona virus disease 2019 (COVID-19) has globally infected over 20 million people causing over 0.7million deaths [1]. Individuals with COVID 19 have had a wide scope of symptoms reported – going from mellow manifestations to serious illness. Respiratory problems like shortness of breath or difficulty in breathing is one of them. Elder people having lung disease can possess serious complications from COVID-19 illness as they appear to be at higher risk [2]. Some common human corona viruses that infect public around the world are 229E, HKU1, OC43,and NL63. Before debilitating individuals, viruses like 2019-nCoV, SARS- CoV, and MERS- CoV infect animals and evolve to human corona viruses [3]. Persons having respiratory problems can expose anyone (who is in close contact with them) to infective beads. Surroundings of a tainted individual can cause contact transmission as droplets carrying virus may withal arrive on his adjacent surfaces [4]. To curb certain respiratory viral ailments, including COVID-19, wearing a clinical mask is very necessary. The public should be aware of whether to put on the mask for source control or aversion of COVID-19. Potential points of interest of the utilization of masks lie in reducing vulnerability of risk from a noxious individual during the “pre-symptomatic” period and stigmatization of discrete persons putting on masks to restraint the spread of virus. WHO stresses on prioritizing medical masks and respirators for health care assistants[4]. Therefore, face mask detection has become a crucial task in present global society. Face mask detection involves in detecting the location of the face and then determining whether it has a mask on it or not. The issue is proximately cognate to general object detection to detect the classes of objects. Face identification categorically deals with distinguishing a specific group of entities i.e. Face. It has numerous applications, such as autonomous driving, education, surveillance, and so on [5]. This paper presents a simplified approach to serve the above purpose using the basic Machine Learning (ML) packages such as TensorFlow, Keras, OpenCV and Scikit-Learn. The rest of the paper is organized as follows: Section II explores related work associated with face mask detection. Section III discusses the nature of the used dataset. Section IV presents the details of the packages incorporated to build the proposed model. Section V gives an overview of our method. Experimental results and analysis are reported in section VI. Section VII concludes and draws the line towards future works.

# CHAPTER – 2

**SYSTEM ANALYSIS**

**AND DESCRIPTION**

## CHAPTER-2

**SYSTEM ANALYSIS AND DESCRIPTION**

### EXISTING SYSTEM

Convolutional networks are also undoubtedly used for undertakings of the classification task. Standard architectures like AlexNet [19] and VGGNet [20] are packed or contain a load of the convolutional layer. AlexNet the winner of the ImageNet LSVRC-2012 competition comprises 5 convolution layers plus 3 fully connected layers while VGGNet is an improvement of AlexNet, progressively shares large kernels with several 3x3 kernels. New architectures like ResNet [21] implement an accelerated link on training accuracy which allows for much deeper networks to avoid overload processing. These architectures are often applied in image recognition frameworks for original feature-extraction. In the existing system people are using the mask and not mask only but they are not detecting the improper mask which is very important too in contacting with the covid.

Although numerous researchers have committed efforts in designing efficient algorithms for face detection and recognition but there exists an essential difference between ‘detection of the face under mask’ and ‘detection of mask over face’. As per available literature, very little body of research is attempted to detect mask over face. Thus, our work aims to a develop technique that can accurately detect mask over the face in public areas (such as airports. railway stations, crowded markets, bus stops, etc. to curtail the spread of Coronavirus and thereby contributing to public healthcare. Further, it is not easy to detect faces with/without a mask in public as the dataset available for detecting masks on human faces is relatively small leading to the hard training of the model. So, the concept of transfer learning is used here to transfer the learned kernels from networks trained for a similar face detection task on an extensive dataset. The dataset covers various face images including faces with masks, faces without masks, faces with and without masks in one image and confusing images without masks. With an extensive dataset containing 45,000 images, our technique achieves outstanding accuracy of 98.2%. Face detection is a key area in the field of Computer Vision and Pattern Recognition. A significant body of research has contributed sophisticated to algorithms for face detection in past. The primary research on face detection was done in 2001 using the design of handcraft feature and application of traditional machine learning algorithms to train effective classifiers for detection and recognition. The problems encountered with this approach include high complexity in feature design.

#### DISADVANTAGES OF EXISTING SYSTEM:

* Takes more time to classify the mask images because of its long complex structure.
* It is not deployed completely

**2.2 PROPOSED MODEL**

Our proposed study uses the architectural features of VGG-16 as the foundation network for face recognition and the Fully-Convolutional segmentation network. The VGG-16 network is fairly robust in extracting features and less costly in computational terms. Nevertheless, most segmentation architectures consecutively rely on input image downsampling and upsampling. In proposed system we are detecting the people with the mask and no mask.During this global pandemic, Face mask detection has seen an outstanding development in image processing and Computer vision. Face masks can be detected by observing patterns in an image using various machine learning algorithms. Here three machine learning algorithms were considered, namely, CNN(Convolutional Neural Network), SVM(Support Vector Machine), and PCA (Principal Component Analysis). Dataset used consists of images of people with masks and without masks extracted from Kaggle.

In this literature, the efficiency of the detection of face masks after performing K-cross validation has been calculated. With most algorithms that handle image processing, the filters are typically created by an engineer based on heuristics. CNNs can learn what characteristics in the filters are the most important. That saves a lot of time and trial and error work since we don't need as many parameters.It doesn't seem like a huge savings until you are working with high resolution images that have thousands of pixels. The convolutional neural network algorithm's main purpose is to get data into forms that are easier to process without losing the features that are important for figuring out what the data represents. This also makes them great candidates for handling huge datasets.

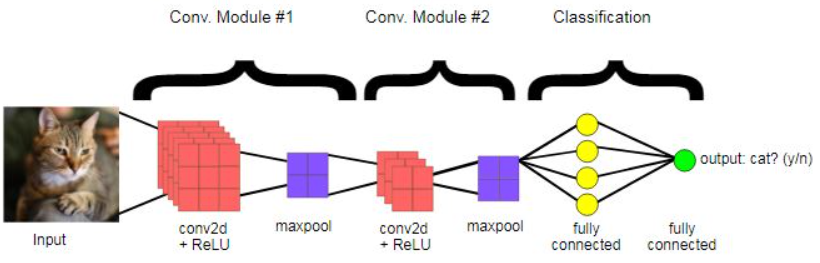
A big difference between a CNN and a regular neural network is that CNNs use convolutions to handle the math behind the scenes. A convolution is used instead of matrix multiplication in at least one layer of the CNN. Convolutions take to two functions and return a function. CNNs work by applying filters to your input data. What makes them so special is that CNNs are able to tune the filters as training happens. That way the results are fine-tuned in real time, even when you have huge data sets, like with images. Since the filters can be updated to train the CNN better, this removes the need for hand-created filters. That gives us more flexibility in the number of filters we can apply to a data set and the relevance of those filters. Using this algorithm, we can work on more sophisticated problems like face recognition.One of things that prevents a lot of problems from using CNNs is a lack of data. While networks can be trained with relatively few data points (~10,000 >), the more data there is available, the better tuned the CNN will be. Convolutional neural networks are based on neuroscience findings. They are made of layers of artificial neurons called nodes. These nodes are functions that calculate the weighted sum of the inputs and return an activation map. This is the convolution part of the neural network.

Each node in a layer is defined by its weight values. When you give a layer some data, like an image, it takes the pixel values and picks out some of the visual features.

When you're working with data in a CNN, each layer returns activation maps. These maps point out important features in the data set. If you gave the CNN an image, it'll point out features based on pixel values, like colors, and give you an activation function.

Usually with images, a CNN will initially find the edges of the picture. Then this slight definition of the image will get passed to the next layer. Then that layer will start detecting things like corners and color groups. Then that image definition will get passed to the next layer and the cycle continues until a prediction is made.

As the layers get more defined, this is called max pooling. It only returns the most relevant features from the layer in the activation map. This is what gets passed to each successive layer until you get the final layer.



#### INTRODUCTION TO CNN : WHY CNN?

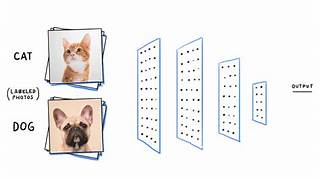
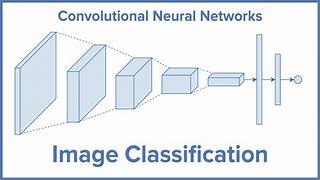
A **Convolutional Neural Network (CNN)** is a type of Deep Learning neural network architecture commonly used in Computer Vision. Computer vision is a field of Artificial Intelligence that enables a computer to understand and interpret the image or visual data.

When it comes to Machine Learning, Artificaial Neural Networks perform really well. Neural Networks are used in various datasets like images, audio, and text. Different types of Neural Networks are used for different purposes, for example for predicting the sequence of words we use [**Recurrent Neural Networks**](https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/) more precisely an LSTM, similarly for image classification we use Convolution Neural networks. In this blog, we are going to build a basic building block for CNN.

In a regular Neural Network there are three types of layers:

1. **Input Layers:** It’s the layer in which we give input to our model. The number of neurons in this layer is equal to the total number of features in our data (number of pixels in the case of an image).
2. **Hidden Layer:** The input from the Input layer is then fed into the hidden layer. There can be many hidden layers depending on our model and data size. Each hidden layer can have different numbers of neurons which are generally greater than the number of features. The output from each layer is computed by matrix multiplication of the output of the previous layer with learnable weights of that layer and then by the addition of learnable biases followed by activation function which makes the network nonlinear.
3. **Output Layer:** The output from the hidden layer is then fed into a logistic function like sigmoid or softmax which converts the output of each class into the probability score of each class.

The data is fed into the model and output from each layer is obtained from the above step is called [**feedforward**](https://www.geeksforgeeks.org/understanding-multi-layer-feed-forward-networks/), we then calculate the error using an error function, some common error functions are cross-entropy, square loss error, etc. The error function measures how well the network is performing. After that, we backpropagate into the model by calculating the derivatives. This step is called [**Backpropagation**](https://www.geeksforgeeks.org/backpropagation-in-data-mining/) which basically is used to minimize the loss.



[**Step 1: Convolution Operation**](http://www.superdatascience.com/blogs/deep-learning-a-z-convolutional-neural-networks-cnn-step-1-convolution-operation/)

The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out.

[**Step 1(b): ReLU Layer**](http://www.superdatascience.com/blogs/deep-learning-a-z-convolutional-neural-networks-cnn-step-1b-relu-layer/)

The second part of this step will involve the Rectified Linear Unit or ReLU. We will cover ReLU layers and explore how linearity functions in the context of Convolution Neural Networks.

Not necessary for understanding CNN's, but there's no harm in a quick lesson to improve your skills.

[**Step 2: Pooling**](http://www.superdatascience.com/blogs/deep-learning-a-z-convolutional-neural-networks-cnn-step-2-max-pooling/)

In this part, we'll cover pooling and will get to understand exactly how it generally works. Our nexus here, however, will be a specific type of pooling; max pooling. We'll cover various approaches, though, including mean (or sum) pooling. This part will end with a demonstration made using a visual interactive tool that will definitely sort the whole concept out for you.

[**Step 3: Flattening**](http://www.superdatascience.com/blogs/deep-learning-a-z-convolutional-neural-networks-cnn-step-3-flattening/)

This will be a brief breakdown of the flattening process and how we move from pooled to flattened layers when working with Convolution Neural Networks.

[**Step 4: Full Connection**](http://www.superdatascience.com/blogs/deep-learning-a-z-convolutional-neural-networks-cnn-step-4-full-connection/)

In this part, everything that we covered throughout the section will be merged together. By learning this, you'll get to envision a fuller picture of how Convolution Neural Networks operate and how the "neurons" that are finally produced learn the classification of images.

#### CNN FOR CLASSIFICATION:

Image classification involves the extraction of features from the image to observe some patterns in the dataset. Using an ANN for the purpose of image classification would end up being very costly in terms of computation since the trainable parameters become extremely large.For example, if we have a 50 X 50 image of a cat, and we want to train our traditional ANN on that image to classify it into a dog or a cat the trainable parameters become –  
(50\*50) \* 100 image pixels multiplied by hidden layer + 100 bias + 2 \* 100 output neurons + 2 bias = 2,50,302. We use filters when using CNNs. Filters exist of many different types according to their purpose.

CNNs are trained using a large dataset of labeled images, where the network learns to recognize patterns and features that are associated with specific objects or classes. Once trained, a CNN can be used to classify new images, or extract features for use in other applications such as object detection or image segmentation.

CNNs have achieved state-of-the-art performance on a wide range of image recognition tasks, including object classification, object detection, and image segmentation. They are widely used in computer vision, image processing, and other related fields, and have been applied to a wide range of applications, including self-driving cars, medical imaging, and security systems.

* A convolutional neural network, or CNN, is a deep learning neural network sketched for processing structured arrays of data such as portrayals.
* CNN are very satisfactory at picking up on design in the input image, such as lines, gradients, circles, or even eyes and faces.
* This characteristic that makes convolutional neural network so robust for computer vision.
* CNN can run directly on a underdone image and do not need any preprocessing.
* A convolutional neural network is a feed forward neural network, seldom with up to 20.
* The strength of a convolutional neural network comes from a particular kind of layer called the convolutional layer.
* CNN contains many convolutional layers assembled on top of each other, each one competent of recognizing more sophisticated shapes.
* With three or four convolutional layers it is viable to recognize handwritten digits and with 25 layers it is possible to differentiate human faces.
* The agenda for this sphere is to activate machines to view the world as humans do, perceive it in a alike fashion and even use the knowledge for a multitude of duty such as image and video recognition, image inspection and classification, media recreation, recommendation systems, natural language processing, etc.

**Convolutional Neural Network Design :**

* The construction of a convolutional neural network is a multi-layered feed-forward neural network, made by assembling many unseen layers on top of each other in a particular order.
* It is the sequential design that give permission to CNN to learn hierarchical attributes.
* In CNN, some of them followed by grouping layers and hidden layers are typically convolutional layers followed by activation layers.
* The pre-processing needed in a ConvNet is kindred to that of the related pattern of neurons in the human brain and was motivated by the organization of the Visual Cortex.

convolutional layer is the first layer of a convolutional network. While convolutional layers can be followed by additional convolutional layers or pooling layers, the fully-connected layer is the final layer. With each layer, the CNN increases in its complexity, identifying greater portions of the image. Earlier layers focus on simple features, such as colors and edges. As the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object.

**Convolutional layer:**

The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs. It requires a few components, which are input data, a filter, and a feature map. Let’s assume that the input will be a color image, which is made up of a matrix of pixels in 3D. This means that the input will have three dimensions—a height, width, and depth—which correspond to RGB in an image. We also have a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution.

The feature detector is a two-dimensional (2-D) array of weights, which represents part of the image. While they can vary in size, the filter size is typically a 3x3 matrix; this also determines the size of the receptive field. The filter is then applied to an area of the image, and a dot product is calculated between the input pixels and the filter. This dot product is then fed into an output array. Afterwards, the filter shifts by a stride, repeating the process until the kernel has swept across the entire image. The final output from the series of dot products from the input and the filter is known as a feature map, activation map, or a convolved feature.

Note that the weights in the feature detector remain fixed as it moves across the image, which is also known as parameter sharing. Some parameters, like the weight values, adjust during training through the process of backpropagation and gradient descent. However, there are three hyperparameters which affect the volume size of the output that need to be set before the training of the neural network begins. These include:

1. The**number of filters**affects the depth of the output. For example, three distinct filters would yield three different feature maps, creating a depth of three.

2. **Stride** is the distance, or number of pixels, that the kernel moves over the input matrix. While stride values of two or greater is rare, a larger stride yields a smaller output.

3. **Zero-padding** is usually used when the filters do not fit the input image. This sets all elements that fall outside of the input matrix to zero, producing a larger or equally sized output. There are three types of padding:

* **Valid padding:** This is also known as no padding. In this case, the last convolution is dropped if dimensions do not align.
* **Same padding:** This padding ensures that the output layer has the same size as the input layer
* **Full padding:**This type of padding increases the size of the output by adding zeros to the border of the input.

After each convolution operation, a CNN applies a Rectified Linear Unit (ReLU) transformation to the feature map, introducing nonlinearity to the model.

Additional convolutional layer

As we mentioned earlier, another convolution layer can follow the initial convolution layer. When this happens, the structure of the CNN can become hierarchical as the later layers can see the pixels within the receptive fields of prior layers.  As an example, let’s assume that we’re trying to determine if an image contains a bicycle. You can think of the bicycle as a sum of parts. It is comprised of a frame, handlebars, wheels, pedals, et cetera. Each individual part of the bicycle makes up a lower-level pattern in the neural net, and the combination of its parts represents a higher-level pattern, creating a feature hierarchy within the CNN. Ultimately, the convolutional layer converts the image into numerical values, allowing the neural network to interpret and extract relevant patterns.

**Pooling layer:**

Pooling layers, also known as downsampling, conducts dimensionality reduction, reducing the number of parameters in the input. Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights. Instead, the kernel applies an aggregation function to the values within the receptive field, populating the output array. There are two main types of pooling:

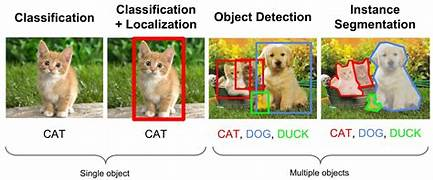
* **Max pooling:** As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.
* **Average pooling:** As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.

While a lot of information is lost in the pooling layer, it also has a number of benefits to the CNN. They help to reduce complexity, improve efficiency, and limit risk of overfitting.

Fully-connected layer

The name of the full-connected layer aptly describes itself. As mentioned earlier, the pixel values of the input image are not directly connected to the output layer in partially connected layers. However, in the fully-connected layer, each node in the output layer connects directly to a node in the previous layer.

This layer performs the task of classification based on the features extracted through the previous layers and their different filters. While convolutional and pooling layers tend to use ReLu functions, FC layers usually leverage a softmax activation function to classify inputs appropriately, producing a probability from 0 to 1.

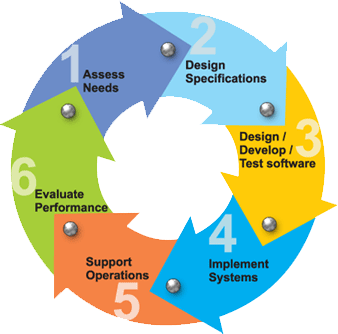


#### 

#### ALGORITHM FOR CNN BASED CLASSIFICATION:

* Import the dataset.
* Explore the data to figure out what they look like.
* Pre-process the data by using techniques.
* Split the data into attributes and labels.
* Divide the data into training and testing sets.
* Train the CNN algorithm.
* Make some predictions.
* Evaluate the results of the algorithm.

#### SOFTWARE DEVELOPMENT LIFE CYCLE :

There is various software development approaches defined and designed which are used/employed during development process of software, these approaches are also referred as "Software Development Process Models".

#### REQUIREMENTS

Business requirements are gathered in this phase. This phase is the main focus of the project managers and stake holders. Meeting with managers, stake holders and users are held in order to determine the requirements. Who is going to use the system? How will they use the system? What data should be input into the system? What the should be output by the system? These are general questions that get answered during a requirements gathering phase. This produces a nice big list of functionality that the system should provide, which describes functions the system should performs, business logic that processes data, what data is stored and used by the system, and how the user interface should work. The overall result is the system as a whole and how it performs, not how it is actually going to do it.

#### DESIGN

The software system design is produced from the results of the requirements phase. Architects have the ball in their court during this phase and this is the phase in which their focus lies. This is where the details on how the system will work is produced. Architecture, including hardware and software, communication, software design (UML is produced here) are all part of the deliverables of a design phase.

#### IMPLEMENTATION

Code is produced from the deliverables of the design phase during implementation, and this is the longest phase of the software development life cycle. For a developer, this is the main focus of the life cycle because this is where the code is produced. Implementation my overlap with both the design and testing phases. Many tools exists (CASE tools) to actually automate the production of code using information gathered and produced during the design phase.

#### TESTING

During testing, the implementation is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. Unit tests and system/acceptance tests are done during this phase. Unit tests act on a specific component of the system, while system tests act on the system as a whole. So in a nutshell, that is a very basic overview of the general software development life cycle model. Now let’s delve into some of the traditional and widely used variations.

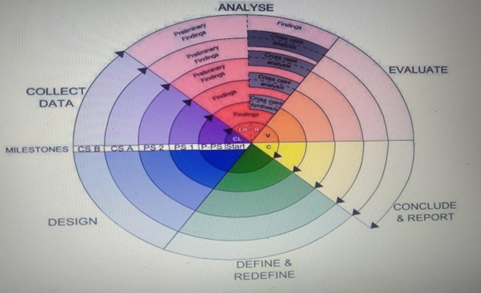
#### STUDY OF THE SYSTEM:

In the flexibility of uses the interface has been developed a graphics concepts in mind, associated through a browser interface. The GUI’s at the top level has been categorized as follows:

* + 1. Administrative User Interface Design
    2. The Operational and Generic User Interface Design

The administrative user interface concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. The Interface helps the administration with all the transactional states like data insertion, data deletion, and data updating along with executive data search capabilities.

The operational and generic user interface helps the users upon the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information helps the ordinary users in managing their own information in a customized manner as per the assisted flexibilities



**2.5 SYSTEM ANALYSIS:**

The Systems Development Life Cycle (SDLC), or Software Development Life Cycle in [systems](http://en.wikipedia.org/wiki/Systems_engineering) [engineering,](http://en.wikipedia.org/wiki/Systems_engineering) [information systems](http://en.wikipedia.org/wiki/Systems_engineering) and [software engineering,](http://en.wikipedia.org/wiki/Software_engineering) is the process of creating or altering systems, and the models and [methodologies](http://en.wikipedia.org/wiki/Methodologies) that people use to develop these systems.

In software engineering the SDLC concept underpins many kinds of [software development methodologies.](http://en.wikipedia.org/wiki/Software_development_methodologies) These methodologies form the framework for planning and controlling the creation of an information system the [software development process.](http://en.wikipedia.org/wiki/Software_development_process)

#### SOFTWARE MODEL OR ARCHITECTURAL ANALYSIS:

### Structured project management techniques (such as an SDLC) enhance management’s control over projects by dividing complex tasks into manageable sections. A software life cycle model is either a descriptive or prescriptive characterization of how software is or should be developed. But none of the SDLC models discuss the key issues like Change management, Incident management and Release management processes within the SDLC process, but, it is addressed in the overall project management. In the proposed hypothetical model, the concept of user-developer interaction in the conventional SDLC model has been converted into a three dimensional model which comprises of the user, owner and the developer. In the proposed hypothetical model, the concept of user-developer interaction in the conventional SDLC model has been converted into a three dimensional model which comprises of the user, owner and the developer. The one size fits all‖ approach to applying SDLC methodologies is no longer appropriate. We have made an attempt to address the above mentioned defects by using a new hypothetical model for SDLC described elsewhere. The drawback of addressing these management processes under the overall project management is missing of key technical issues pertaining to software development process that is, these issues are talked in the project management at the surface level but not at the ground level.

### What is SDLC?

SDLC stands for Software Development Life Cycle. A Software Development Life Cycle is essentially a series of steps, or phases, that provide a model for the development and lifecycle management of an application or piece of software. SDLC is the process consisting of a series of planned activities to develop or alter the software products.

### Benefits of the SDLC Process:

The intent of a SDLC process it to help produce a product that is cost-efficient, effective, and of high quality. Once an application is created, the SDLC maps the proper deployment and decommissioning of the software once it becomes a legacy. The SDLC methodology usually contains the following stages: Analysis (requirements and design), construction, testing, release, and maintenance (response). Veracode makes it possible to integrate automated security testing into the SDLC process through use of its cloud based platform.

1.Requirements Gathering:

In this phase we gather all the requirements from the client, i.e. what are the client expected input, output……

2.Analysis:

In this phase based upon the client requirements we prepare one documentation is called “High Level Design Document”. It contains Abstract, Functional Requirements, Non Functional Requirements, Existing System, Proposed System, SRS,………

3.Design:

It is difficult to understand the High Level Design Document for all the members, so to understand easily we use “Low Level Design Document”. To design this document we use UML (Unified Modeling Language). In this we have Use case, Sequence, Collaboration……..

1. Coding:

In this phase we develop the coding module by module. After developing all the modules we integrate them.

1. Testing:

After developing we have to check weather client requirements are satisfied or not. If not we are again going to develop.

1. Implementation:

In testing phase if client requirements are satisfied, we go for implementation. i.e. we need to deploy the application in some server.

1. Maintenance:

After deployment, if at all any problems come from the client side; we are providing maintenance for that application.

### 

# CHAPTER-3

# REQUIREMENT SPECIFICATION

## CHAPTER-3

## REQUIREMENT SPECIFICATIONS

* 1. **FUNCTIONAL REQUIREMENTS**

REQUIREMENT ANALYSIS

The project involved analyzing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigations from one screen to the other well ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the browser version had to be chosen so that it is compatible with most of the Browsers.

**REQUIREMENT SPECIFICATION**

**Functional Requirements**

* Graphical User interface with the User.

**Software Requirements**

For developing the application the following are the Software Requirements:

1. Python
2. Django

**Operating Systems supported**

1. Windows 7
2. Windows XP
3. Windows 8
4. Windows 10

**Technologies and Languages used to Develop**

1. Python
2. Jupyter Notebook
3. Pycharm

**Debugger and Emulator**

* Any Browser (Particularly Chrome)

**Operating Systems Supported :**

* windows 7 , windows 8 , windows 10 , windows XP.

**Technologies and languages used to Develop :**

* Python , Jupyter Notebook , PyCharm.

**Software Requirements :**

* Python

**Hardware Requirements**

For developing the application the following are the Hardware Requirements:

* Processor: Pentium IV or higher
* RAM: 256 MB
* Space on Hard Disk: minimum 512MB.

**INPUT DESIGN :**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

1. What data should be given as input?
2. How the data should be arranged or coded?
3. The dialog to guide the operating personnel in providing input.
4. Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES**

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.
* Operating Systems Supported : windows 7 , windows 8 , windows 10 , windows XP.
* Technologies and languages used to Develop : Python , JupyterNotebook , Pycharm.
* Software Requirements : Python
* Hardware Requirements : Pentium 4 or higher processor , RAM : 256 Mb , HardDisk : min 512 Mb
* Functional Requirements : Graphical user interface with user

# CHAPTER – 4 LANGUAGES OF IMPLEMENTATION

## CHAPTER – 4

## LANGUAGES OF IMPLEMENTATION

* 1. **INTRODUCTION TO SCRIPT**

#### What is a script?

Up to this point, I have concentrated on the interactive programming capability of Python. This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode.

#### Scripts are reusable?

Basically, a script is a text file containing the statements that comprise a Python program. Once you have created the script, you can execute it over and over without having to retype it each time.

#### Scripts are editable

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor. Then you can execute each of the individual versions. In this way, it is easy to create different programs with a minimum amount of typing.

#### You will need a text editor

Just about any text editor will suffice for creating Python script files.

You can use Microsoft Notepad, Microsoft WordPad, Microsoft Word, or just about any word processor if you want to.

DIFFERENCE BETWEEN A SCRIPT AND A PROGRAM

#### Script:

Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or byte code, whereas the applications they control are traditionally compiled to native machine code.

Program:

The program has an executable form that the computer can use directly to execute the instructions. The same program in its human-readable source code form, from which executable programs are derived(e.g., compiled)

## PYTHON

What is Python?

Chances you are asking yourself this. You may have found this book because you want to learn to program but don’t know anything about programming languages. Or you may have heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to “big name” languages. Hopefully I can explain it for you.

PYTHON CONCEPTS

If you’re not interested in the how’s and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it’s a great one to start programming with.

* + - Open source general-purpose language.
    - Object Oriented, Procedural, Functional
    - Easy to interface with C/ObjC/Java/Fortran
    - Easy-ish to interface with C++ (via SWIG)

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

* **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

### HISTORY OF PYTHON

* Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.
* Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol- 68, SmallTalk, and Unix shell and other scripting languages.
* Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).
* Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

### PYTHON FEATURES

* **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
* **A broad standard library** − Python's bulk of the library is very portable and cross- platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases** − Python provides interfaces to all major commercial databases.
* **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* **Scalable** − Python provides a better structure and support for large programs than shell scripting. Apart from the above-mentioned features, Python has a big list of good features, few are listed below:
  + It supports functional and structured programming methods as well as OOP.
  + It can be used as a scripting language or can be compiled to byte-code for building large applications.
  + It provides very high-level dynamic data types and supports dynamic type checking.
  + IT supports automatic garbage collection.
  + It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

### DYNAMIC V/S STATIC

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of “thing” each data value is.

**For example**, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a “float” type. This tells the compiler that the only data that can be used for that variable must be a floating point number, i.e. a number with a decimal point. If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating point number) you need in your program. With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment).If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double. With Python, it doesn’t matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values.

**For example**, say you are dividing two numbers. One is a floating point number and one is an integer. Python realizes that it’s more accurate to keep track of decimals so it automatically calculates the result as a floating point number.

### VARIABLES

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

### STANDARD DATA TYPES

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types:

* Numbers
* String
* List
* Tuple
* Dictionary

### PYTHON NUMBERS

Number data types store numeric values. Number objects are created when you assign a value to them.

### PYTHON STRINGS

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of strings can be taken using the

slice operator ([ ] and [:] ) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

### PYTHON LISTS

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type.

The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (\*) is the repetition operator.

### PYTHON TUPLES

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.

The main differences between lists and tuples are: Lists are enclosed in brackets ( [ ] ) and their elements and size can be changed, while tuples are enclosed in parentheses ( ( ) ) and cannot be updated. Tuples can be thought of as **read-only** lists.

### PYTHON DICTIONARY

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or

strings. Values, on the other hand, can be any arbitrary Python object. Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

### DIFFERENT MODES IN PYTHON

Python has two basic modes:

.

#### Normal :

The normal mode is the mode where the scripted and finished .py files are run in the Python interpreter.

#### Interactive:

Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole.

### PYTHON LIBRARIES ARE

#### Requests

The most famous http library written by kennethreitz. It’s a must have for every python developer.

#### Scarpy

If you are involved in web scraping then this is a must have library for you. After using this library you won’t use any other.

#### wxPython

A gui toolkit for python. I have primarily used it in place of tkinter. You will really love it.

#### Pillow

A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.

#### SQLAlchemy

A database library. Many love it and many hate it. The choice is yours.

#### BeautifulSoup

I know it’s slow but this xml and html parsing library is very useful for beginners.

#### Twisted

The most important tool for any network application developer. It has a very beautiful api and is used by a lot of famous python developers.

#### NumPy

How can we leave this very important library ? It provides some advance math functionalities to python.

#### SciPy

When we talk about NumPy then we have to talk about scipy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.

#### Matplotlib

A numerical plotting library. It is very useful for any data scientist or any data analyzer.

#### Pygame

Which developer does not like to play games and develop them ? This library will help you achieve your goal of 2d game development.

#### Pyglet

A 3d animation and game creation engine. This is the engine in which the famous [python](https://github.com/fogleman/Minecraft) [port](https://github.com/fogleman/Minecraft) of mine-craft was made.

#### pyQT

A GUI toolkit for python. It is my second choice after wxpython for developing GUI’s for my python scripts.

#### pyGtk

Another python GUI library. It is the same library in which the famous Bittorrent client is created.

#### Scapy

A packet sniffer and analyzer for python made in python.

#### Pywin32

A python library which provides some useful methods and classes for interacting with windows.

#### nltk

Natural Language Toolkit – I realize most people won’t be using this one, but it’s generic enough. It is a very useful library if you want to manipulate strings. But it’s capacity is beyond that. Do check it out.

#### nose

A testing framework for python. It is used by millions of python developers. It is a must have if you do test driven development.

#### SumPy

SymPy can do algebraic evaluation, differentiation, expansion, complex numbers, etc. It is contained in a pure Python distribution.

#### IPython

It just can’t stress enough how useful this tool is. It is a python prompt on steroids. It has completion, history, shell capabilities, and a lot more. Make sure that you take a look at it.

### NUMPY

NumPy’s main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called axes. The number of axes is rank.

* Offers Matlab-ish capabilities within Python
* Fast array operations
* 2D arrays, multi-D arrays, linear algebra etc.

**Matplotlib :** High quality plotting library.

### PYTHON MODULES

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library.

To support this, Python has a way to put definitions in a file and use them in a script or in an interactive instance of the interpreter. Such a file is called a module; definitions from a module can be imported into other modules or into the main module.

### TESTING CODE

* As indicated above, code is usually developed in a file using an editor.
* To test the code, import it into a Python session and try to run it.
* Usually there is an error, so you go back to the file, make a correction, and test again.
* This process is repeated until you are satisfied that the code works.
* The entire process is known as the development cycle.
* There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid.
* This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

### FUNCTIONS IN PYTHON

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have need to perform a certain type of calculation many times, then define a function**.**

You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub-program and called when needed. That means that a function is a piece of code written to carry out a specified task.

To carry out that specific task, the function might or might not need multiple inputs. When the task is carried out, the function can or cannot return one or more values. There are three types of functions in python:

help() ,min() ,print().

### PYTHON NAMESPACE

Generally speaking, a **namespace** (sometimes also called a context) is a naming system for making names unique to avoid ambiguity. Everybody knows a name-spacing system from daily life, i.e. the naming of people in first name and family name (surname).

**An example** is a network: each network device (workstation, server, printer, ...) needs a unique name and address. Yet another example is the directory structure of file systems.

* + The same file name can be used in different directories, the files can be uniquely accessed via the pathnames.Many programming languages use namespaces or contexts for identifiers. An identifier defined in a namespace is associated with that namespace.
  + This way, the same identifier can be independently defined in multiple namespaces. (Like the same file names in different directories) Programming languages, which support namespaces, may have different rules that determine to which namespace an identifier belongs.
  + Namespaces in Python are implemented as Python dictionaries, this means it is a mapping from names (keys) to objects (values). The user doesn't have to know this to write a Python program and when using namespaces.

#### Some namespaces in Python:

**global names** of a module

**local names** in a function or method invocation

**built-in names**: this namespace contains built-in functions (e.g. abs(), cmp(), ...) and built-in exception names

### GARBAGE COLLECTION

Garbage Collector exposes the underlying memory management mechanism of Python, the automatic garbage collector. The module includes functions for controlling how the collector operates and to examine the objects known to the system, either pending collection or stuck in reference cycles and unable to be freed.

# CHAPTER – 5 SYSTEM DESIGN

## CHAPTER–5

## SYSTEM DESIGN

### INTRODUCTION

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer’s goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word “Quality”. Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer’s view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage. The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed, design takes us toward how to satisfy the needs. The design of a system is perhaps the most critical factor affection the quality of the software; it has a major impact on the later phase, particularly testing, maintenance. The output of this phase is the design document. This document is similar to a blueprint for the solution and is used later during implementation, testing and maintenance. The design activity is often divided into two separate phases System Design and Detailed Design.

System Design also called top-level design aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results. At the end of the system design all the major data structures, file formats, output formats, and the major modules in the system and their specifications are decided.

During, Detailed Design, the internal logic of each of the modules specified in system design is decided. During this phase, the details of the data of a module is usually specified in a high-level design description

language, which is independent of the target language in which the software will eventually be implemented.

In system design the focus is on identifying the modules, whereas during detailed design the focus is on designing the logic for each of the modules. In other works, in system design the attention is on what components are needed, while in detailed design how the components can be implemented in software is the issue.

Design is concerned with identifying software components specifying relationships among components. Specifying software structure and providing blue print for the document phase. Modularity is one of the desirable properties of large systems. It implies that the system is divided into several parts. In such a manner, the interaction between parts is minimal clearly specified.

During the system design activities, Developers bridge the gap between the requirements specification, produced during requirements elicitation and analysis, and the system that is delivered to the user.

Design is the place where the quality is fostered in development. Software design is a process through which requirements are translated into a representation of software.

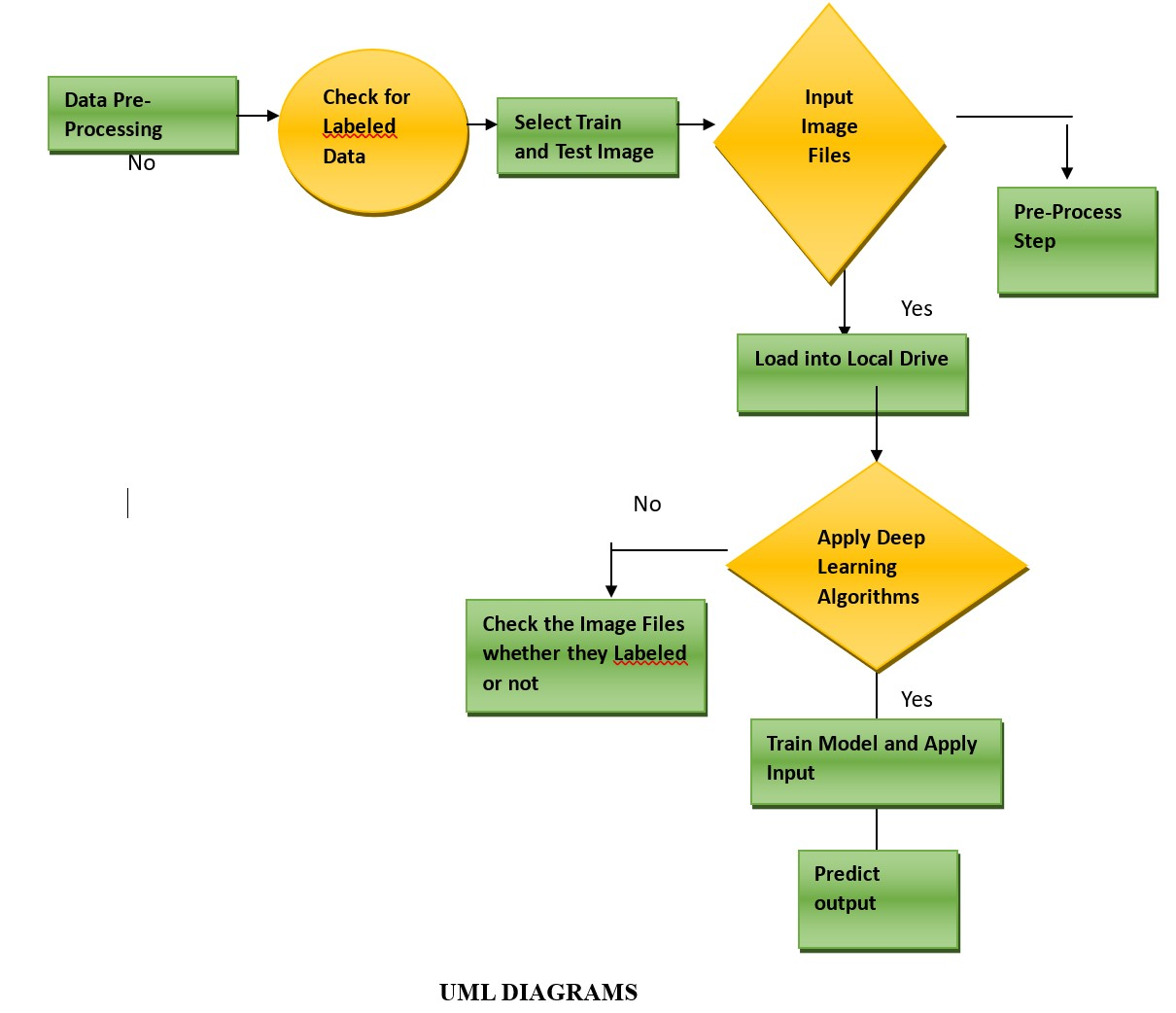
### DATA FLOW DIAGRAMS

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail. DFDs are the model of the proposed system. They clearly should show the requirements on which the new system should be built. Later during design activity this is taken as the basis for drawing the system’s structure charts. The Basic Notation used to create a DFD’s are as follows:
5. **Dataflow:** Data move in a specific direction from an origin to a destination.

**2. Process:** People, procedures, or devices that use or produce (Transform) Data. The physical component is not identified.

**3.Source:** External sources or destination of data, which may be People, programs, organizations or other entities.

**4.Data Store:** Here data are stored or referenced by a process in the System.



UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

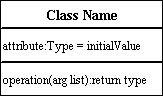
The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

#### UML class diagram?

Class diagrams are the backbone of almost every object-oriented method including UML. They describe the static structure of a system.

#### Basic class diagrams symbols and notations:

Classes represent an abstraction of entities with common characteristics. Associations represent the relationships between classes.Illustrate classes with rectangles divided into compartments. Place the name of the class in the first partition (centered, bolded, and capitalized), list the attributes in the second partition, and write operations into the third.



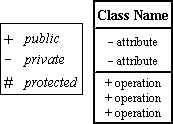
#### Active Class

Active classes initiate and control the flow of activity, while passive classes store data and serve other classes. Illustrate active classes with a thicker border.

Active Class

#### Visibility

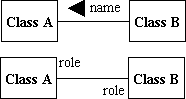
Use visibility markers to signify who can access the information contained within a class. Private visibility hides information from anything outside the class partition. Public visibility allows all other classes to view the marked information. Protected visibility allows child classes to access information they inherited from a parent class. [.](http://www.smartdraw.com/resources/tutorials/Text-and-Tables)



#### Associations

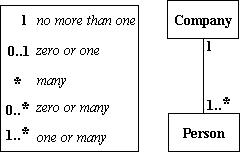
Associations represent static relationships between classes. Place association names above, on, or below the association line. Use a filled arrow to indicate the direction of the relationship. Place roles near the end of an association. Roles represent the way the two classes see each other.

**Note**: It's uncommon to name both the association and the class roles.



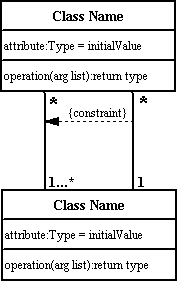
#### Multiplicity

Place multiplicity notations near the ends of an association. These symbols indicate the number of instances of one class linked to one instance of the other class. For example, one company will have one or more employees, but each employee works for one company only.



#### Constraint

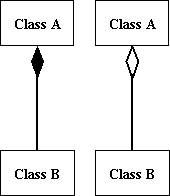
Place constraints inside curly braces {}.

http://wc1.smartdraw.com/resources/tutorials/images/uml_constraint.gif*Simple Constraint*

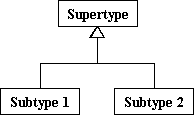
#### Composition and Aggregation

Composition is a special type of aggregation that denotes a strong ownership between Class A, the whole, and Class B, its part. Illustrate **composition** with a filled diamond. Use a hollow diamond to represent a simple **aggregation** relationship, in which the "whole" class plays a more important role than the "part"

class, but the two classes are not dependent on each other. The diamond end in both a composition and aggregation relationship points toward the "whole" class or the aggregate

**Generalization**

Generalization is another name for inheritance or an "is a" relationship. It refers to a relationship between two classes where one class is a specialized version of another. For example, Honda is a type of car. So the class Honda would have a generalization relationship with the class car.

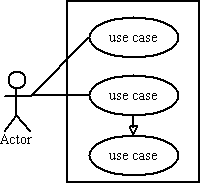


In real life coding examples, the difference between inheritance and aggregation can be confusing. If you have an aggregation relationship, the aggregate (the whole) can access only the PUBLIC functions of the part class. On the other hand, inheritance allows the inheriting class to access both the PUBLIC and PROTECTED functions of the super class.

#### Use Case Diagram

Use case diagrams model the functionality of a system using actors and use cases. Use cases are services or functions provided by the system to its users.

#### Basic Use Case Diagrams Symbols and Notations System

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system’s boundaries.

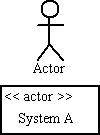
**Use Case**

Use CaseDraw use cases using ovals. Label with ovals with verbs that represent the system's functions.

#### Actors

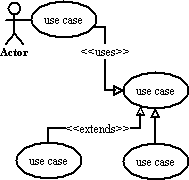
#### 

#### Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.



#### Relationships

Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain usecase.



#### Sequence diagram

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time.

#### Basic Sequence Diagram Symbols and Notation

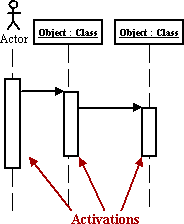
#### Class roles

Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles but, don't list object attributes.

Class roles

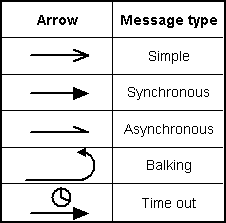
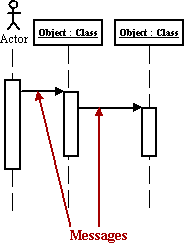
#### Activation

Activation boxes represent the time an object needs to complete a task.



#### Messages

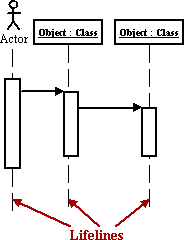
Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.



*Various types of Sequence and Collaboration Diagrams*

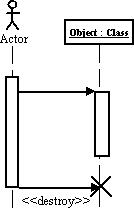
#### Lifelines

Lifelines are vertical dashed lines that indicate the object's presence over time.

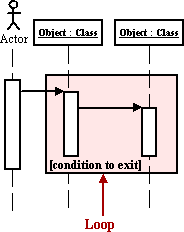


#### Destroying Objects

Objects can be terminated early using an arrow labeled "<< destroy >>" that points to an X.



#### Loops

 A repetition or loop within a sequence diagram is depicted as a rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets [ ].

#### Collaboration Diagram

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.

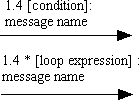
#### Basic Collaboration Diagram Symbols and Notations Class roles

Class rolesClass roles describe how objects behave. Use the UML object symbol to illustrate class roles, but don't list object attributes.

#### Association rules

Association rolesAssociation roles describe how an association will behave given a particular situation. You can draw association roles using simple lines labeled with stereotypes.

#### Messages

Unlike sequence diagrams, collaboration diagrams do not have an explicit way to denote time and instead number messages in order of execution. Sequence numbering can become nested using the Dewey decimal system. For example, nested messages under the first message are labeled 1.1, 1.2, 1.3, and so on. The a condition for a message is usually placed in square brackets immediately following the sequence number. Use a \* after the sequence number to indicate a loop.

#### Activity Diagram

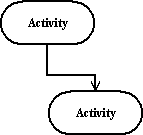
An activity diagram illustrates the dynamic nature of a system by modeling the flow of control from activity to activity. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and internal operation. Because an activity diagram is a special kind of state chart diagram, it uses some of the same modeling conventions.

#### Basic Activity Diagram Symbols and Notations Action States

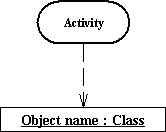
Action statesAction states represent the non interruptible actions of objects. You can draw an action state in Smart Draw using a rectangle with rounded corners.

#### Action Flow

Action flow arrows illustrate the relationships among action states.



#### Object Flow

Object flow refers to the creation and modification of objects by activities. An object flow arrow from an action to an object means that the action creates or influences the object. An object flow arrow from an object to an action indicates that the action state uses the object.

#### Initial State

A filled circle followed by an arrow represents the initial action state.

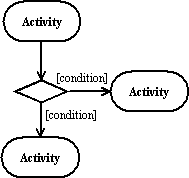
Initial State

#### Final State

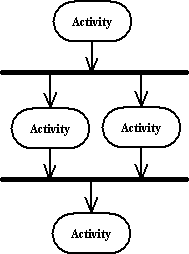
An arrow pointing to a filled circle nested inside another circle represents the final action state.

Final State

#### Branching

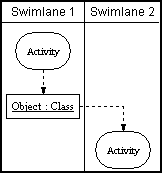
A diamond represents a decision with alternate paths. The outgoing alternates should be labeled with a condition or guard expression. You can also label one of the paths "else."

#### Synchronization

A synchronization bar helps illustrate parallel transitions. Synchronization is also called forking and joining.

#### Swimlanes

Swimlanes group related activities into one column.



#### State Chart Diagram

A state chart diagram shows the behavior of classes in response to external stimuli. This diagram models the dynamic flow of control from state to state within a system.

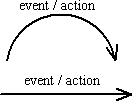
#### Basic State Chart Diagram Symbols and Notations States

States represent situations during the life of an object. You can easily illustrate a state in Smart Draw by using a rectangle with rounded corners.

States

#### Transition

A solid arrow represents the path between different states of an object. Label the transition with the event that triggered it and the action that results from it.



#### Initial State

A filled circle followed by an arrow represents the object's initial state.

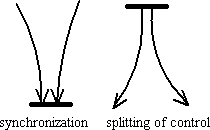
Initial State

#### Final State

An arrow pointing to a filled circle nested inside another circle represents the object's final state.

Final State

#### Synchronization and Splitting of Control

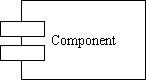
A short heavy bar with two transitions entering it represents a synchronization of control. A short heavy bar with two transitions leaving it represents a splitting of control that creates multiple states.

#### UML Component Diagram

A component diagram describes the organization of the physical components in a system.

#### Basic Component Diagram Symbols and Notations Component

A component is a physical building block of the system. It is represented as a rectangle with tabs. [Learn](http://www.smartdraw.com/resources/tutorials/Objects) [how to resize grouped objects like components.](http://www.smartdraw.com/resources/tutorials/Objects)

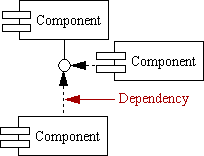


#### Interface

InterfaceAn interface describes a group of operations used or created by components.

#### Dependencies

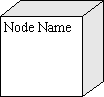
Draw dependencies among components using dashed arrows. [Learn about line styles in Smart-Draw.](http://www.smartdraw.com/resources/tutorials/Lines)



#### UML Deployment Diagram

Deployment diagrams depict the physical resources in a system including nodes, components, and connections.

#### Basic development diagram and notations

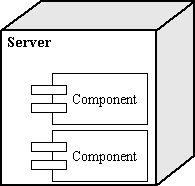
A node is a physical resource that executes code components. [Learn how to resize grouped objects like](http://www.smartdraw.com/resources/tutorials/Objects) [nodes.](http://www.smartdraw.com/resources/tutorials/Objects)

#### Association

AssociationAssociation refers to a physical connection between nodes, such as Ethernet. [Learn how to connect two](http://www.smartdraw.com/resources/tutorials/Lines) [nodes.](http://www.smartdraw.com/resources/tutorials/Lines)

#### Components and Nodes

Place components inside the node that deploys them.



## UML Diagrams Overview

UML combines best techniques from data modeling (entity relationship diagrams), business modeling (work flows), object modeling, and component modeling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies. UML has synthesized the notations of the Booch method, the Object-modeling technique (OMT) and Object-oriented software engineering (OOSE) by fusing them into a

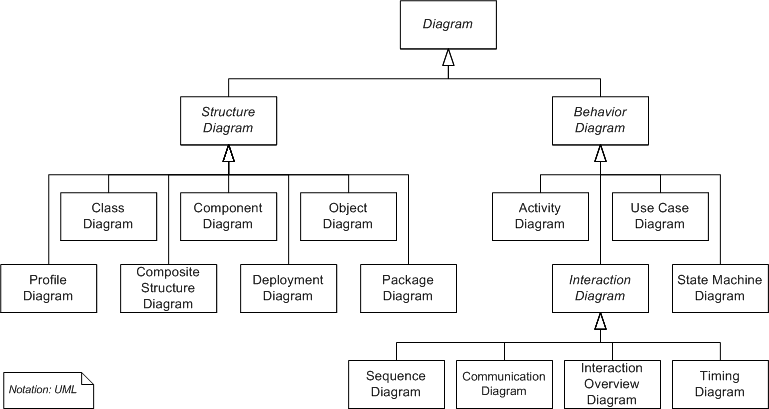
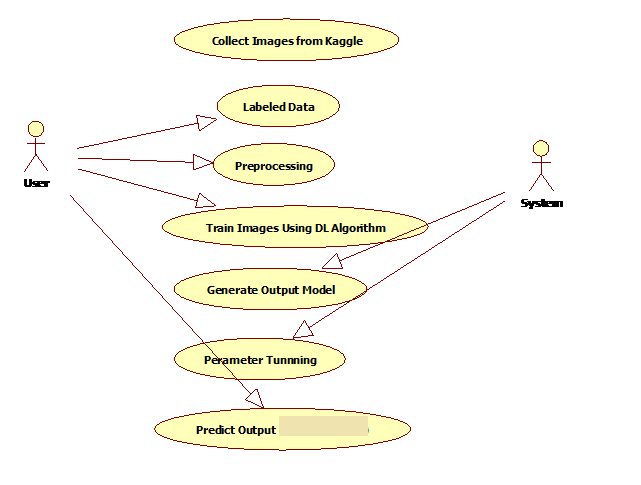


Fig 7: UML diagrams Overview

single, common and widely usable modeling language. UML aims to be a standard modeling language which can model concurrent and distributed systems.

#### Use Case Diagram



**Fig 8: Use Case Diagram**

**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information

.

#### 

#### Fig:8-Class Diagram

#### Sequence Diagram

****A **sequence diagram** in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows, as parallel vertical lines ("lifelines"), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphi

**Fig 9: Sequence Diagram**

#### Activity Diagram

 Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

**Fig 10 : Activity Diagram**

#### System Architecture:

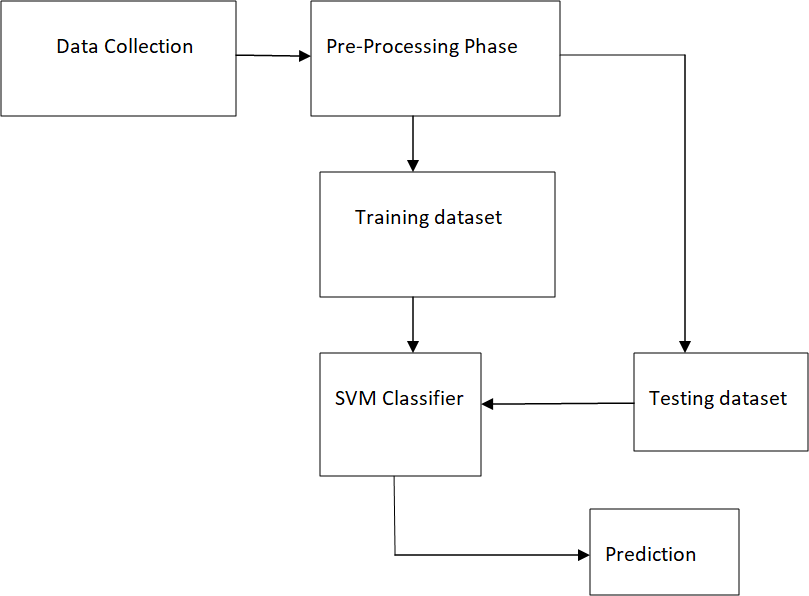


Fig 11: System Architecture

Considering input dataset as lung cancer patient dataset which is in format of comma separated values which in the form of spreadsheets. Input data is given to machine for pre-processing. Pre-processing is nothing but a process of processing data and making it suitable for a machine learning model.

Steps involved in Pre-processing:

* Data Cleaning
* Data Partitioning
* Feature Scaling

Data Cleaning:

Cleaning can be done in many ways some ways are used to remove the unwanted row, or drop the unwanted column. Another way to calculate the mean for a row or column where we find the missing value and substitute that value in the place of missing value.

Data Partitioning:

It is done to split the dataset into two subsets that is train data and test data. The reason for splitting the dataset is that suppose if we given training to our machine learning model by a dataset and we completely test it by different set. Then it creates difficulty to our model. Training data is 80% and Test data is 20%.

Future Scaling:

It is the final step in pre-processing phase. It is one of the technique to standardize the independent variables of dataset in a specific range.

After, the completion of Pre-processing we will get the training dataset which is used to train the machine by using Support Vector Machine Classifier. SVM will generates some classification rules on training dataset and the test dataset is given from pre-processing stage to SVM based on the generation of classification rules it is used the predict the classification output. The classification report is used to generate the precision, re-call, F1-score, support.

# CHAPTER – 6 IMPLEMENTATION

## CHAPTER-6

## IMPLEMENTATION

* 1. **Screenshots**

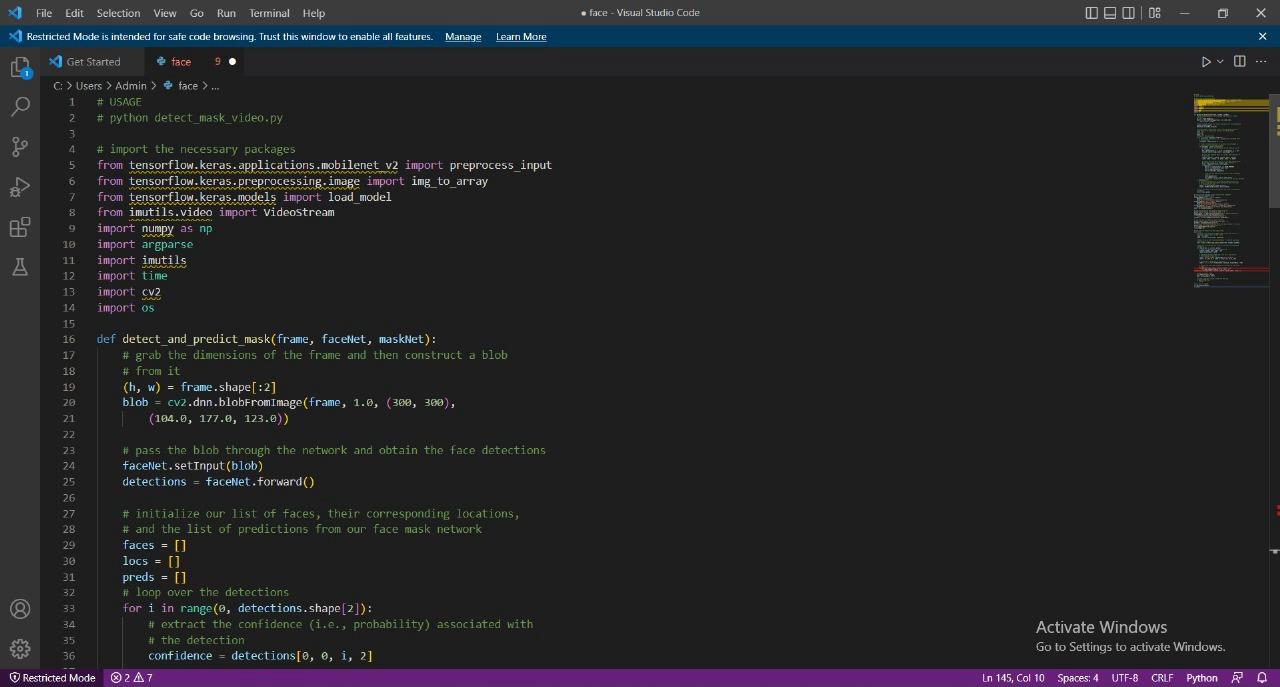


Fig 11: Code Screenshot

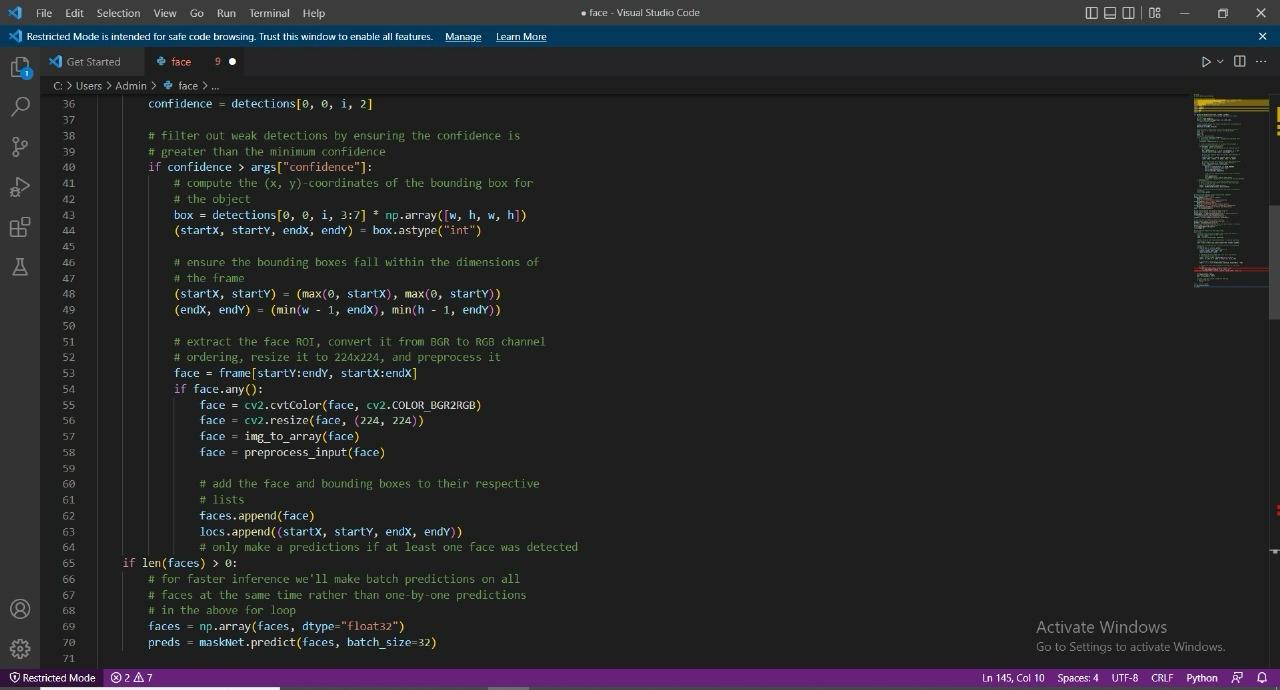


Fig 12: Code Screenshots

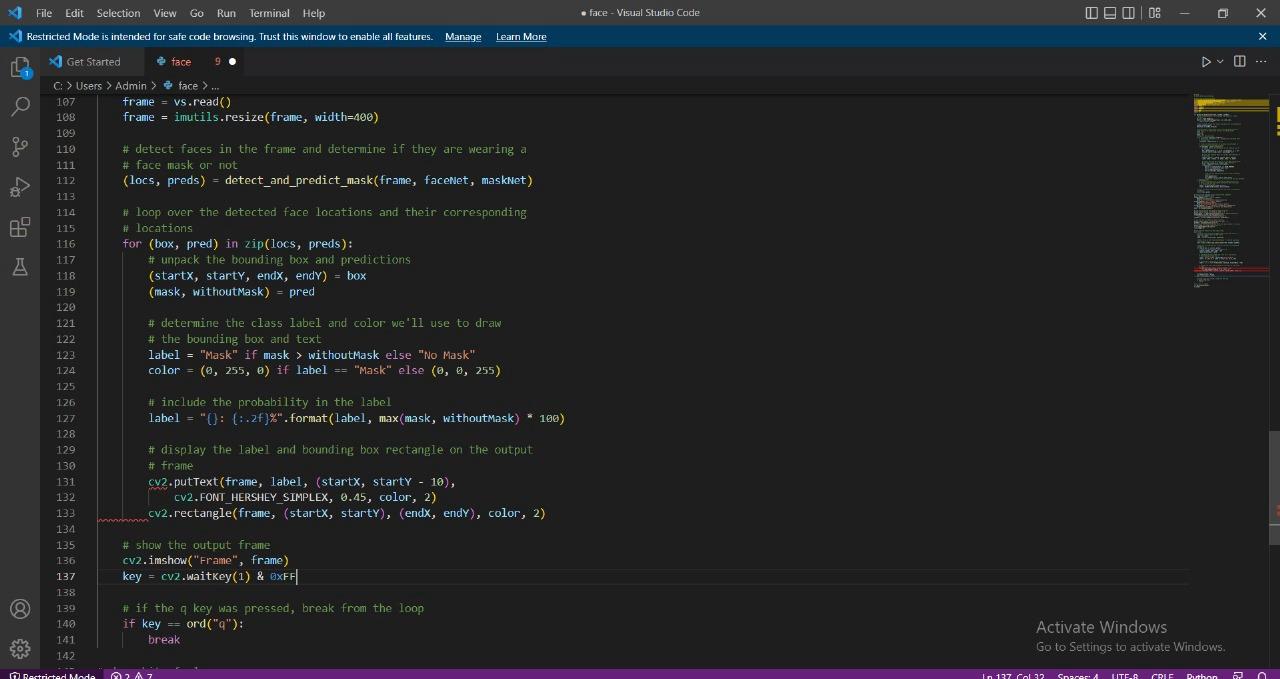


Fig 12: code screenshots

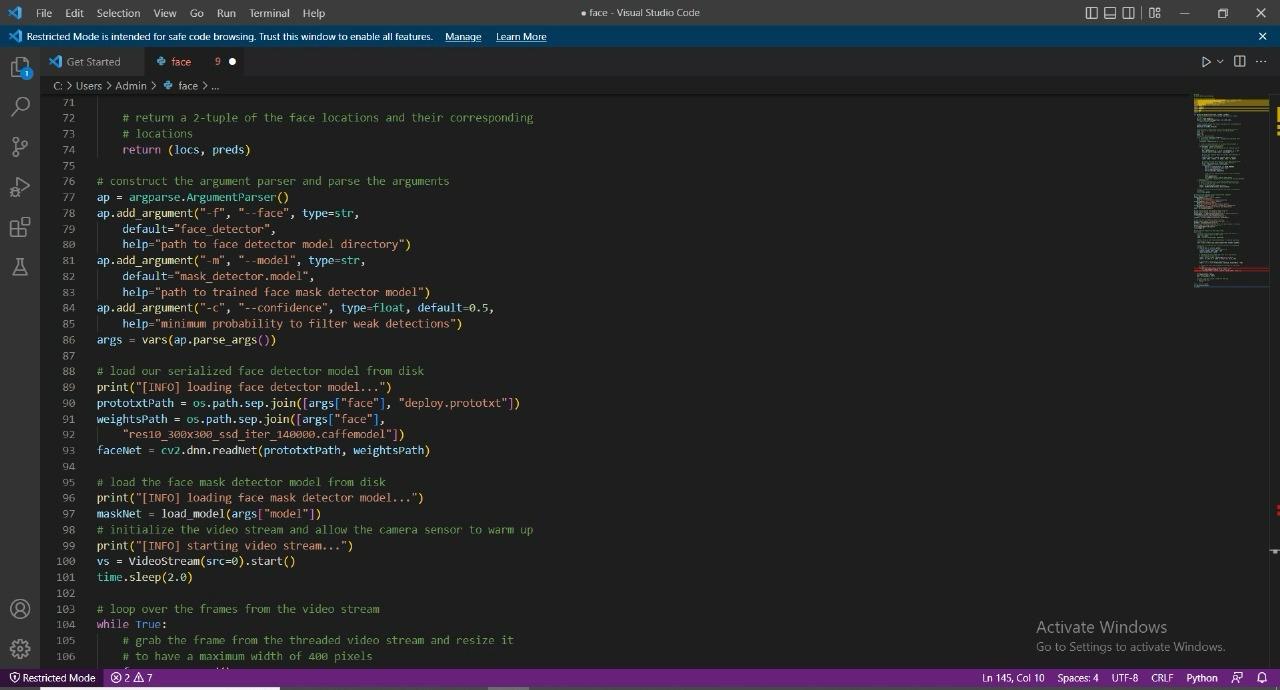
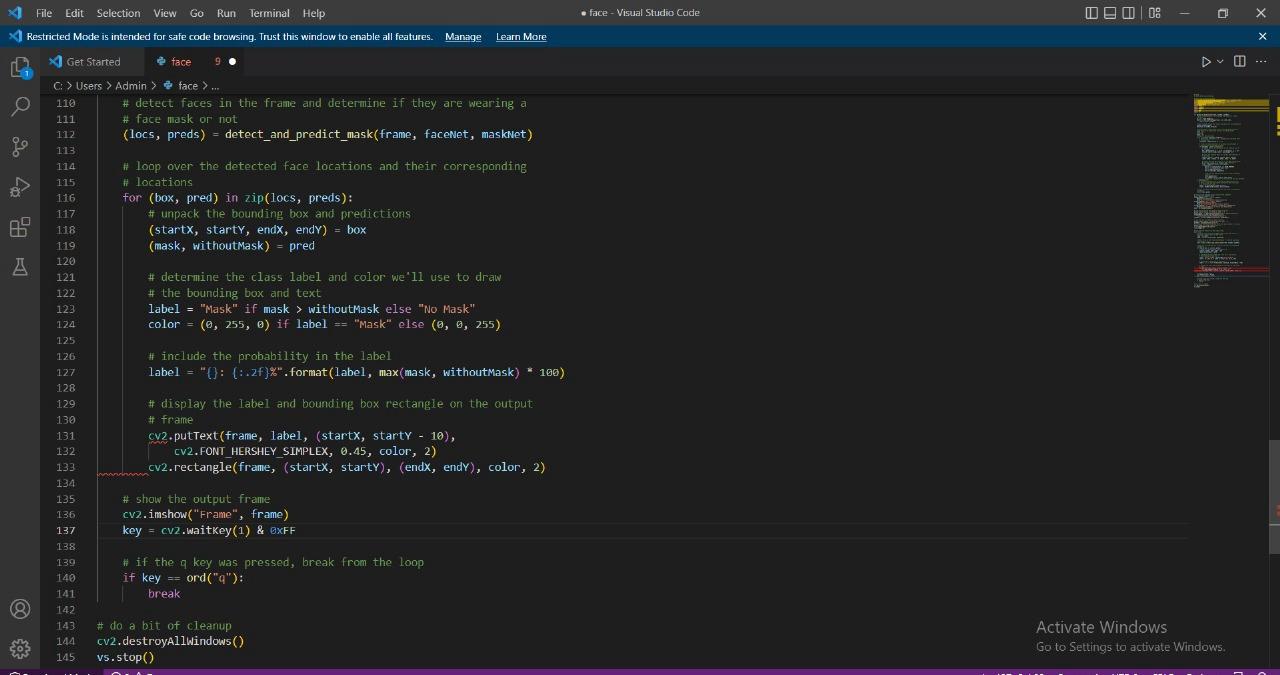


Fig 13: Code Screenshot



## Fig 14: Code Screenshot

## OUTPUT:

## 

## 



Fig 15: Output Screenshots

# CHAPTER-7

# SYSTEM TESING

## CHAPTER-7

## SYSTEM TESTING

The goal of testing is to find mistakes. The goal of testing is to find every potential flaw or vulnerability in a work product. It offers a means of testing the functionality of individual parts, subassemblies, assemblies, and/or final products. It is a procedure for testing software to make sure it satisfies user requirements and expectations and doesn't malfunction in an unacceptable way. Different test kinds exist. Every test type responds to a certain testing need**.**Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

## 

## TYPES OF TESTS

**7.1 Unit testing:**

The process of designing test cases for unit testing ensures that the core logic of the program is operating correctly and that program inputs result in legitimate outputs. Validation should be done on all internal code flows and decision branches. It is the testing of the application's separate software components. prior to integration, it is completed following the conclusion of a single unit. This is an intrusive structural test that depends on an understanding of its structure. Unit tests evaluate a particular application, system configuration, or business process at the component level. Unit tests make assurance that every distinct path in a business process has inputs and outputs that are well-defined and that it operates precisely according to the stated specifications. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**7.2: Integration testing**

Software components that have been merged are tested in integration tests to see if they genuinely operate as a single program. Testing is event-driven and focuses mostly on the fundamental results of fields or screens. Integration tests verify that even though unit testing successfully demonstrated that each component was satisfied alone, the combination of components is accurate and consistent. The purpose of integration testing is to identify any issues that may come from the combining of different components.

**7.3 Functional Test:**

Functional tests offer methodical proof that the functions being tested are available in accordance with the technical and business requirements, system documentation, and user manuals.Focus of functional testing is on the following areas:

Valid Input: Recognized classes of valid input need to be approved.

Invalid Input: Certain forms of invalid input need to be disregarded.

Functions: The functions that have been discovered need to be used.

Output: It is necessary to exercise the designated types of application outputs.

Systems/Procedures: You need to call upon the interacting systems or procedures.

Functional test preparation and organization are centered on requirements, important features, or unique test cases. Furthermore, testing needs to take into account data fields, specified procedures, sequential processes, and systematic coverage related to identifying business process flows. Additional tests are found before functional testing is finished. Data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**7.4 System Test:**

System testing verifies that all requirements are met by the integrated software system as a whole. It puts a setup to the test in order to guarantee dependable outcomes. The configuration-oriented system integration test is an illustration of a system test. System testing emphasizes pre-driven process connections and integration points and is based on process flows and descriptions. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

### 7.5WHITE BOX TESTING

White box testing is testing of a software solution's internal structure, design, and coding. In this type of testing, the code is visible to the tester. It focuses primarily on verifying the flow of inputs and outputs through the application, improving design and usability, strengthening security. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing. It is usually performed by developers. It is one of two parts of the Box Testing approach to software testing. Its counterpart, Black box testing, involves testing from an external or end-user type perspective.

#### What do you verify in White Box Testing?

White box testing involves the testing of the software code for the following:

* + - Internal security holes
    - Broken or poorly structured paths in the coding processes
    - The flow of specific inputs through the code
    - Expected output
    - The functionality of conditional loops
    - Testing of each statement, object, and function on an individual basis.

#### How do you perform White Box Testing?

To give you a simplified explanation of white box testing, we have divided it into two basic steps. This is what testers do when testing an application using the white box testing technique:

#### Step 1 :

Understand the source code the first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing.

#### Step 2 :

Create test cases and execute the second basic step to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of processes in the application.

### WHITE BOX TESTING TECNIQUES:

A major White box testing technique is Code Coverage analysis. Code Coverage analysis eliminates gaps in a Test Case suite. It identifies areas of a program that are not exercised by a set of test cases. Once gaps are identified, you create test cases to verify untested parts of the code, thereby increasing the quality of the software product There are automated tools available to perform Code coverage analysis. Below are a few coverage analysis techniques 30 Statement Coverage- This technique requires every possible statement in the code to be tested at least once during the testing process of software engineering.

### TYPES OF WHITE BOX TESTING

White box testing encompasses several testing types used to evaluate the usability of an application, block of code or specific software package. There are listed below :

**Unit Testing:**

It is often the first type of testing done on an application. Unit Testing is performed on each unit or block of code as it is developed. Unit Testing is essentially done by the programmer. As a software developer, you develop a few lines of code, a single function or an object and test it to make sure it works before continuing Unit Testing helps identify a majority of bugs, early in the software development lifecycle. Bugs identified in this stage are cheaper and easy to fix.

**Testing for Memory Leaks :** Memory leaks are leading causes of slower running applications. A QA specialist who is experienced at detecting memory leaks is essential in cases where you have a slow running software application. Apart from above, a few testing types are part of both black box and white box testing.

**White Box Penetration Testing:** In this testing, the tester/developer has full information of the application's source code, detailed network information, IP addresses involved and all server information the application runs on. The aim is to attack the code from several angles to expose security threats.

**White Box Mutation Testing :** Mutation testing is often used to discover the best coding techniques to use for expanding a software solution.

#### White Box Testing Tools Below is a list of top white box testing tools

* ParasoftJtest
* EclEmma
* NUnit
* PyUnit
* HTMLUnit
* CppUnit

### ADVANTAGES OF WHITE BOX TESTING

* Code optimization by finding hidden errors.
* White box tests cases can be easily automated.
* Testing is more thorough as all code paths are usually covered.
* Testing can start early in SDLC even if GUI is not available.

### DISADVANTAGES OF BLACK BOX TESTING

* White box testing can be quite complex and expensive.
* Developers who usually execute white box test cases detest it. The white box testing by developers is not detailed can lead to production errors.
* White box testing requires professional resources, with a detailed understanding of programming and implementation.

### 7.6 BLACK BOX TESTING

Black box testing is defined as a testing technique in which functionality of the Application Under Test (AUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on software requirements and specifications. In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

#### How to do Black Box Testing

Here are the generic steps followed to carry out any type of Black Box Testing.

* + - Initially, the requirements and specifications of the system are examined.
    - Tester determines expected outputs for all those inputs.
    - Software tester constructs test cases with the selected inputs.
    - The test cases are executed.
    - Software tester compares the actual outputs with the expected outputs.
    - Defects if any are fixed and re-tested.

### TYPES OF BLACK BOX TESTING

There are many types of Black Box Testing but the following are the prominent ones –

#### Functional testing :

This black box testing type is related to the functional requirements of a system; it is done by software testers.

#### Non-functional testing :

This type of black box testing is not related to testing of specific functionality, but non- functional requirements such as performance, scalability, usability.

#### Regression testing :

Regression Testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

#### Tools used for Black Box Testing:

Tools used for Black box testing largely depends on the type of black box testing you are doing

* + - For Functional/ Regression Tests you can use - QTP, Selenium
    - For Non-Functional Tests, you can use – Load Runner, J-meter

#### Black Box Testing Techniques:

Following are the prominent Test Strategy amongst the many used in Black box Testing.

#### Equivalence Class Testing :

It is used to minimize the number of possible test cases to an optimum level while maintains reasonable test coverage.

#### Boundary Value Testing :

Boundary value testing is focused on the values at boundaries. This technique determines whether a certain range of values are acceptable by the system or not.

#### Regression testing :

Regression Testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

**7.7 Unit Testing:**

Although it is not unusual for coding and unit testing to be carried out as two separate phases, unit testing is typically carried out as part of a combined code and unit test phase of the software lifecycle.

**Test plan and methodology**

Functional tests will be meticulously prepared, and field testing will be done by hand.

**Objectives of the test:**

• Every field entry must function correctly.

• You have to click the designated link to activate the pages.

• There shouldn't be any delays in the entry screen, messages, or answers.

**Features to be tested:**

• Ensure that all submissions follow the proper format;

• No duplicate entries should be permitted;

• Ensure that all links direct users to the appropriate page.

**7.8 Integration Testing**

The process of incrementally integrating two or more integrated software components on a single platform to identify interface flaws that lead to failures is known as software integration testing.

The purpose of an integration test is to verify that software applications or system components, or even higher up, company-level software applications, work together flawlessly.

**Test Findings:**

Every test case that was previously specified was successful. No flaws were found.

**7.9 Acceptance Testing**

Acceptance by Users Any project's testing phase is crucial, and it involves a lot of end user input. It also guarantees that the system satisfies the functional specifications.

**Test Findings:**

Every test case that was previously specified was successful. No flaws were found.

**Advantages:**

This aims to reduce the number of parameters and makes it possible to classify masked face images. This deep quantization technique presents many advantages. It ensures a lightweight representation that makes the real-world masked face recognition process a feasible task.

* Intelligent Alerts.
* Facial Recognition.
* Camera Agnostic.
* Easy Implementation.

**Disadvantages:**

* Threatens privacy.
* Imposes on personal freedom.
* Violates personal rights.
* Data vulnerabilities

**CHAPTER-8**

**CONCLUSION**

### CHAPTER-8

### CONCLUSION

### 8.1 CONCLUSION

In this paper, we briefly explained the motivation of the work at first. Then, we illustrated the learning and performance task of the model. Using basic ML tools and simplified techniques the method has achieved reasonably high accuracy. It can be used for a variety of applications. Wearing a mask may be obligatory in the near future, considering the Covid-19 crisis. Many public service providers will ask the customers to wear masks correctly to avail of their services. The deployed model will contribute immensely to the public health care system. In future it can be extended to detect if a person is wearing the mask properly or not. The model can be further improved to detect if the mask is virus prone or not i.e., the type of the mask is surgical, N95 or not.

# CHAPTER-9

# FUTURE ENHANCEMENT

# 

# CHAPTER – 9

# FUTURE ENHANCEMENT

### FUTURE ENHANCEMENT

1. **Real-time Performance Optimization:**

- Develop and optimize models for real-time face mask detection to ensure quick and efficient processing, especially in applications where timely responses are critical.

2.**Unconstrained Environments:**

- Enhance models to perform well in diverse and uncontrolled environments, accounting for variations in lighting, backgrounds, and camera angles. This will improve the robustness of face mask detection systems.

**3. Enhanced User Interaction:**

- Implement models that can interact with users, providing feedback or instructions related to mask usage. This can contribute to better user compliance and overall effectiveness.

**4.Privacy-Preserving Techniques:**

- Explore privacy-preserving machine learning techniques to address concerns related to storing or processing sensitive facial data. This may involve on-device processing or federated learning approaches.

**5.User-Specific Customization:**

- Implement models that can be customized for individual users, considering factors such as facial features, preferred mask styles, and comfort levels. This could improve both accuracy and user acceptance.

6.**Behavioral Context Integration:**

- Integrate behavioral cues into the detection process, such as assessing whether a person is actively wearing a mask or if the mask is worn correctly. This adds an extra layer of context to the detection system.

**7.Adaptation to New Variants:**

- Design models that can adapt to new variants of face masks, considering changes in materials, shapes, and overall designs. This adaptability is essential for the long-term effectiveness of face mask detection systems.

**8.Continual Learning:**

- Implement continual learning approaches, allowing the model to adapt and learn from new data over time. This is important for maintaining accuracy in dynamic environments.

**9.Edge Computing Deployment:**

- Optimize models for deployment on edge devices, reducing reliance on centralized servers. This is beneficial for applications in remote or resource-constrained environments.

**10. Human-Centric Design:**

- Consider user feedback and human-centric design principles to improve the overall user experience and acceptance of face mask detection technologies.

**11.Explainability and Transparency:**

- Enhance model explainability to provide insights into the decision-making process, fostering trust and understanding, especially in sensitive applications.

# CHAPTER-10

# BIBLIOGRAPHY

**CHAPTER-10**

**BIBLIOGRAPHY**

[1] M. Hashemi, “Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation”, Journal of Big Data, vol. 6, no. 1, 2019. Available: 10.1186/s40537-019-0263-7. 2020.

[2] S. Ghosh, N. Das and M. Nasipuri, “Reshaping inputs for convolutional neural network: Some common and uncommon methods”, Pattern Recognition, vol. 93, pp. 79-94, 2019. Available: 10.1016/j.patcog.2019.04.009.

[3] “Guide to the Sequential model - Keras Documentation”, Faroit.com, 2020. [Online]. Available: https://faroit.com/keras-docs/1.0.1/gettingstarted/sequential-model-guide/. 2020.

[4] Nwankpa, C., Ijomah, W., Gachagan, A. and Marshall, S., 2020. Activation Functions: Comparison Of Trends In Practice And Research For Deep Learning. [online] arXiv.org. Available at: [https://arxiv.org/abs/1811.03378. 2020](https://arxiv.org/abs/1811.03378.%202020).

[5] K. Team, “Keras documentation: MaxPooling2D layer”, Keras.io,2020. [Online]. Available: [https://keras.io/api/layers/pooling layers/ max pooling2d/. 2020](https://keras.io/api/layers/pooling%20layers/%20max%20pooling2d/.%202020).

[6] “Coronavirus — Human Coronavirus Types — CDC”, Cdc.gov, 2020.[Online]. Available: https://www.cdc.gov/coronavirus/types.html. 2020.

[7] W.H.O., “Advice on the use of masks in the context of COVID-19:interim guidance”, 2020.

[8] B.Suvarnamukhi and M.Seshashayee, “Big Data Concepts and Techniques in Data Processing”, International Journal of Computer Sciences and Engineering, vol. 6, no. 10, pp. 712-714, 2018. Available:10.26438/ijcse /v6i10.712714.

[9] C. Kanan and G. Cottrell, “Color-to-Grayscale: Does the Method Matter in Image Recognition?”, PLoS ONE, vol.7, no.1, p.e29740,2012.Available:10.1371/journal.pone.0029740.

[10] D. Meena and R. Sharan, “An approach to face detection and recognition,” 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE), Jaipur, 2016, pp. 1-6, doi:

82

[11] “OpenCV”, Opencv.org, 2020. [Online]. Available: <https://opencv.org/.2020>.

[12] K. Team, “Keras documentation: About Keras”, Keras.io, 2020. [Online]. Available: https://keras.io/about. 2020.

[13] Opencv-python-tutroals.readthedocs.io.2020. Colorspaces OpenCV-Python Tutorials1Documentation. Available at: https://opencv-python-tutroals.readthedocs.io/en/latest/py tutorials/py imgproc/py colorspaces/py colorspaces.html. 2020.

[14] M. Jiang, X. Fan and H. Yan, “RetinaMask: A Face Mask detector”, arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/2005.03950.2020.

[17]FaceMask Detection”, Kaggle.com, 2020. [Online]. Available: https://www.kaggle.com/andrewmvd/face-mask-detection. 2020.

[18] S. Ge, J. Li, Q. Ye and Z. Luo, “Detecting Masked Faces in the Wild with LLE-CNNs,” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 426-434, doi:10.1109/CVPR.2017.53

[19] F. Hohman, M. Kahng, R. Pienta and D. H. Chau, “Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers,” in IEEE Transactions on Visualization and Computer Graphics, vol. 25, no. 8, pp. 2674-2693, 1 Aug. 2019, doi: 10.1109/TVCG.2018.2843369.

[20] R. Yamashita, M. Nishio, R. Do and K. Togashi, “Convolutional neural networks: an overview and application in radiology”, Insights into Imaging, vol. 9, no. 4, pp. 611-629, 2018. Available: 10.1007/s13244-018-0639-9.

83