DETECTION OF MELANOMA SKIN CANCER

A Project Report Submitted in

partial fulfilment of the

Requirement for the award of

the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

BY

Vaishnavi Gogineni 19951A05M7

Mallepally Vinitha 19951A05P2

Vamshi Thanuku 19951A05N0

Department of Computer Science and Engineering

INSTITUTE OF AERONAUTICAL ENGINEERING
(Autonomous)

Dundigal, Hyderabad- 500043, Telangana

May,2023

DECLARATION

I certify that,

a) The work contained in this report is original and has been done by me under the guidance of my supervisor(s).

b) The work has not been submitted to any other Institute for any degree or diploma.

c) We have followed the guidelines provided by the Institute in preparing the report.

d) We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e) Whatever We have used materials (data, theoretical analysis, figures, and text) from other sources, We have given due credit to them by citing them in the text of the report and giving their details in the references. Further, We have taken permission from the copyright owners of the sources, wherever necessary.

Place: Hyderabad

Signature of the student

19951A05M7 [illegible]

19951A05P2 M. Vinitha

19951A05N0 [illegible]

Date: 19-05-2023

CERTIFICATE

This is to certify that the project report entitled **Detection of Melanoma Skin Cancer** submitted by **Vaishnavi Gogineni, Mallepally Vinitha, Vamshi Thanuku** the **Institute of Aeronautical Engineering**, Hyderabad in partial fulfilment of the requirements for the award of the Degree **Bachelor of Technology** in **Computer Science and Engineering** is a bonafide record of work carried out by them under my guidance and supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute for the award of any Degree.

[illegible]

Supervisor

Mrs. Anusha.R

Assistant professor

[illegible]

Head of the Department

Dr.C Madhusudhana Rao

Professor & HOD, CSE

Date: 19-05-2023

APPROVAL SHEET

This project report **Detection of Melanoma Skin Cancer** done by **Vaishnavi Gogineni, Mallepally Vinitha, Vamshi Thanuku** is approved for the award of the Degree Bachelor of Technology in **Computer Science and Engineering**.

[illegible]

Examiners

[illegible]

Supervisor(s)

Mrs. Anusha.R

Assistant Professor

[illegible]

Principal

Dr.L V Narasimha Prasad

Date: 19-05-2023

Place: Hyderabad

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be complete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success. I thank our college management and respected **Sri M. Rajashekhar Reddy, Chairman, IARE, Dundigal** for providing me the necessary infrastructure to conduct the project work.

I express my sincere thanks to **Dr. L. V. Narasimha Prasad, Professor and Principal,** who has been a major source of information for my work, **Dr. C Madhusudhana Rao, Head of the Department, Professor of CSE** for extending her support to continue this project work.

I am especially thankful to our supervisor **Mrs. Anusha.R Assistant Professor, Department of CSE,** for her internal support and professionalism which helped me in shaping the project into a successful one.

I would like to take this opportunity to express my gratitude to everyone who has directly or indirectly assisted me in bringing this effort to its current form.

ABSTRACT

Computer vision, deep learning, and machine learning are used to locate and focus data from complex images. Images can now identify illness and treat it. Deep neural networks are used in dermatology to distinguish melanoma images. Two major issues in melanoma location research are the focus of this post. Minor changes to the dataset's limits, the key variable, influence classifier exactness. This model examined Trade Learning challenges. In light of this initial review, we recommend that preparation test cycles be repeated to build trustworthy prediction models. Second, a flexible design philosophy that allows for modifications to the preliminary dataset is crucial. Based on clinical and dermoscopic images, we suggested a half-breed method based on cloud, dimness, and edge figuring to provide Melanoma Area with the board. This engineering should adjust to information volume by shortening the predictable retrain. A distributed system guarantees yield fulfillment in a significantly more acceptable time frame in studies conducted on a single PC using various conveyance methods.

Keywords: Computer vision techniques, Deep Learning, Machine Learning, Transfer Learning, Melanoma Detection

CONTENTS

LIST OF FIGURES

CHAPTER-1
INTRODUCTION

1.1 Introduction

Melanoma is caused by melanocytes, epidermal cells that make melanin. Even though only a small percentage of cutaneous disorders can be treated, this rise is the leading cause of death [1]. Rates of melanoma have increased, but they vary by age group. The under-50s' rate decreased by 1.2 percent between 2007 and 2016, while the over-50s' rate increased by 2.2 percent. The American Cancer Society anticipates 100350 new cases and 6850 male and female deaths from cancer in 2020. 1 Early melanoma diagnosis continues to be challenging [2]. His diagnosis is influenced by the doctor's expertise in distinguishing skin lesions. A biopsy is required for false diagnoses. Endurance is improved when melanoma is detected early, particularly in high-risk patients. Dermatologists use intense light amplification dermoscopy and ocular inspection to diagnose melanoma [3]. Innovation has the potential to alter our perspective on medicine and aid in the development of concrete frameworks for patient-centered decision-making [4]. The clinical examination and reaction, according to doctors, are inseparable. It is necessary for the dynamic cycle. Clinical entertainers and inventors must guarantee product quality in a collaborative setting. Dermoscopy, also known as epiluminescence microscopy, provides painless early detection of melanoma. Using portable equipment, a doctor can evaluate the size, shape, and color of pigmented skin lesions. Clinically, many melanomas are obvious. They're progressed and have melanoma's standard morphology. Blood tests can be done before or during therapy for advanced melanomas. Blood levels of LDH are checked by doctors before treatment. Diameter. The diameter typically exceeds 6 millimeters or has increased.

Evolving: Two symptoms of melanoma? The skin around the spot was covered in pigment, which spread from the border of the spot. swelling or redness outside the mole. discomfort, soreness, or itching. Scaliness, dripping, bleeding, or a mole-related lump or bump.

CHAPTER-II

LITERATURE REVIEW

In[1]Skin conditions are usually diagnosed through a physical examination, sometimes followed by further tests such as dermoscopy, biopsy, and histological examination. Skin diseases are common and can be difficult to identify through photographs due to their delicate structure[1]. However, deep convolutional neural networks (CNNs) can provide detailed analysis of skin lesions using pixels and infection markers, without requiring rough illustrations. In this study, a CNN was developed and trained using a dataset of 129, 450 clinical images of 2, 032 different skin disorders[1]. The accuracy of the CNN was compared to that of board- certified dermatologists who used biopsy-proven clinical images to differentiate between benign and malignant skin conditions. The results showed that the CNN was able to diagnose skin conditions as effectively as dermatologists. With the widespread use of smartphones, basic diagnostic care for skin conditions may become widely available to the 6.3 billion mobile phone subscribers by 2021[1].

In[2]Despite the increasing incidence of cutaneous melanoma, early detection remains a crucial public health concern. Recent cases of melanomas with a small diameter (less than or equal to 6 mm) have led to a reconsideration of the 1985 ABCD (Asymmetry, Border irregularity, Color variegation, Diameter greater than 6 mm) rule. Methodology: A search of PubMed and Cochrane Library was conducted using the terms "ABCD," "melanoma," and "small diameter melanoma" from 1980 to 2004, with additional relevant content found in the reference lists of identified publications. Results: Current research suggests that the diameter threshold in the ABCD rule cannot be decreased from the current recommendation of greater than 6 mm. Findings propose the inclusion of pigmented lesions in the ABCDE rule to emphasize the common history of melanoma[2]. Healthcare providers and patients should thoroughly evaluate the size, shape, symptoms (such as itching or discomfort), texture (especially changes in texture), and color of nevi. Conclusion: To improve early detection of cutaneous melanoma, the ABCD rule should be updated to ABCDE and include "evolving" to emphasize the atypical appearance of pigmented skin lesions[2].

In[3]The ABCD dermoscopy rule is used by healthcare providers and radiologists to differentiate between benign and malignant skin lesions. However, solely relying on a dermoscopic score for diagnosis may lead to misinterpretation of early-stage lesions. To improve the accuracy of ABCD characteristics, a dermatological expert system (DermESy) was developed to differentiate between benign, malignant, and indeterminate lesions. DermESy is a rule-based expert system that incorporates dermatological expertise and accurate dermoscopic evaluation[3]. Using the total dermoscopic score (TDS), which is comparable to that of an expert dermatologist, DermESy categorizes dermoscopic images as benign, malignant, or indeterminate lesions, taking into account shape, symmetry, and color variation. The system can also detect "C" scores and significant color variation through color data extraction. A skin lesion's "D" score is determined by dermoscopic structure segmentation. This review enhances the ABCD dermoscopy rule by considering the spatial features of dermoscopic patterns. DermESy's drawing tool helps dermatologists locate specific features. With 97.86% sensitivity, specificity, and accuracy, DermESy can differentiate between benign and malignant skin lesions. DermESy's TDS evaluation was validated and compared to expert dermatologists' TDS evaluations on identical dermoscopy images, demonstrating the system's effectiveness and reliability[3].

In[4]Recently developed CDSSs are capable of detecting melanoma, but their use is limited to specialists due to the need for specialized knowledge. Moreover, these systems are unable to detect nonmelanoma skin cancer (NMSC), which is also a common skin disease. To address these limitations, a CDSS called Nevus Doctor (ND) is being developed. The aim is to create a system that can be used by non-specialists and can detect a wide range of skin lesions[4]. A study was conducted to assess the effectiveness of ND in detecting melanoma and NMSC. The study analyzed 870 dermoscopic images of skin lesions, including 44 melanomas and 101 NMSCs, using ND and a comparable CDSS called Mole Expert (ME). ND was found to have comparable melanoma detection rates as ME. ND was able to detect NMSC with 100% sensitivity but only 12% specificity. ME was able to correctly identify some melanomas that were misclassified by ND, and vice versa. ND can detect NMSC without reducing melanoma detection rates[4].

In[5]With skin cancer rates increasing, a lack of clinical expertise, and limited resources, there is growing interest in using artificial intelligence (AI) to aid in skin disease diagnosis. AI systems utilizing improved learning algorithms have been developed to differentiate between benign and malignant skin lesions in clinical, dermoscopic, and histological images[5]. While AI systems have shown promise in accurately classifying skin lesions, they are still in the early stages of clinical use and are not yet capable of identifying skin developments that pose a threat. This review discusses the current state of AI-based image recognition systems for detecting skin cancer and suggests potential improvements to these AI models to assist dermatologists in diagnosing skin diseases[5].

CHAPTER-3

METHODOLOGY

3.1 Existing Method

The existence of noise and artifacts, low difference, and poor picture illumination should all be taken into account in ongoing melanoma detection calculations.

3.1.1 Drawbacks

1. The significant concern is how much time and additional room important to develop a be fuddled model on a lot of information to get further developed execution.
2. The effort required to update one or more models is the second drawback.

This article centers around two significant parts of melanoma acknowledgment research. Classifier accuracy is impacted by even the smallest modifications to the dataset's boundaries, which are the most critical component being investigated.

3.2 Proposed System

Two key melanoma detection research topics have been highlighted in this article. The accuracy of classifiers is the first thing that is taken into consideration, and how even a small change in the dataset's characteristics might affect that accuracy.We demonstrate that strong prediction models require ongoing training and testing cycles, and that a distributed strategy ensures that output is obtained significantly faster.

3.3 System Requirements

3.3.1 Software Requirements

Software requirements define the functionality and constraints that a software system should possess. They describe what the software should do to meet the needs of its users. Software requirements can include features, user interfaces, data processing, data storage, and more.

- Python idle 3.7 version
- Anaconda 3.7
- Jupyter
- Googlecolab

3.3.2 Hardware Requirements

Hardware requirements specify the necessary hardware components and configurations that are needed to run the software system. These requirements define the minimum or recommended hardware specifications to ensure the software operates efficiently. Hardware requirements can include processor speed, memory, disk space, network connectivity, and more.

• **Operating system**	**: windows, linux**
• **Processor**	**: minimum intel i3**
• **Ram**	**: minimum 4 gb**
• **Hard disk**	**: minimum 250gb**

3.3.3 Functional Requirements

Functional requirements describe the specific functions and capabilities that the software system should provide to its users. These requirements outline what the software should do in response to different inputs and under various conditions. Functional requirements are typically expressed as use cases or user stories and can be represented through diagrams like activity diagrams or sequence diagrams.

Examples of functional requirements could be:

Users should be able to log in to the system using their unique credentials.

The system should allow users to add products to a shopping cart.

The system should send email notifications to users when a new message is received.

3.3.4 Non- Functional Requirements

Non-functional requirements specify the qualities or attributes that the software system should possess, aside from its specific functionalities. These requirements define the overall behaviour, performance, security, usability, and other characteristics of the software. Non-functional requirements are often categorized into different aspects such as performance, reliability, security, maintainability, and usability. Examples of non-functional requirements could be:

- The system should respond to user actions within 2 seconds for 95% of requests.
- The system should maintain 99.9% uptime availability.

CHAPTER-4

SYSTEM DESIGN

4.1 Block Diagram

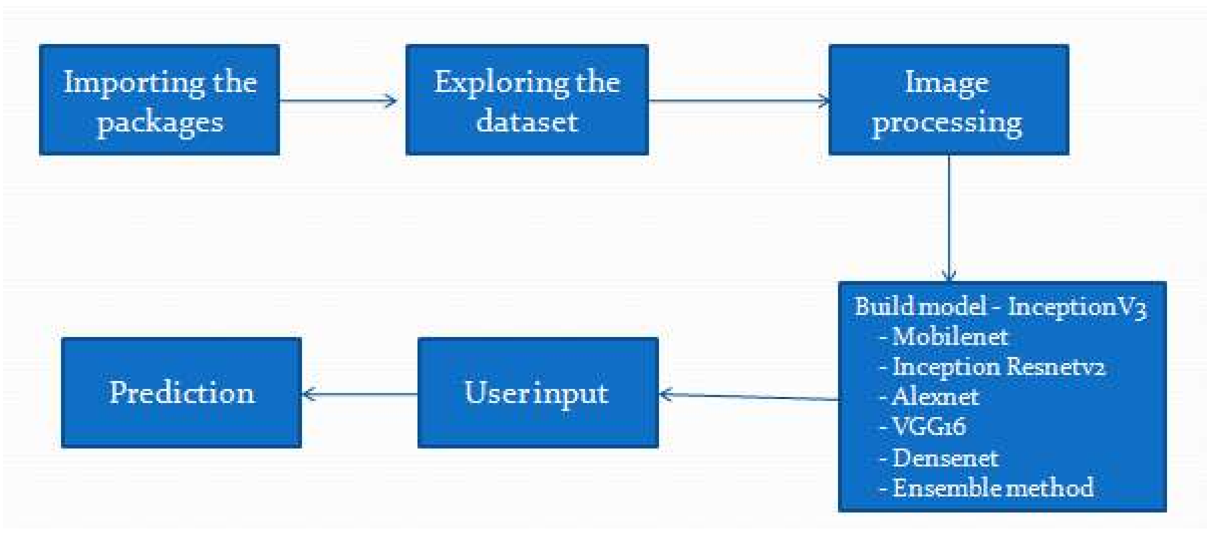


Fig. 1: Block Diagram

4.1.1 Data Flow Diagram

1. The bubble chart is another term for the DFD, which is a simple graphical representation used to illustrate a system's input data, various processing stages applied to the data, and the resulting output datagenerated by the system.
2. The DFD (data flow diagram) is an essential modeling tool is generally make use of the components of a system, including the system process, data utilized by the process, external entities that interact with the system, and the information flow within the system.
3. DFD is a visual technique that presents how information flows through a system and is transformed by a sequence of processes. It uses graphics to illustrate information flow and the alterations made to the data as it travels from input to output.
4. The bubble chart or DFD is a modeling tool that can be utilized to represent a system at any level of abstraction. DFD can be subdivided into various levels, each representing an increase in information flow and functional detail.

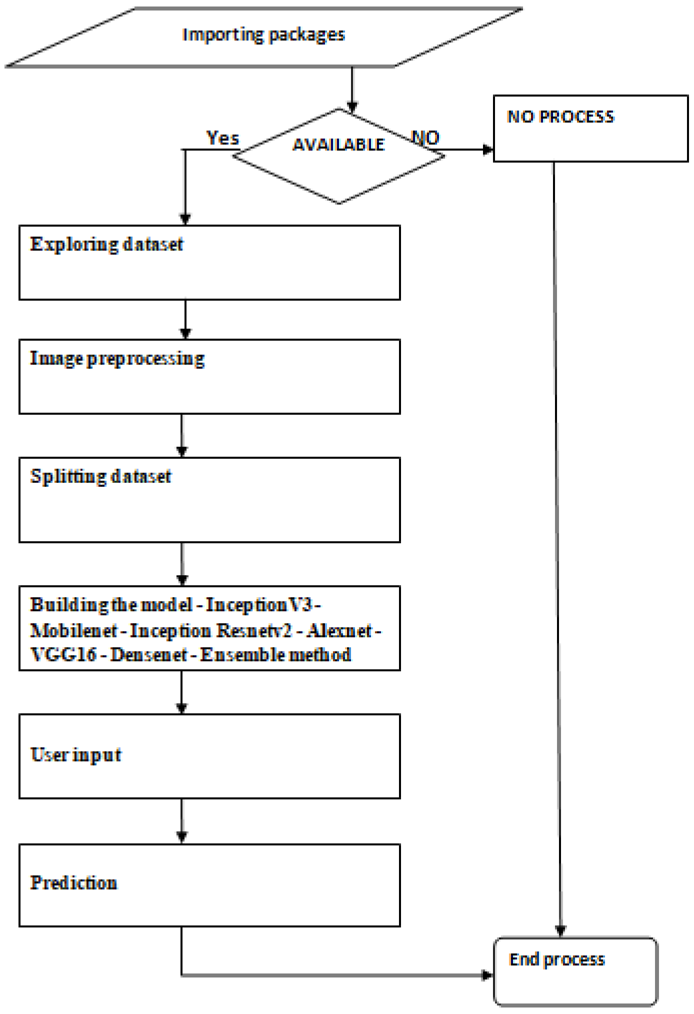


Fig. 2: Proposed Architecture

4.2 System Study

At this phase, the feasibility of the project is evaluated, and a preliminary project plan with cost estimations and a basic business proposal is presented. The system's feasibility must be thoroughly examined during the system analysis to ensure that the proposed solution does not impose any undue burden on the business. Identifying the primary system requirements is critical for the feasibility analysis.

The three main factors of feasibility analysis are

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

4.2.1. Economical Feasibility

The purpose of this study is to evaluate the financial implications of implementing the system on the company. The company has limited funds for investing in the system’s research and development, so the costs must be justified with evidence. Consequently, the created system was completed within budget, as most of the technology used was publicly available, and only the specialized equipment had to be purchased

4.2.2 Technical Feasibility

The objective of this investigation is to assess the technical feasibility of the proposed system. It is critical that the system does not overburden the existing technical resources. Due to the limitations of technical resources, it is anticipated that there will be high expectations from the customer. Therefore, thedeveloped system should have minimal technical requirements, requiring little or no adjustments during its implementation.

4.2.3 Social Feasibility

The objective of this investigation is to appraise the extent of user approval of the system, which involves appraising the education needed for the user to competently operate the system. The system should not be daunting to the user, but instead, viewed as essential. The methods used to inform and acquaint the user with the system are the sole determinant of the level of user acceptance. As the end-user of the system, the user’s trust in the system must be enhanced to stimulate positive feedback.

4.3 UML Diagrams

- Unified Modeling Language (UML) diagrams are a set of graphical notations used to represent various aspects of a system or software application.UML diagrams provide a standardized way of visualizing, specifying, constructing, and documenting the different components and interactions within a system.

- UML diagrams are widely used in software development and other fields related to system analysis and design. They help stakeholders, including developers, designers, and clients, to understand the structure, behavior, and relationships of a system. UML diagrams can be categorized into two main types: structural diagrams and behavioral diagrams.

UML diagrams serve as a powerful communication and documentation tool throughout the software development lifecycle, helping teams collaborate, identify design flaws, and ensure that the system meets the desired requirements.

4.3.1 Goals:

The goals of UML diagrams are as follows:

Visualize System Structure: UML diagrams provide a visual representation of the system’s structure, including its components, relationships, and organization. They help stakeholders understand how different parts of the system are related and interact with each other.

Capture System Behaviour: UML diagrams enable the depiction of system behavior, including how objects interact and how the system responds to events. They help in understanding the flow of activities and processes within the system.

Facilitate Communication: UML diagrams serve as a common language for communication between stakeholders, including developers, designers, clients, and domain experts. They help convey complex ideas and concepts in a visual and easily understandable format, reducing ambiguities and misunderstandings.

Aid in System Analysis and Design: UML diagrams support system analysis and design by providing a standardized notation for representing requirements, constraints, and design decisions. They help in identifying potential issues, validating design choices, and ensuring that the system meets the desired objectives.

Enable System Documentation: UML diagrams serve as documentation artifacts that capture the system’s design and architecture. They provide a reference for future development, maintenance, and enhancements. UML diagrams also aid in knowledge transfer between team members and help new members understand the system quickly.

Support System Validation and Verification: UML diagrams can be used to validate and verify system designs against requirements. They help in identifying inconsistencies, gaps, and errors in the design early in the development process, reducing the cost and effort of rework.

Aid in Code Generation and Implementation: UML diagrams can be used as a basis for generating code or implementing the system. They provide a high-level blueprint of thesystem's structure andbehavior, which can be translated into code or used as a reference during the implementation phase.

Overall, the goals of UML diagrams are to enhance communication, capture system structure and behavior, support analysis and design activities, facilitate documentation, and aid in system validation and implementation.

4.3.2 Use case diagram:

Use case diagram in the Unified Modeling Language (UML) is actually a type of structural diagram rather than a behavioural diagram. It depicts the interactions between actors and a system, showcasing the various use cases (or functionalities) provided by the system. It illustrates the functionality and behaviour of a system from the perspective of its users.

In a use case diagram, the primary elements are actors and use cases, which are represented as graphical symbols connected by lines.

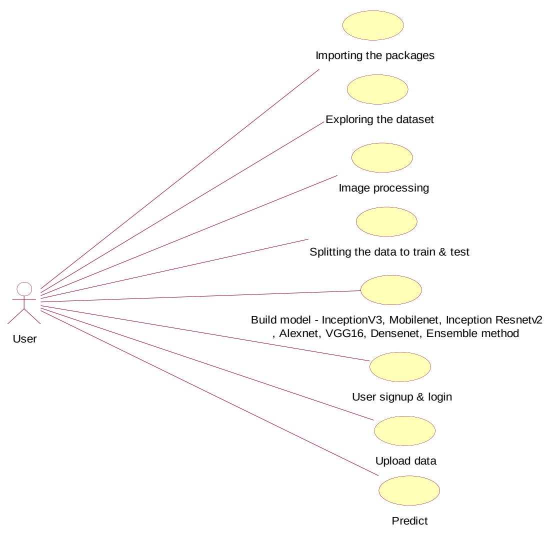

Importing the packages
Exploring the dataset
Image processing
Splitting the data to train & test
Build model - InceptionV3, Mobilenet, Inception Resnetv2 , Alexnet, VGG16, Densenet, Ensemble method
User
User signup & login
Upload data
Predict

Fig. 3:Use Case Diagram

4.3.3 Class diagram:

A class diagram is indeed a type of structural diagram in the Unified Modeling Language (UML) that represents the static structure of a system or software application. It provides a visual representation of the classes, their attributes, methods, and the relationships between them. In a class diagram, classes are the primary elements, and they are typically depicted as rectangles with three sections as you described. The top section contains the class name, the middle section includes the class's attributes (variables), and the bottom section lists the class's methods (functions or operations). Class diagrams are widely used in software development and play a crucial role in designing, modelling, and documenting the structure of a system. They offer a high-level overview of the classes in the system, their attributes, and methods, and illustrate the relationships and associations between classes.

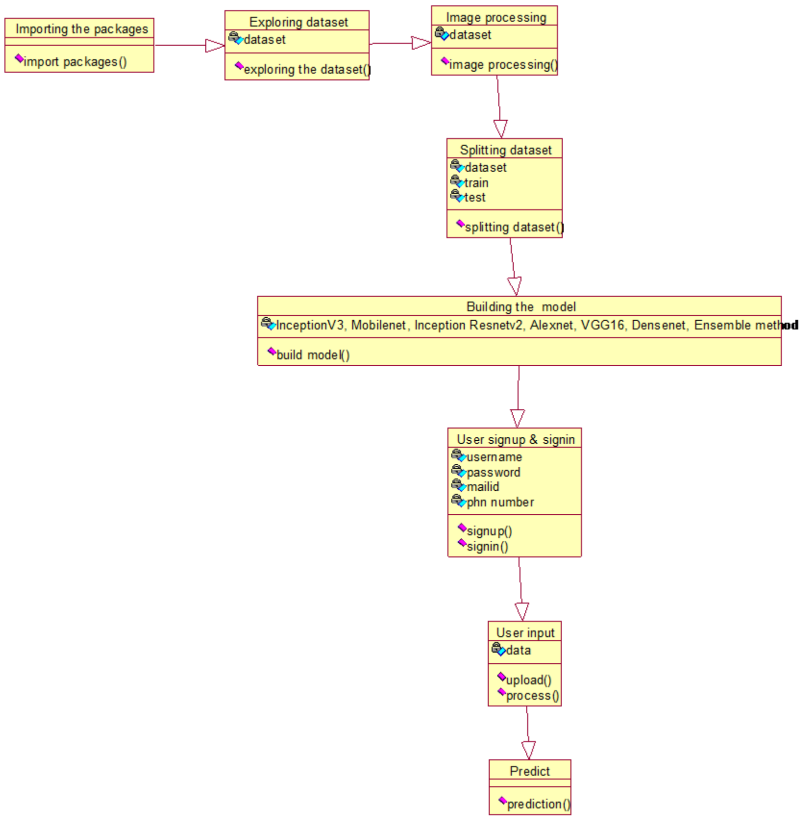


Fig. 4: Class Diagram

4.3.4 Activity diagram:

An activity diagram is a type ofbehavioral diagramin the Unified Modeling Language (UML) that represents the flow of activities or processes within a system. It focuses on the sequence of actions, decisions, and concurrency of activities. Activity diagrams are used to model the dynamic aspects of a system, emphasizing the behaviour and work flow.

Activity diagrams help in visualizing the workflow, decision points, and concurrency of activities within a system. They are particularly useful for modeling business processes, use case scenarios, and system behavior. Activity diagrams aid in understanding the overall flow of activities, identifying dependencies and conditions, and optimizing the workflow of a system. They can also be used as a basis for implementing and coordinating system functionality.

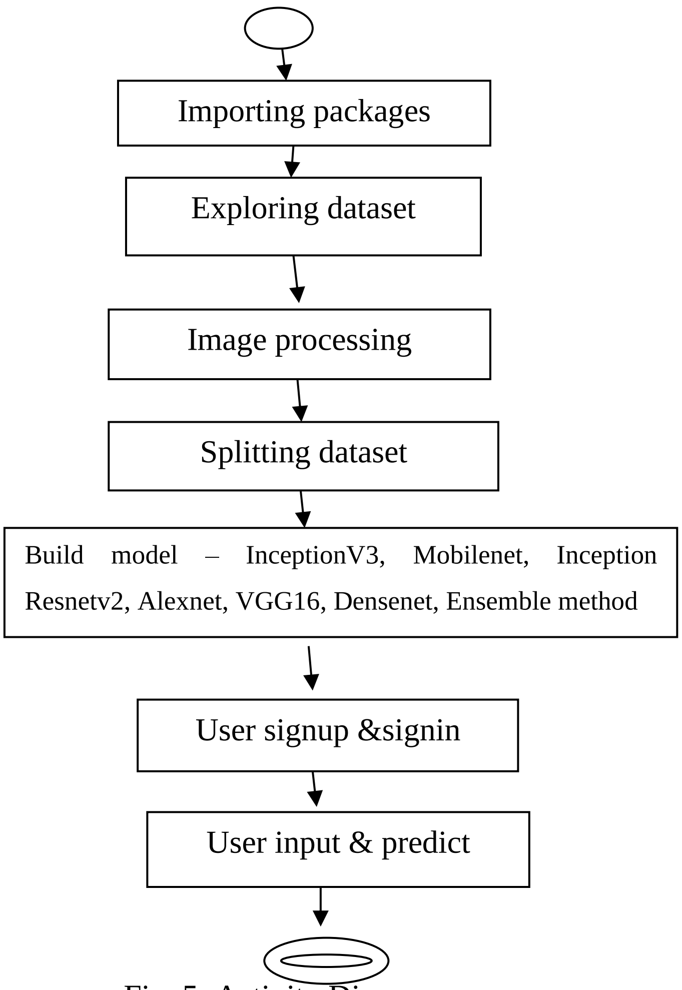


Fig. 5: Activity Diagram

4.3.5 Sequence diagram:

In a sequence diagram, the objects or actors involved in the system are represented as vertical lifelines or dashed rectangles. These lifelines depict the timeline of the objects or actors and show their existence and involvement during the sequence of interactions.It focuses on the chronological sequence of events and the flow of messages between the participating elements.

They help in understanding the order of interactions, message exchanges, and the flow of control during the execution of a system. Sequence diagrams are particularly valuable for designing and analyzing the communication between objects or actors, identifying potential issues, and ensuring that the system meets the desired behavior and requirements.

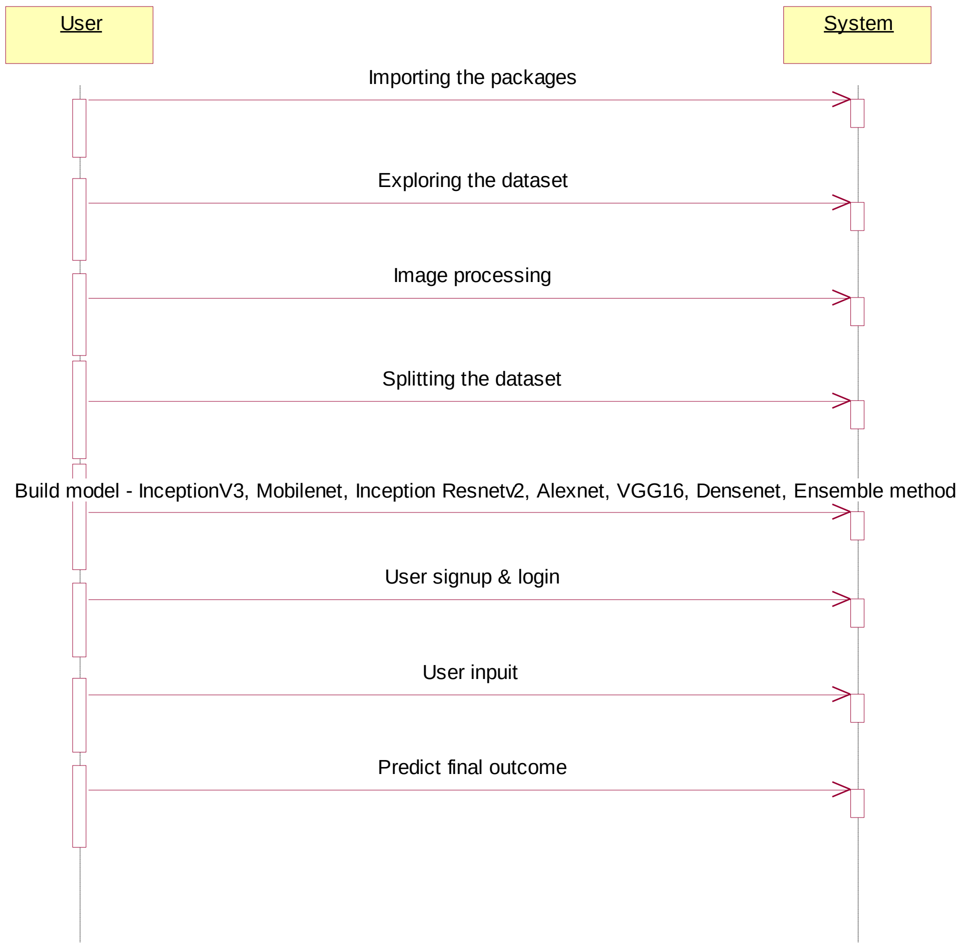


Fig. 6: Sequence Diagram

4.3.6 Collaboration diagram:

UML (Unified Modeling language) has a type of behavioural Diagram which is known as A Communication Diagram which is also known as Collaboration Diagram it states the relation between interactions and relationships among objects or actors in a system. Collaboration diagrams focus on the dynamic aspects of a system, emphasizing the exchange of messages and the structural relationships between objects. Collaboration diagrams help in visualizing the interactions and relationships between objects or actors within a system. They focus on the exchange of messages and the structural aspects of the collaboration. Collaboration diagrams are particularly useful for understanding the dynamic behaviour of a system, identifying dependencies and communication patterns, and analyzing the interactions between objects. They aid in the design, communication, and validation of system behavior.

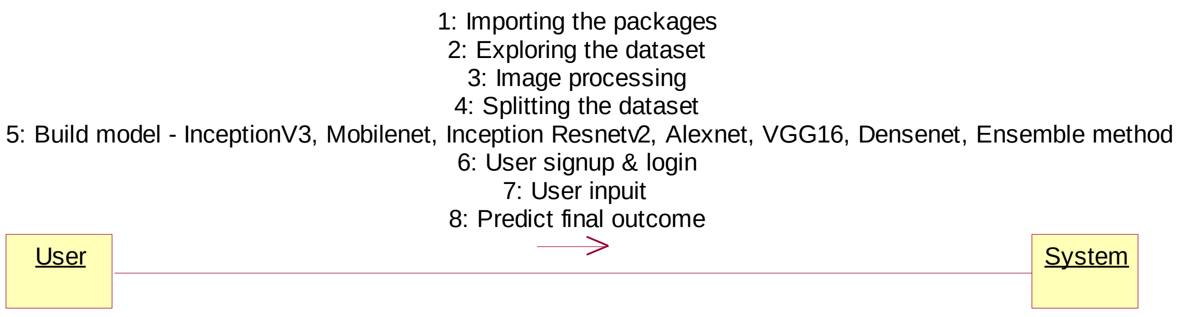


Fig. 7: Collaboration Diagram

4.3.7 Component diagram:

A component diagram is a type of structural diagram in the Unified Modeling Language (UML) that depicts the high-level organization and relationships between the components of a system. It provides a visual representation of the physical or logical components that make up a system and shows how these components interact and collaborate to fulfill the system's functionality.Component diagrams help in understanding the overall structure and organization of a system. They provide a high-level view of the components and their relationships, emphasizing the modularity, reusability, and interactions between components. Component diagrams are valuable for architectural design, system integration, and identifying the dependencies and interactions between components. They aid in the communication, analysis, and maintenance of complex systems.

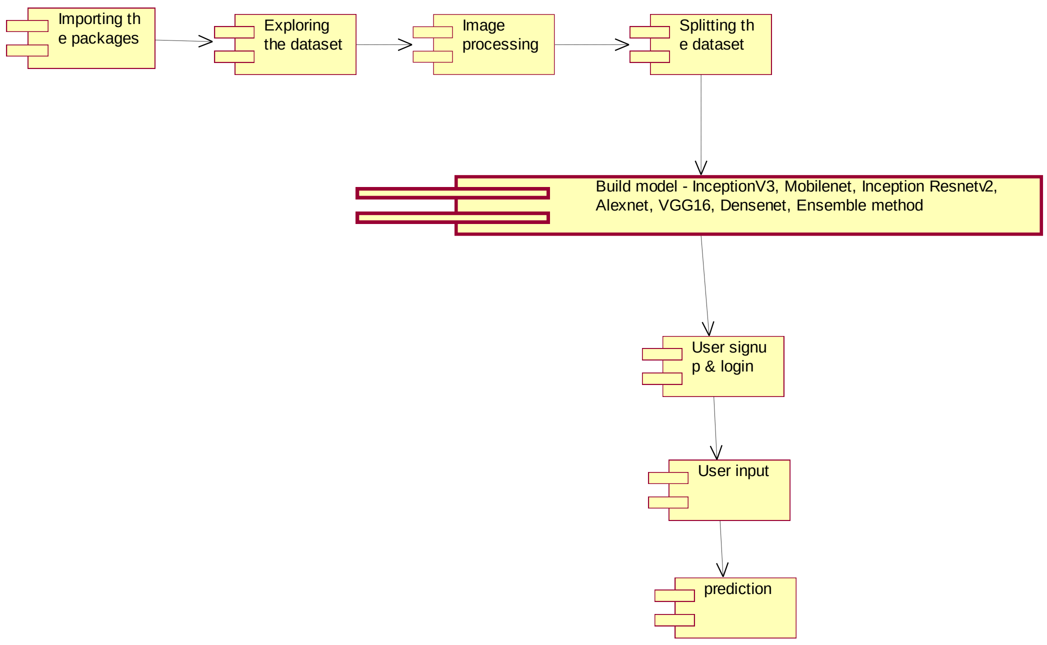


Fig. 8: Component Diagram

4.3.8 Deployment diagram:

Deployment diagram is a type of structural diagram in the Unified Modeling Languag e (UML) that depicts the physical deployment of software components onto hardware nodes in a system. It illustrates the distribution of software artifacts across different nodes or hardware environments and shows how the system's components and nodes interact with each other.Deployment diagrams help in understanding the physical distribution and configuration of a system. They provide a high-level view of how software components are deployed onto hardware nodes or computing resources.

User
System

Fig. 9: Deployment Diagram

CHAPTER-5
IMPLEMENTATION

5.1 Software Environmemt

5.1.1 Python:

Python is currently the most popular high-level, multi-purpose programming language. It supports both procedural and object-oriented programming paradigms. One of Python's notable features is that programs written in Python are typically shorter compared to other languages like Java. This is because Python programmers need to write less code, and the language's requirement for indentation ensures that the code remains readable.

Python is widely used by major tech companies such as Google, Amazon, Facebook, Instagram, Dropbox, and Uber. It is a high-level, interpreted programming language that was initially released in 1991ibyiGuidoivaniRossum. Python emphasizes code readability and utilizes whitespace extensively.

Python incorporates an automatic memory management system and a dynamic type system. It offers a comprehensive is standard library that supports various programming paradigms, including imperative, functional, procedural, and object-oriented programming. Python is an interpreted language, which means that the interpreter processes the Python code during runtime, eliminating the need for compilation before running the program. This is similar to languages like PHP and Perl. Python also provides an interactive environment where programmers can interact with the interpreter in real-time while writing programs. Python's greatest strength lies in its extensive standard library, which can be utilized for a wide range of tasks. It is commonly used in areas such as machine learning, GUI development (with libraries like PyQt, Kivy, and Tkinter), web development (with frameworks like Django), image processing (using libraries like OpenCV and Pillow), web scraping (using tools like Selenium, Beautiful Soup, and Scrapy), and testing frameworks.

5.1.2 Advantages of Python

1. Extensive Libraries: Python comes with a vast standard library that provides ready-to-use code for various purposes. This saves developers time and effort as they can leverage existing code for tasks like regular expressions, unit testing, web development, image manipulation, and more.

2. Extensibility: Python can be easily extended by integrating code from other programming languages such as C++ or C. This flexibility allows developers to optimize performance-critical sections of their code or use existing libraries written in other languages.

3. Embeddable: Python is embeddable, meaning it can be seamlessly integrated into other programming languages. Developers can add Python code to their projects in languages like C++ to provide scripting capabilities or enhance functionality.

4. Improved Productivity: Python's simplicity and readability contribute to increased developer productivity. The language's clean syntax and extensive libraries enable programmers to write code more efficiently, resulting in faster development cycles and reduced time-to-market.

5. IoT Opportunities: Python has become popular in the field of Internet of Things (IoT) due to its versatility and compatibility with platforms like Raspberry Pi. Its ease of use and support for hardware interfaces make it a suitable choice for building IoT applications

Download the Correct version into the system

Step 1: Open your web browser (e.g., Google Chrome) and go to the official Python website: https://www.python.org

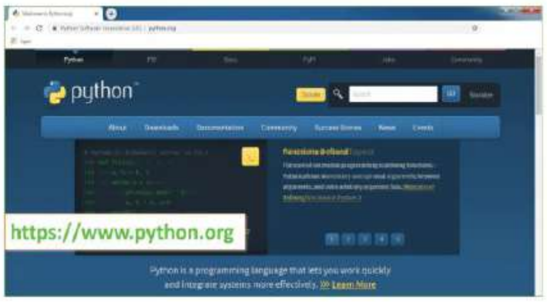
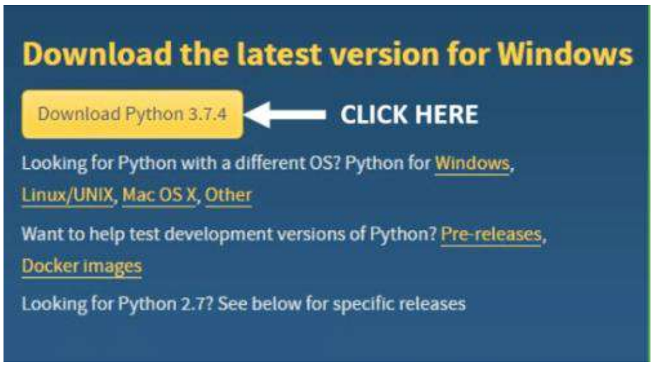

python
https://www.python.org
Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Fig 10 : Python Website Page

Step 2: On the Python website's homepage, you should see a prominent section labelled "Downloads." Click on the "Downloads" link to access the downloads page.



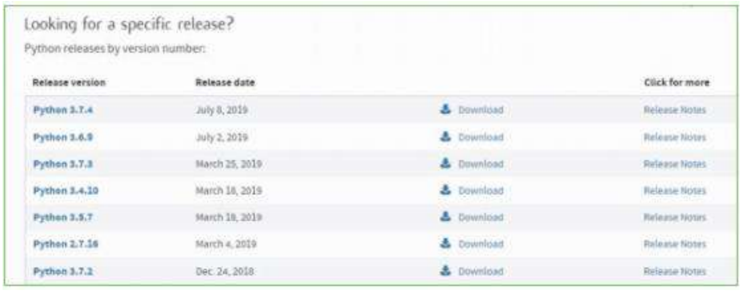
Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases, Docker images

Looking for Python 2.7? See below for specific releases

Fig. 11: Python Websites’s Homepage

Step 3: On the downloads page, you will find various versions of Python available for different operating systems. Choose the appropriate version for your operating system. For example, if you're using Windows, you might see a section for "Windows" with download links.



Looking for a specific release?

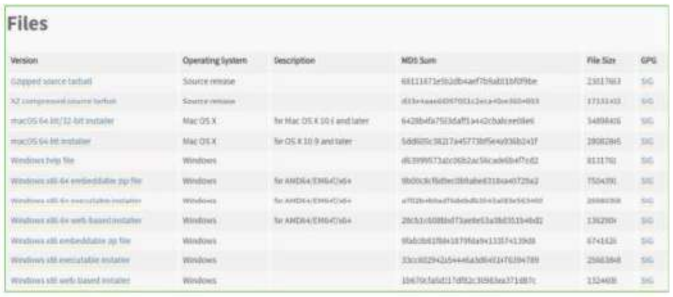
Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Fig. 12: Python Versions

Step 4: Click on the download link associated with the version of Python you want to install. This will start the download process.

Step 5: Here you see a different version of python along with the operating system.



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		[illegible]	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	[illegible]	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	[illegible]	28082845	SIG
Windows help file	Windows		[illegible]	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	[illegible]	7504391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	[illegible]	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	[illegible]	1362904	SIG
Windows x86 embeddable zip file	Windows		[illegible]	6741626	SIG
Windows x86 executable installer	Windows		[illegible]	25663848	SIG
Windows x86 web-based installer	Windows		[illegible]	1324608	SIG

Fig. 13: Files

Since you mentioned Windows 64-bit, look for the Windows x86-64 web-based installer. Click on the link corresponding to that version.

The web-based installer file (e.g., python-3.x.x-amd64-webinstall.exe) will begin downloading.

Once the download is complete, navigate to the location where the installer file wasi saved.

Double-click on the installer fileto run it. A User Account Control (UAC) dialog may appear, asking for permission to make changes to your computer. Click "Yes" to proceed.

Installation of Python

Step 1: Now open the downloaded version to continue the installation process.

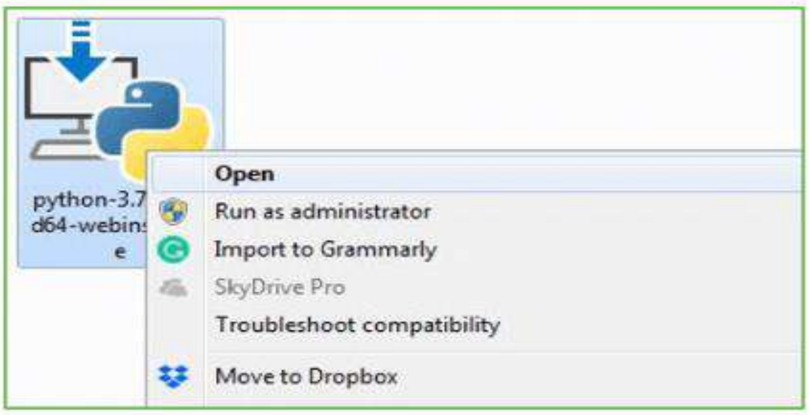
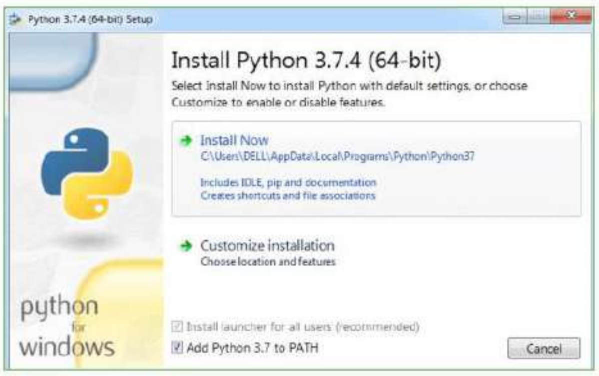


Fig. 14:Python Downloaded Version

Step 2: Click checkbox on Add Python 3.7 to PATH and later click on the install Now


Python 3.7.4 (64-bit) Setup

Install Python 3.7.4 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\DELL\AppData\Local\Programs\Python\Python37

Includes IDLE, pip and documentation
Creates shortcuts and file associations

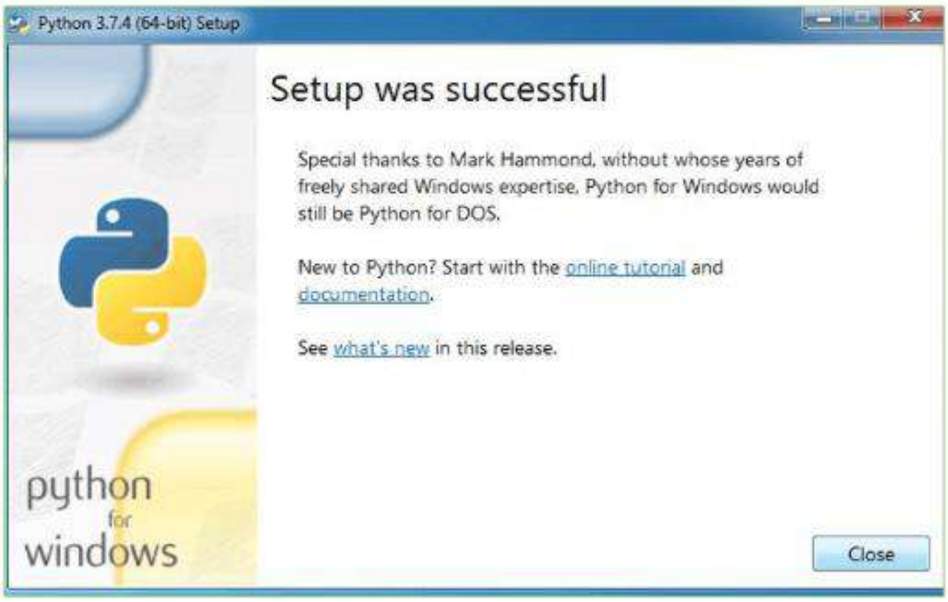
Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.7 to PATH

Cancel

Fig.15: Python 3.7

Step 3: Select the "Install Now" button to begin the installation process. You can also customize the installation by clicking on the "Customize installation" button if desired. Wait for the installation to complete. It may take a few minutes.



Python 3.7.4 (64-bit) Setup

Setup was successful

Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

New to Python? Start with the online tutorial and documentation.

See what's new in this release.

Fig. 16: Python 3.7 Setup

Step 4:Once the installation is finished, you can verify the installation

Verify the Python Installation

Step 1: Once the installation is finished, you can verify the installation by opening the command prompt. Pressithe Windows key, type "cmd," and press Enterto open the command prompt.

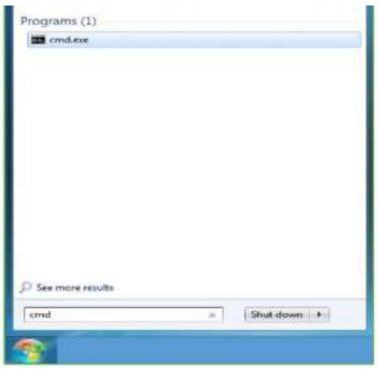
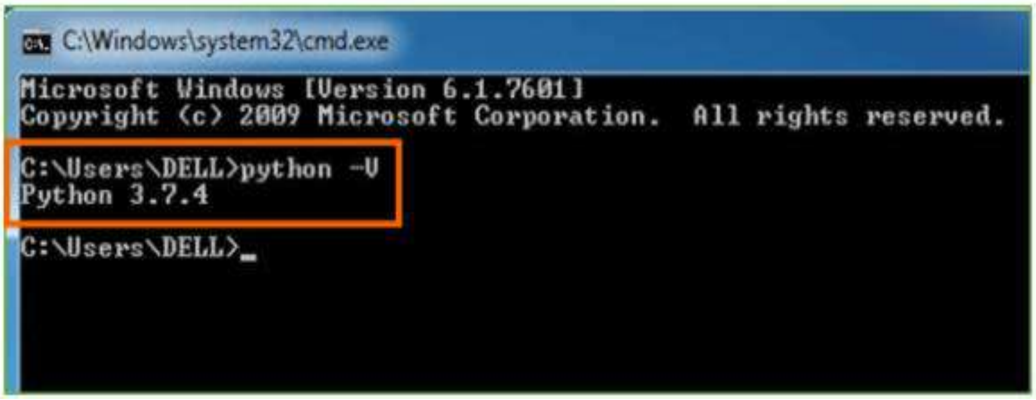


Fig. 17: Window page

Step 2: In the command prompt, type "python --v" (without quotes) and press Enter. If Python is installed correctly, it will display the version number (e.g., "Python 3.7.4).



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\DELL>python -V
Python 3.7.4

C:\Users\DELL>_
```

Fig. 18: Command prompt

Step 3: You will get the answer as 3.7.4

Note: . If Python is installed correctly, it will display the version number (e.g., "Python 3.7.4").

Check how the Python IDLE works

The Python IDLE (Integrated Development and Learning Environment) is a simple integrated development environment that come bundled with the Python programming language. It provides a basic interactive Python shell and an editor for writing and running Python code. Here's a brief overview of how the Python IDLE works:

Step 1: Once you have Python installed on your system, you can typically find the Python IDLE in the list of installed applications or by searching for "IDLE" in the Start menu (Windows) or Applications folder (macOS).

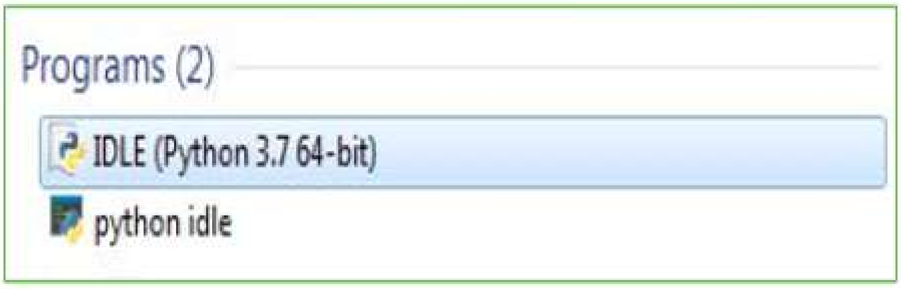


Fig. 19: Windows run command

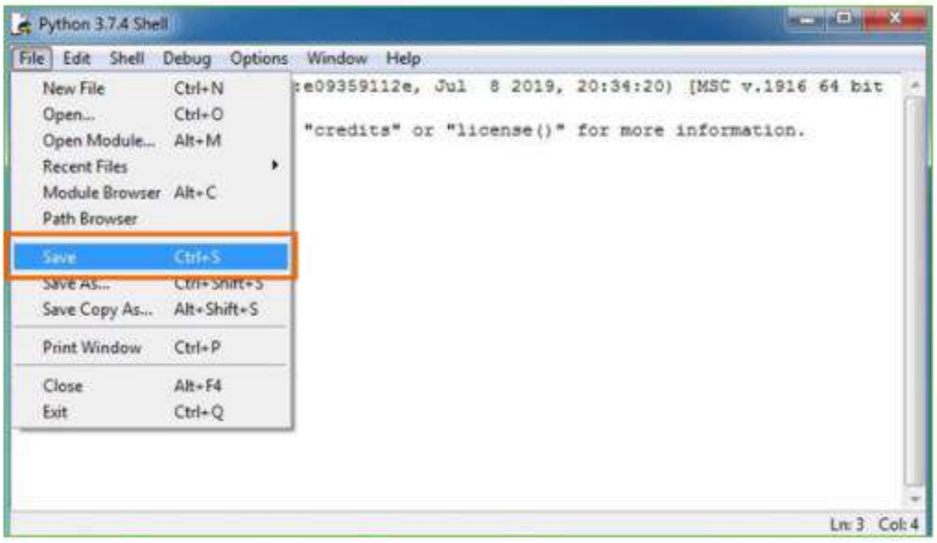
Step 2: When you open the Python IDLE, you'll see an interactive Python shell window, also known as the Python shell or REPL (Read-Eval-Print Loop). This shell allows you to enter Python commands and see the results immediately. You can type

Python statements and press Enter to execute them. The shell displays the output of the executed statements and allows you to interactively work with Python.

In addition to the interactive shell, the Python IDLE provides an editor where you can write and save Python scripts. To open a new editor window, go to the File menu and select "New File" or use the keyboard shortcut (Ctrl+N or Command+N). You can write your Python code in the editor window.

Step 3: Once you've written your Python code in the editor, you can save the script with a .py extension. To save the file, go to the File menu and choose "Save" or use the keyboard shortcut (Ctrl+S or Command+S). You can specify a file name and the location where you want to save it.

To run a Python script in the IDLE, you have a couple of options. You can either go to the Run menu and choose "Run Module" or use the keyboard shortcut (F5). Another option is to right-click in the editor window and select "Run Module" from the context menu. The Python IDLE will execute the script, and any output or error messages will be displayed in the interactive shell.



```
:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit

 "credits" or "license()" for more information.
```

Fig. 20: Python Shell

The Python IDLE also provides basic debugging features. You can set breakpoints in your code, step through the execution, inspect variables, and analyze the flow of your program.

The Python IDLE offers some additional features, such as syntax highlighting, code completion, and automatic indentation, which can enhance your coding experience.

5.1.3 Modules used in python

5.1.3.1 TensorFlow

It is a popular and widely used open-source software library for differentiable programming and data flow across various tasks. It is primarily used for machine learning and deep learning applications, particularly in the field of neural networks. TensorFlow was developed by the Google Brain team and has been extensively utilized for both research and production purposes at Google.

Originally created for internal use at Google, TensorFlow was released to the public under the Apache 2.0 open-source license on November 9, 2015. This move allowed developers and researchers worldwide to access and contribute to the library, fostering its rapid growth and adoption within the machine learning community. TensorFlow provides a comprehensive ecosystem of tools, libraries, and resources that facilitate the development and deployment of machine learning models.

5.1.3.2 Numpy

Numpy is a widely-used general-purpose array processing package in Python. It provides a powerful multidimensional array object, along with efficient operations for manipulating and interacting with these arrays. Numpy is considered a fundamental module for scientific computing in Python, offering a range of essential features and capabilities.

One of the key features of Numpy is its ability to integrate with code written in languages like C/C++ and Fortran, allowing for efficient execution of numerical computations. The N-dimensional array object provided by Numpy is highly optimized for performance, enabling fast and efficient array operations. Numpy also offers advanced functions for broadcasting, which simplifies element-wise computations between arrays of different shapes.

In addition to its scientific applications, Numpy serves as a versatile data container for handling multi-dimensional data. Its flexible data types enable seamless integration with various databases, facilitating efficient data manipulation and analysis. Numpy also includes a range of useful functionalities for linear algebra operations, Fourier transforms, and random number generation.

Overall, Numpy plays a crucial role in scientific computing and data analysis in Python, providing essential tools and capabilities for working with arrays and performing numerical computations efficiently.

5.1.3.3 Pandas

Pandas is a powerful open-source Python library that provides efficient data manpulation and analysis tools. It offers high-performance data structures that greatly enhance Python's capabilities in the domain of data preprocessing and munging. Prior to the introduction of Pandas, Python had limited impact in the field of data analysis, but Pandas addressed this gap effectively.

With Pandas, regardless of the data source, we can perform the five standard processes of data processing and analysis: data preparation, data modification, data modeling, and data analysis. Pandas simplifies and streamlines these tasks, providing a comprehensive set of functions and methods for data manipulation, cleaning, transformation, and exploration. It allows users to handle diverse data types and structures efficiently, making it an invaluable tool for both academic and professional sectors.

Python, in conjunction with Pandas, is widely used in various fields such as finance, economics, statistics, analytics, and more. Its versatility and ease of use have made it a preferred choice for data analysis tasks, enabling researchers and professionals to extract valuable insights and make informed decisions based on data-driven analysis. The combination of Python and Pandas offers a robust platform for data processing, analysis, and modeling in a wide range of domains.

5.1.3.4 Matplotlib

Matplotlib is a powerful Python 2D plotting library that enables the creation of high-quality figures in various formats for both hardcopy outputs and interactive environments across different platforms. It offers extensive functionality for generating a wide range of visualizations, such as plots, histograms, power spectra, bar charts, error charts, scatter plots, and more.

Matplotlib can be seamlessly integrated into Python scripts, as well as used in interactive environments like Python and IPython shells, Jupyter Notebooks, web application servers, and GUI toolkits. It aims to provide an intuitive and easy-to-use interface for basic plotting tasks while also offering flexibility and advanced customization options for more complex requirements.

With Matplotlib, you can quickly generate visualizations by writing just a few lines of code. The pyplot module, which provides a MATLAB-like interface, is particularly useful for simple plotting tasks, especially when combined with IPython. However, for more advanced users, Matplotlib offers full control over line styles, font properties, axes properties, and more through its object-oriented interface. This allows users to fine-tune every aspect of their plots and tailor them to their specific needs. Matplotlib's syntax and function names are also familiar to users familiar with MATLAB, making the transition easier for those coming from MATLAB.

To get started with Matplotlib, you can explore the extensive collection of sample plots and the thumbnail gallery, which provide examples and inspiration for creating various types of visualizations. Whether you are a beginner or a power user, Matplotlib provides a comprehensive plotting solution that caters to a wide range of plotting requirements and allows you to create visually appealing and informative figures.

5.2 Machine Learning

Machine learning is a field of study and practice that involves the development of algorithms and models that enable computer systems to learn and make predictions or decisions based on data without being explicitly programmed. It is a subset of artificial intelligence (AI) that focuses on the development of algorithms and techniques that allow computers to learn and improve from experience.

In the context of data science, machine learning is primarily concerned with the extraction of patterns and insights from large and complex datasets. It aims to develop models that can automatically learn and adapt from data, enabling computers to make predictions, classifications, or decisions based on patterns and trends found in the data.

Machine learning differs from traditional programming approaches in that instead of being explicitly programmed with a set of rules and instructions, the system learns and improves from data through the iterative process of training. During training, a machine learning model is exposed to a labeled dataset, where it learns the underlying patterns and relationships betweeninput data and corresponding output labels.Once the model is trained, it can be used to make predictions or decisions on new, unseen data.

It's important to note that machine learning is not a magical solution that can solve all problems. It is not about creating human-like intelligence or consciousness. Instead, it is a powerful tool that can automate and improve decision-making processes based on patterns and insights extracted from data.

Machine learning techniques can be broadly categorized into three main types: supervised learning, unsupervised learning, and reinforcement learning. Each type has its own characteristics and applications, and data scientists choose the most appropriate technique based on the nature of the problem and the available data.

In summary, machine learning is a field that focuses on the development of algorithms and models that enable computers to learn from data and make predictions or decisions. It is a powerful tool in the field of data science, allowing for the extraction of patterns and insights from large datasets and automating decision-making processes.

5.2.1 Categories Of Machine Learning :-

At a high level, machine learning can be broadly categorized into supervised learning and unsupervised learning.

Supervised learning involves training a model using labeled data, where the input data is accompanied by the corresponding output labels. The goal is to learn a mapping function that can predict the labels of new, unseen data based on their features. Supervised learning tasks can be further divided into classification and regression tasks.

In classification tasks, the goal is to assign input data points to discrete categories or classes. For example, given a dataset of emails, a classification model can be trained to classify emails as either spam or not spam. The model learns from labeled examples and generalizes the learned patterns to classify new, unseen emails correctly.

In regression tasks, the goal is to predict a continuous numerical value or quantity based on the input features. For example, a regression model can be trained on historical house prices data to predict the price of a house based on its features like area, number of rooms, etc. The model learns the patterns and relationships between the features and the target variable to make accurate predictions.

On the other hand, unsupervised learning involves training a model on unlabeled data, where the goal is to discover patterns, structures, or relationships within the data. Unsupervised learning algorithms do not have access to explicit labels or categories. Instead, they aim to find hidden patterns or group similar data points together.

Clustering is a common unsupervised learning task, where the algorithm identifies clusters or groups in the data based on their similarity. It groups data points that are more similar to each other compared to data points in other clusters. Dimensionality reduction is another unsupervised learning task that aims to reduce the number of input features while preserving the important information. It helps in visualizing high-dimensional data or removing redundant features.

These two types of machine learning, supervised and unsupervised, provide different approaches to learning from data. Depending on the problem at hand and the availability of labeled data, data scientists can choose the appropriate type of learning to build models and extract insights from the data.

5.3 Building the models

1.InceptionV3: Inception v3 is a CNN architecture developed by Google. It is designed for image analysis and object detection tasks. It aims to balance network depth and computational efficiency by utilizing various inception modules, which allow for efficient feature extraction with fewer parameters.

2. MobileNet: MobileNet is a lightweight CNN architecture specifically designed for mobile and embedded devices. It employs depthwise separable convolutions to reduce the number of parameters and computations required while maintaining good accuracy. MobileNet models are efficient and well-suited for applications with limited computational resources.

3. Inception Resnetv2: Inception-ResNet-v2 combines the concepts of the Inception architecture with residual connections from the ResNet architecture. Residual connections help alleviate the vanishing gradient problem and enable the network to be deeper. Inception-ResNet-v2 achieves a good balance between depth and computational efficiency while maintaining high accuracy.

4. AlexNet: AlexNet is a pioneering CNN architecture that gained significant attention when it won the ImageNet Large ScaleVisual Recognition Challenge (ILSVRC) in 2012. It played a crucial role in popularizing deep learning for computer vision tasks. AlexNet consists of multiple convolutional and fully connected layers and introduced concepts such as ReLU activation and dropout regularization.

5. VGG16: VGG16 is a CNN architecture developed by the Visual Geometry Group (VGG) at the University of Oxford. It was a runner-up in the ILSVRC 2014 competition. VGG16 is known for its simplicity and uniformity. It has 16 layers, including multiple convolutional layers with small filter sizes and max-pooling layers.

6. Densenet: DenseNet is a CNN architecture that introduces dense connections between layers. In a DenseNet, each layer receives direct inputs from all preceding layers and passes its outputs to all subsequent layers. Dense connections promote feature reuse, encourage gradient flow, and alleviate the vanishing gradient problem. DenseNets are known for their compactness and strong performance.

7. Ensemble method: The ensemble method is not a specific architecture but rather a technique that combines predictions from multiple models to improve overall performance. Ensemble methods leverage the diversity of different models to achieve better accuracy, robustness, and generalization. Techniques such as averaging predictions, bagging, boosting, and stacking can be employed to create ensembles.

These architectures and techniques have been influential in the field of machine learning and have demonstrated success in various computer vision tasks. Researchers and practitioners often leverage these models and methods as starting points or benchmarks when working on image analysis, object detection, and other related tasks.

5.4 Algorithm for Implementation

The algorithm you provided outlines the steps for implementing a melanoma classification system using various CNN architectures and ensemble methods. Here's a breakdown of the steps:

1**. Importing the packages**: This step involves importing the necessary libraries and packages, such as TensorFlow, Keras, and Flask, that are required for the implementation.

2. **Exploring the dataset**: You would explore the Melanoma classification database to understand the data and its characteristics. This step may involve loading the dataset, visualizing the images, and analyzing the distribution of classes.

3. **Image processing**: Image data augmentation and segmentation techniques using Keras would be applied to pre-process the images. Data augmentation techniques can include rotations, flips, and zooms, which help increase the diversity of the training data. Segmentation techniques can assist in identifying regions of interest in the images.

4. **Splitting the data**: The dataset would be split into training and testing sets. The training set is used to train the models, while the testing set is used to evaluate the performance of the trained models.

5. **Building the models**: Multiple CNN architectures, including InceptionV3, MobileNet, Inception Resnetv2, AlexNet, VGG16, and DenseNet, as well as ensemble

methods, would be implemented. Each model or ensemble would be constructed using the appropriate architecture and configurations.

6. **Training the models**: The models are trained using the training data. This involves feeding the preprocessed images to the models and adjusting the model's parameters tominimize the loss and optimize the performance.

7. **Building the Flask application**: The Flask framework is used to create a cloud platform for the application. Additionally, SQLite is employed for implementing the signup and signin functionality.

8. **User input**: Users would provide input, such as an image, through the Flask interface.

9**. Preprocessing** the input: The given input is preprocessed using the same techniques applied during the image processing step. This ensures that the input is in a suitable format for prediction.

10. **Prediction**: The preprocessed input is fed into the trained models to make predictions. Each model or ensemble would provide a prediction based on the input.

11. **Displaying the outcome**: The final outcome, which could be the predicted class or probability scores, is displayed through the cloud platform created with Flask.

It's important to note that the provided algorithm provides a high-level overview of the implementation steps. The specific details and code for each step may vary depending on the chosen libraries, frameworks, and programming language used for the implementation.

5.5 System Testing

5.5.1 Unit testing: Unit testing is indeed a testing technique that focuses on testing individual modules or units of software to ensure their functional correctness. It involves testing each unit in isolation to identify and fix defects early in the development process. Unit testing is typically performed by the developers themselves.

There are various techniques that can be used for unit testing, and two commonly used ones are:

1. **Black Box Testing**: In this technique, the internal workings of the module being tested are not known to the tester. The focus is on testing the module based on its expected input and output. The tester treats the module as a black box and tests its functionality without considering its internal implementation details. The goal is to

ensure that the module behaves correctly from the perspective of the user or external interface.

2. **White Box Testing**: This technique is also known as structural testing or glass box testing. In white box testing, the internal structure and implementation details of the module are known to the tester. The tests are designed based on the understanding ofthe internal logic, code paths, and branches of the module. The aim is to test each function and control flow within the module to ensure that all possible scenarios are covered.

Both black box testing and white box testing have their own advantages and are often used together to achieve comprehensive unit test coverage. Black box testing focuses on the external behavior and functionality of the module, while white box testing dives into the internal implementation to ensure thorough testing.

Unit testing plays a crucial role in software development as it helps in identifying defects early, isolating issues to specific units, and providing confidence in the correctness of individual modules. It is an essential part of the overall testing strategy and contributes to the overall quality of the software.

5.5.2 Data flow testing:

Data flow testing is indeed a testing strategy that focuses on the flow of data within a program. It aims to select paths through the program's control flow in order to explore sequences of events related to the status of variables or data objects. The goal is to identify defects or anomalies in the flow of data, such as incorrect assignments, uninitialized variables, or unused variables.

Data flow testing places emphasis on two key aspects: variable definition and variable usage. It aims to ensure that variables are defined and initialized correctly before they are used, and that their values are correctly propagated and utilized throughout the program. By analyzing the flow of data, data flow testing helps identify potential issues that may arise from incorrect data handling or processing.

There are different techniques and criteria used in data flow testing, such as:

1. Def-Use Coverage: This criterion focuses on ensuring that every definition of a variable is used at least once before it is redefined or goes out of scope. It helps detect potential problems caused by unused or underutilized variables.

2. Use-Def Coverage: This criterion aims to verify that every use of a variable is preceded by a proper definition or assignment. It helps identify issues related to uninitialized or undefined variables.

3. All-Defs Coverage: This criterion ensures that all possible definitions of a variable are covered during testing. It helps in testing different branches or conditions that may lead to different assignments or values for a variable.

4. All-Uses Coverage: This criterion ensures that all uses of a variable are covered during testing. It helps in detecting situations where certain uses of a variable may be missed or overlooked.

By applying data flow testing techniques and criteria, testers can systematically explore the flow of data within a program, identify potential data-related issues, and increase the overall quality and reliability of the software.

5.5.3 Integrated Testing:

Integration testing is a crucial phase in the software testing process that occurs after unit testing. It focuses on testing the interactions and interfaces between individual modules or components of a system when they are integrated together. The purpose of integration testing is to ensure that the integrated components function correctly, perform well, and exhibit the expected reliability.

The main objectives of integration testing include:

1. Functional Verification: Integration testing aims to verify that the integrated modules work together as intended and produce the expected functional outcomes. It ensures that the interactions between different modules do not introduce any defects or unexpected behaviour

2. Interface Validation: Integration testing validates the interfaces between modules, ensuring that data and control flow between them are properly handled. It checks if the data exchanged between modules is correctly interpreted, processed, and passed on.

3. Performance Evaluation: Integration testing assesses the performance of the integrated system by examining how well the modules work together under expected workloads and usage scenarios. It helps identify any performance bottlenecks, resource conflicts, or inefficiencies that may arise due to integration.

4. Reliability Testing: Integration testing also focuses on evaluating the reliability and stability of the integrated system. It checks for any issues related to error handling, exception handling, fault tolerance, and recovery mechanisms when modules interact with each other.

There are different approaches to integration testing, including:

1. Big Bang Integration: In this approach, all the modules are integrated together at once, and the entire system is tested as a whole. This approach is typically used when the modules are relatively independent and can be easily integrated.

2. Top-Down Integration: In this approach, testing starts from the top-level modules or high-level components, and gradually lower-level modules are integrated and tested. Stubs or dummy modules may be used to simulate the behavior of lower-level modules during testing.

3. Bottom-Up Integration: In this approach, testing begins with the lower-level modules, and higher-level modules are gradually integrated and tested. Drivers may be used to simulate the behavior of higher-level modules during testing.

4. Hybrid Integration: This approach combines elements of both top-down and bottom-up integration, depending on the system's architecture and module dependencies. It aims to strike a balance between comprehensive testing and efficient use of resources.

Integration testing plays a crucial role in ensuring that the integrated system functions correctly and meets the desired functional, performance, and reliability requirements. It helps identify issues that may arise due to the interaction between modules and enables their resolution before the final release of the software.

5.5.4 User interface technique:

The use of a technique testing is to identify the defects that are present in Software under GUI[Graphical User Interface].

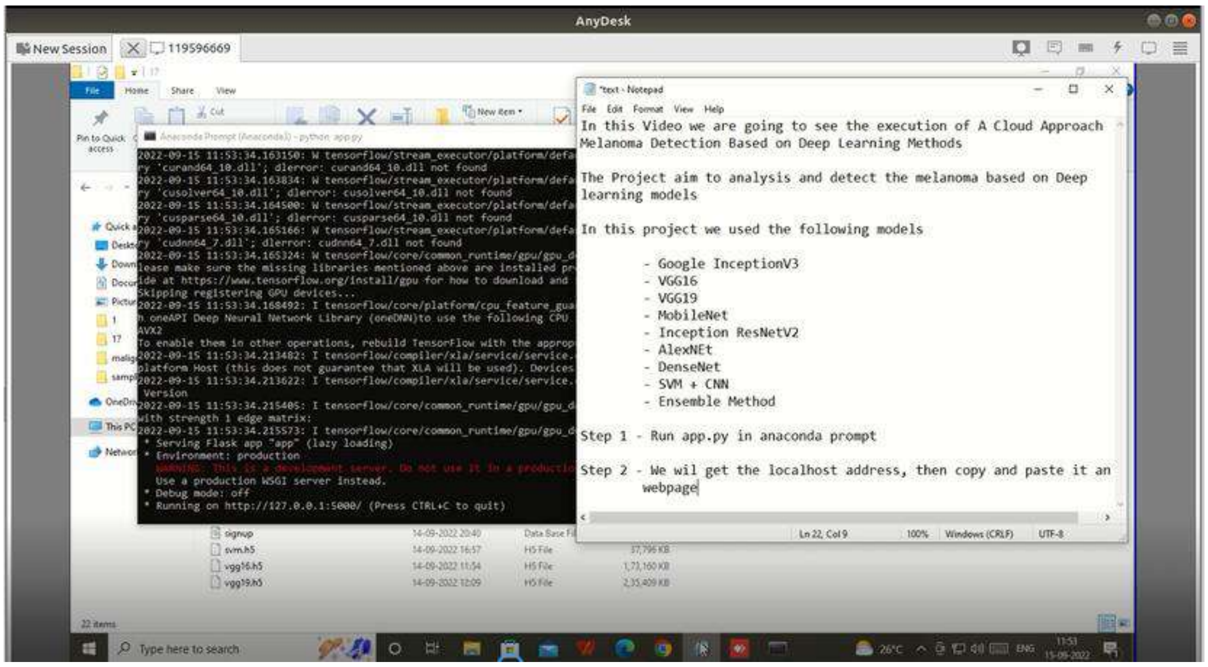
Testcases:

S.NO	**INPUT**	**If available**	**If not available**
1	User register	User get registration	There is no process
2	User login	User get login into the application	There is no process
3	User input & predict	Upload image & get predicted result for loaded image	There is no process

Table 5.1: Test Cases

CHAPTER-6

RESULT AND DISCUSSION



```
2022-09-15 11:53:34.163150: W tensorflow/stream_executor/platform/defa
ry 'curand64_10.dll'; dlerror: curand64_10.dll not found
2022-09-15 11:53:34.163834: W tensorflow/stream_executor/platform/defa
ry 'cusolver64_10.dll'; dlerror: cusolver64_10.dll not found
2022-09-15 11:53:34.164500: W tensorflow/stream_executor/platform/defa
ry 'cusparse64_10.dll'; dlerror: cusparse64_10.dll not found
2022-09-15 11:53:34.165166: W tensorflow/stream_executor/platform/defa
ry 'cudnn64_7.dll'; dlerror: cudnn64_7.dll not found
2022-09-15 11:53:34.165324: W tensorflow/core/common_runtime/gpu/gpu_d
lease make sure the missing libraries mentioned above are installed pr
ide at https://www.tensorflow.org/install/gpu for how to download and
Skipping registering GPU devices...
2022-09-15 11:53:34.168492: I tensorflow/core/platform/cpu_feature_gua
h oneAPI Deep Neural Network Library (oneDNN)to use the following CPU
AVX2
To enable them in other operations, rebuild TensorFlow with the approp
2022-09-15 11:53:34.213482: I tensorflow/compiler/xla/service/service.
platform Host (this does not guarantee that XLA will be used). Devices
2022-09-15 11:53:34.213622: I tensorflow/compiler/xla/service/service.
 Version
2022-09-15 11:53:34.215405: I tensorflow/core/common_runtime/gpu/gpu_d
with strength 1 edge matrix:
2022-09-15 11:53:34.215573: I tensorflow/core/common_runtime/gpu/gpu_d
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a productio
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

```
In this Video we are going to see the execution of A Cloud Approach
Melanoma Detection Based on Deep Learning Methods

The Project aim to analysis and detect the melanoma based on Deep
learning models

In this project we used the following models

        - Google InceptionV3
        - VGG16
        - VGG19
        - MobileNet
        - Inception ResNetV2
        - AlexNEt
        - DenseNet
        - SVM + CNN
        - Ensemble Method

Step 1 - Run app.py in anaconda prompt

Step 2 - We wil get the localhost address, then copy and paste it an
        webpage
```

Fig. 21: Anaconda Prompt

Run app.py in anaconda prompt then we will get the localhost address, then we have to copy and paste it in the

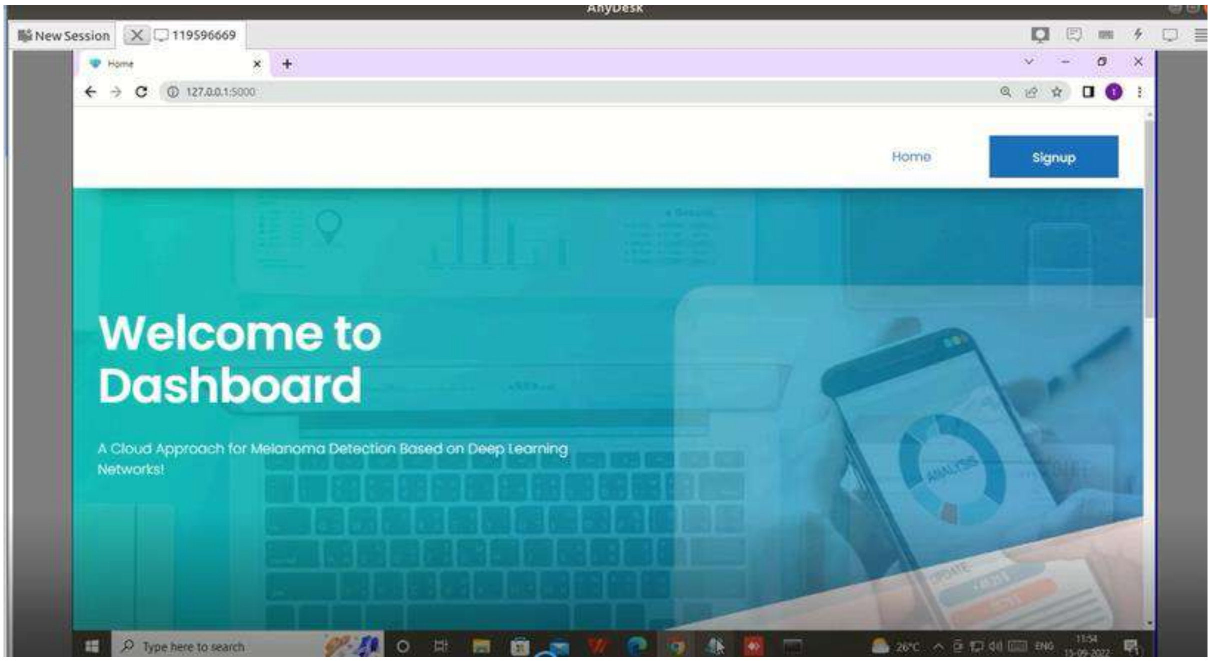

127.0.0.1:5000
Home
Signup
Welcome to Dashboard
A Cloud Approach for Melanoma Detection Based on Deep Learning Networks!

Fig. 22:Home Page

Webpage click on signup button and register for dashboard

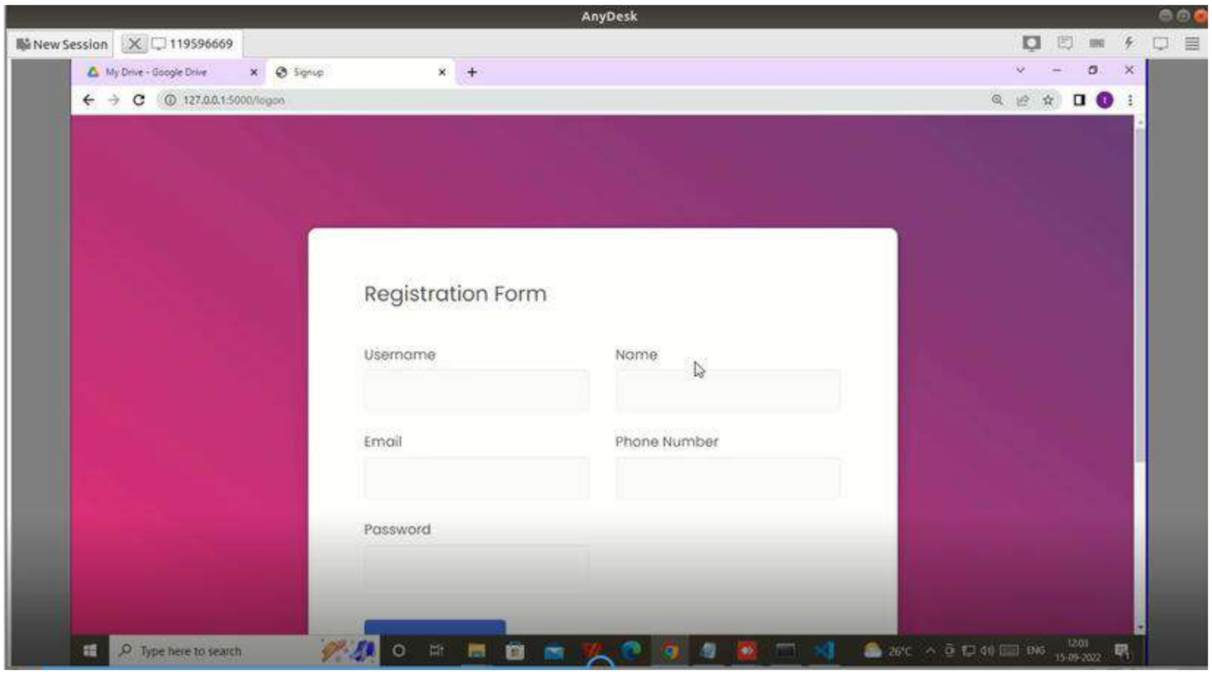


Fig. 23: Registration Page

If we are new to dashboard then we have to register to use the application

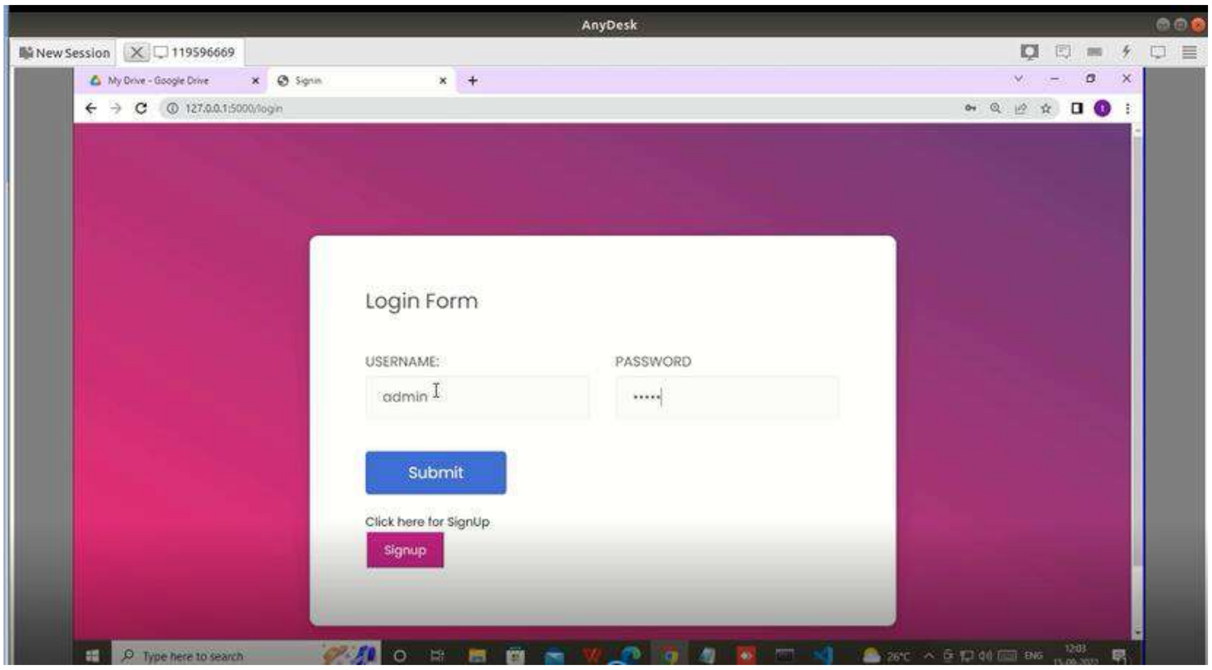
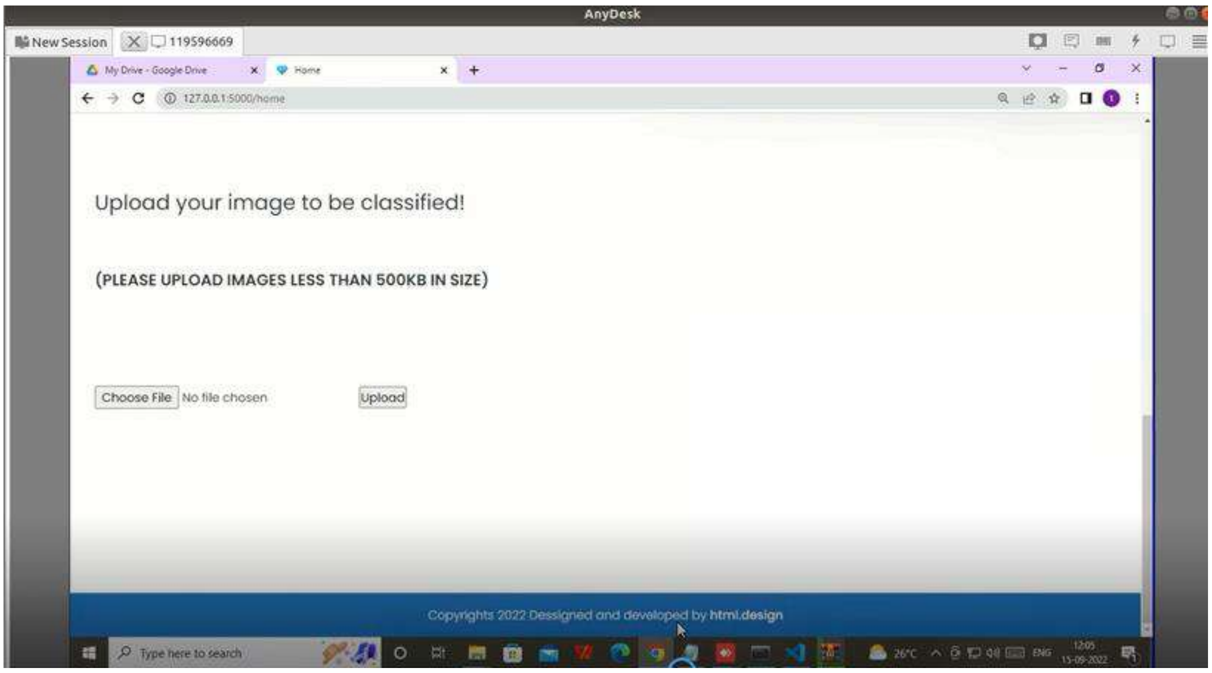


Fig. 24: Login Page

If we are already a member of this dashboard then directly we can login to the page


Upload your image to be classified!

(PLEASE UPLOAD IMAGES LESS THAN 500KB IN SIZE)

Fig. 25: Main Page

Upload the image for analysis

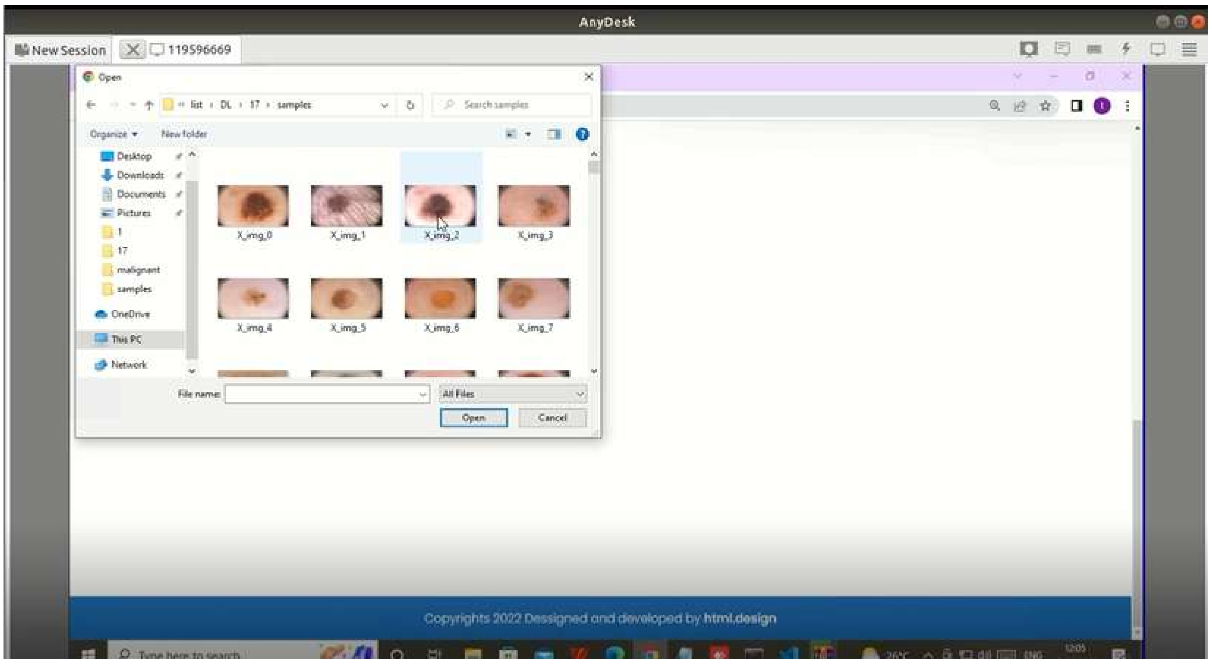


Fig. 26: Upload Input Images

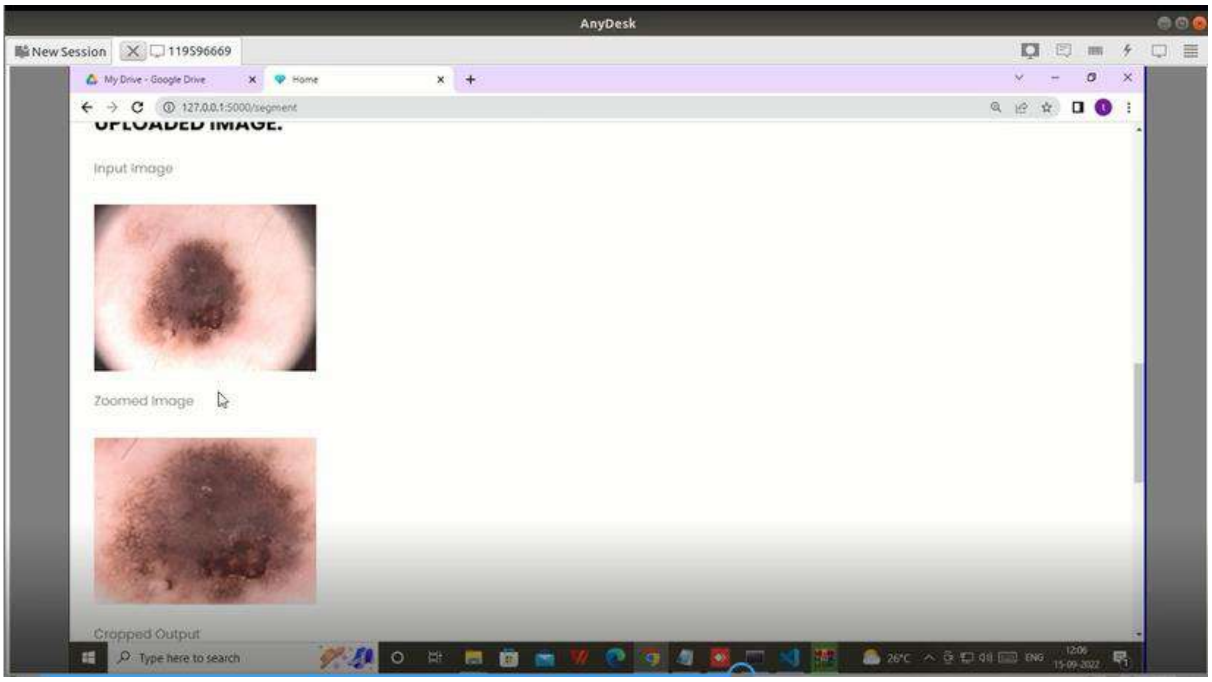
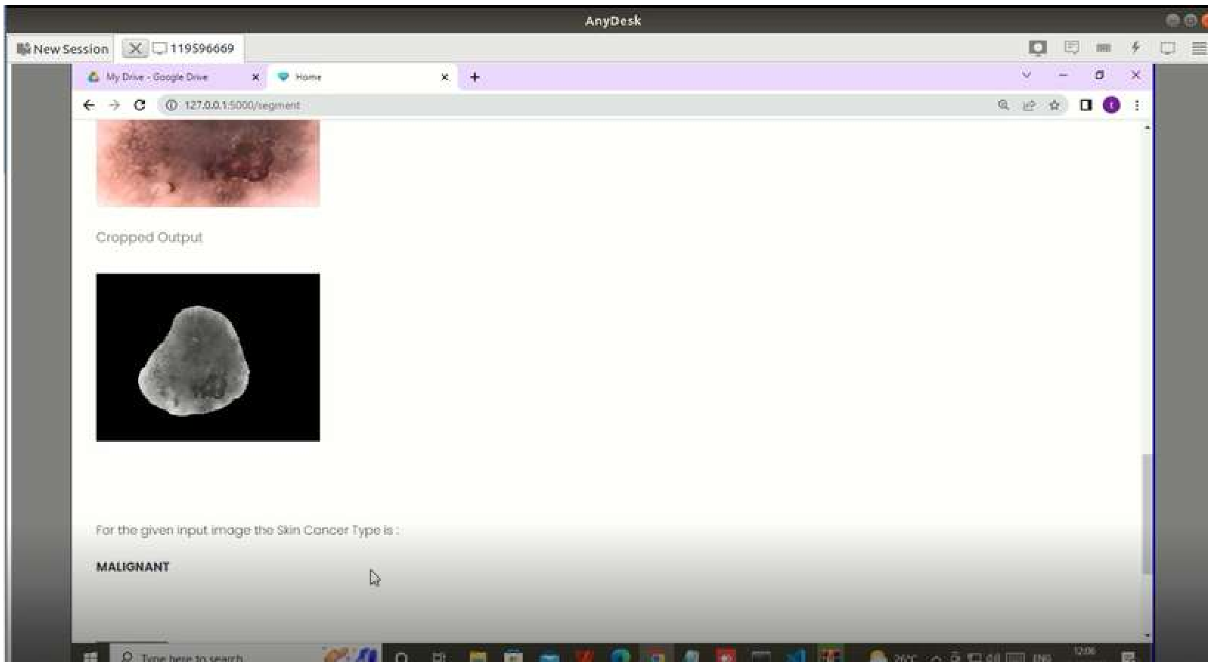


Fig. 27: Input Image

The upload image is first segmented and then processed with image augmentation using keras

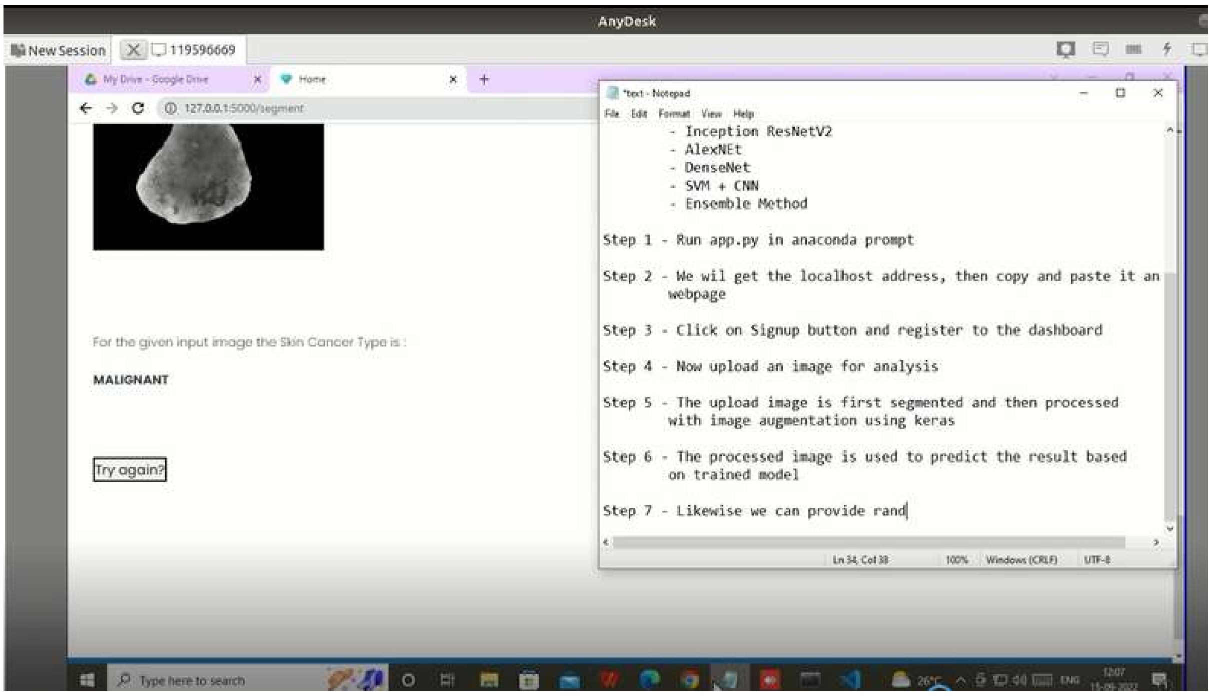


For the given input image the Skin Cancer Type is :

MALIGNANT

Fig. 28: Cropped Output Images

The process image is used to predict the result based on trained model



For the given input image the Skin Cancer Type is :

MALIGNANT

```
        - Inception ResNetV2
        - AlexNEt
        - DenseNet
        - SVM + CNN
        - Ensemble Method

Step 1 - Run app.py in anaconda prompt

Step 2 - We wil get the localhost address, then copy and paste it an
        webpage

Step 3 - Click on Signup button and register to the dashboard

Step 4 - Now upload an image for analysis

Step 5 - The upload image is first segmented and then processed
        with image augmentation using keras

Step 6 - The processed image is used to predict the result based
        on trained model

Step 7 - Likewise we can provide rand
```

Fig. 29: Prediction Result

CHAPTER-7

CONCLUSION AND FUTURE SCOPE OF STUDY

7.1 Conclusion:

The discoveries of this study propose that, no matter what the world class show that has been represented in the writing, the Exchange Learning technique that is regularly utilized will in all probability be incapable. The discoveries of the primary examination, specifically, show that even slight changes to the first preparation dataset can radically diminish a classifier's presentation. The most recent developments in [5] are supported by these observations. Additionally, AlexNet is the most reliable Exchange Learning organization, as demonstrated by our data. In addition, all CNN networks that were in operation utilized conventional ACC because there was neither division nor expansion of information. Continuous retraining is necessary to minimize performance degradation because the best classifier requires numerous training rounds. The second experiment, which made use of a Cloud/Fog/Edge architecture to allow for ongoing retraining, was prompted by these findings. We were able to reduce processing time by up to 76% by performing the necessary continuous retraining step to improve the classifier. As a result, we are able to draw the following conclusion: Imagining disseminated engineering could be beneficial to the final customer in a variety of ways: the accumulation of information "on the organization" for the purpose of enhancing picture data sets and assisting in the early detection of melanoma; handling urgent information locally, at the organization's edge, with the ability to handle local information, which reduces information handling idleness, constant responsiveness, and costs. When it comes to pre-handling and sorting photos of melanoma, this kind of engineering execution goes above and beyond standard practices to meet a new demand. It involves uploading photographs to a cloud service or a central data server for processing. While increasing capacity, decentralization has the potential to speed up computation. The arranged cross variety configuration's absolute action is portrayed beneath. In the cloud, information containers are recharged, and framework arranging is done. At the Haze location where services are held, it is the responsibility of the orchestrator to provide advanced services following each arrangement. On IoMT devices (such as smartphones), local computations are carried out in the Edge region. A component of the IoMT device's Fog arrangement, HiC-Otsu performs a fundamental analysis of the provided data. The content is explained by a QoS mediator to help implement the framework. The typical customer makes use of the services that are provided, but by stacking data, he helps the framework grow its data store.

7.2 Future Scope :

In order to expand on our findings and improve picture learning, we intend to develop more reliable neural network models in the near future. Our data indicate that CNN networks performed better without segmentation. Training should take into account information in the skin surrounding lesions, according to these data. In order to evaluate other approaches to pre-processing, Experiment 1 should be repeated. It was unable to determine the minimum number of training steps that were required to achieve remarkable strength. This issue ought to be the focus of future research in order to cut down on preparation time. The effects of a dispersed environment on time investment money (RT and clock time) are the focus of the following analysis. A more in-depth examination of distributed engineering may be required to determine whether a more complex design might make implementation possible.

REFERENCES

[1] A. Esteva et al., “Dermatologist-level classification of skin cancer with deep neural networks,” Nature, vol. 542, no. 7639, pp. 115–118, 2017.

[2] N. R. Abbasi et al., “Early diagnosis of cutaneous melanoma: Revisiting the ABCD criteria,” JAMA Dermatol., vol. 292, no. 22, pp. 2771–2776, 2004.

[3] S. Chatterjee, D. Dey, S. Munshi, and S. Gorai, “Dermatological expert system implementing the ABCD rule of dermoscopy for skin disease identification,” Expert Syst. Appl., vol. 167, 2021, Art. no. 114204.

[4] K. Møllersen, H. Kirchesch, M. Zortea, T. R. Schopf, K. Hindberg, and F. Godtliebsen, “Computer-aided decision support for melanoma detection applied on melanocytic and nonmelanocytic skin lesions: A comparison of two systems based on automatic analysis of dermoscopic images,”BioMed. Res. Int., vol. 2015, 2015.

[5] M. Goyal, T. Knackstedt, S. Yan, and S. Hassanpour, “Artificial intelligence-based image classification methods for diagnosis of skin cancer: Challenges and opportunities,” Comput. Biol. Med., vol. 127, 2020, Art. no. 104065. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010482520303966 .

[6] V. Shah, P. Autee, and P. Sonawane, “Detection of melanoma from skin lesion images using deep learning techniques,” in Proc. Int. Conf. Data Sci. Eng., 2020, pp. 1–8.

[7] A. Adegun and S. Viriri, “Deep learning techniques for skin lesion analysis and melanoma cancer detection: A survey of state-of-the-art,” Artif. Intell. Rev., vol. 54, no. 2, pp. 811–841, 2021.

[8] M. Garey, D. Johnson, and H. Witsenhausen, “The complexity of the generalized Lloyd-Max problem (corresp.),” IEEE Trans. Inf. Theory, vol. 28, no. 2, pp. 255–256, Mar. 1982.

[9] S. J. Pan and Q. Yang, “A survey on transfer learning,” IEEE Trans. Knowl. Data Eng., vol. 22, no. 10, pp. 1345–1359, 2009.

[10] Y. Zhou and Z. Song, “Binary decision trees for melanoma diagnosis,” in Proc. Int. Workshop Mult. Classifier Syst., 2013, pp. 374–385.

[11] S. Gilmore, R. Hofmann-Wellenhof, and H. P. Soyer, “A support vector machine for decision support in melanoma recognition,” Exp. Dermatol., vol. 19, no. 9, pp. 830–835, 2010.

[12] A. Tenenhaus, A. Nkengne, J.-F. Horn, C. Serruys, A. Giron, and B. Fertil, “Detection of melanoma from dermoscopic images of naevi acquired under uncontrolled conditions,” Skin Res. Technol., vol. 16, no. 1, pp. 85–97, 2010.

[13] D. Ruiz, V. Berenguer, A. Soriano, and B. Sánchez, “A decision support system for the diagnosis of melanoma: A comparative approach,” Expert Syst. Appl., vol. 38, no. 12, pp. 15217–15223, 2011.

[14] Z. Hu, J. Tang, Z. Wang, K. Zhang, L. Zhang, and Q. Sun, “Deep learning for image-based cancer detection and diagnosis–A survey,” Pattern Recognit., vol. 83, pp. 134–149, 2018.

[15] E. Nasr-Esfahani et al., “Melanoma detection by analysis of clinical images using convolutional neural network,” in Proc. 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc., 2016, pp. 1373–1376.