

A Project Report

on

Handwritten Character Recognition Using Deep Learning

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

BACHELOR OF TECHNOLOGY

in

Computer Science & Engineering

by

K.Shreya **184G1A0588**

P.Rupeshwara Reddy **184G1A0569**

G.Rupa **184G1A0567**

V.Sai Sooraj **184G1A05B8**

Under the Guidance of

Mrs S.Sunitha , M.Tech., (Ph.D)

Assistant Professor



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY:

(Affiliated to JNTUA, Accredited by NAAC with 'A' Grade, Approved by AICTE,
New Delhi & Accredited by NBA (EEE, ECE & CSE))

Rotarypuram village, B K Samudram Mandal, Ananthapuramu- 515701.

2021-2022

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA, Accredited by NAAC with 'A' Grade, Approved by AICTE,
New Delhi & Accredited by NBA (EEE, ECE & CSE))
Rotarypuram village, B K Samudram Mandal, Ananthapuramu- 515701.



Certificate

This is to certify that the project report entitled **Handwritten Character Recognition using Deep Learning** is the bonafide work carried out by **K.Shreya** bearing Roll Number **184G1A0588**, **P.Rupeshwara Reddy** bearing Roll Number **184G1A0569**, **G.Rupa** bearing Roll Number **184G1A0567** and **V.Sai Sooraj** bearing Roll Number **184G1A05B8** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2021-2022.

Guide

Mrs S.Sunitha,M.Tech., (Ph.D)
Assistant Professor

Head of the Department

Mr. P. Veera Prakash, M.Tech.(Ph.D)
Assistant Professor & HOD

Date:

EXTERNAL EXAMINER

Place: Rotarypuram

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express my gratitude for all of them.

It is with immense pleasure that we would like to express our indebted gratitude to our Guide **Mrs S.Sunitha M.Tech., (Ph.D), Assistant Professor, Computer Science and Engineering Department**, who has guided us a lot and encouraged us in every step of the project work. We thank her for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We express our deep felt gratitude to **Mr. K. Venkatesh M.Tech., Assistant Professor** project coordinator valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We are very much thankful to **Mr. P. Veera Prakash M.Tech., (Ph.D), Assistant Professor & Head of the Department, Computer Science & Engineering**, for his kind support and for providing necessary facilities to carry out the work.

We wish to convey our special thanks to **Dr. G. Balakrishna, Ph.D, Principal of Srinivasa Ramanujan Institute of Technology** for giving the required information in doing our project work. Not to forget, we thank all other faculty and non-teaching staff, and my friends who had directly or indirectly helped and supported us in completing our project in time.

We also express our sincere thanks to the Management for providing excellent facilities.

Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

184G1A0588

184G1A0569

184G1A0567

184G1A05B8

DECLARATION

We, Ms. K.Shreya bearing reg no: 184G1A0588, Mr. P.Rupeshwara Reddy bearing reg no: 184G1A0569, Ms. G.Rupa bearing reg no:184G1A0554, Mr. V.Sai Sooraj bearing reg no: 184G1A05B8, students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, Rotarypuram, hereby declare that the dissertation entitled “HANDWRITTEN CHARACTER RECOGNITION USING DEEP LEARNING” embodies the report of our project work carried out by us during IV Year Bachelor of Technology under the guidance of Mrs S.Sunitha,,M.Tech., (Ph.D), Department of CSE and this work has been submitted for the partial fulfillment of the requirements for the award of Bachelor of Technology degree.

The results embodied in this project report have not been submitted to any other Universities of Institute for the award of Degree.

K.SHREYA

Reg no:184G1A0588

P.RUPESHWARA REDDY

Reg no:184G1A0569

G.RUPA

Reg no:184G1A0567

V.SAI SOORAJ

Reg no:184G1A05B8

CONTENTS	Page no
List of figures	VIII
List of abbreviations	IX
Abstract	X
Chapter 1: Introduction	1
Chapter 2: Literature Survey	2
2.1 Existing System	4
Chapter 3: Proposed System & Analysis	5
3.1 Proposed System	5
3.2 Feasibility Study	6
3.3 Introduction to Python	9
Chapter 4: Design	28
4.1 UML Diagrams	28
4.2 Architecture	36
Chapter 5: Implementation	40
5.1 Algorithms	40
5.2 Modules	43
5.3 Steps for execution	45
Chapter 6: Testing	46
6.1 Unit Testing	46
6.2 Integration Testing	47
6.3 Functional Testing	47
6.4 White Box Testing	48
6.5 Black Box Testing	48
6.6 Acceptance Testing	49
Chapter 7: Execution and Results	50
7.1 Execution	50
7.2 Results	51

Future work	53
Conclusion	54
References	55

LIST OF FIGURES

Fig No.	Title	Page No.
Fig.3.	Flow of the project	6
Fig.4.1.1	Use Case Diagram	29
Fig.4.1.2	Class Diagram	30
Fig.4.1.3	Deployment Diagram	30
Fig.4.1.4	Sequence Diagram	31
Fig.4.1.5	Collaboration Diagram	32
Fig.4.1.6	Activity Diagram	33
Fig.4.1.7	Component Diagram	33
Fig.4.1.8	ER Diagram	34
Fig.4.1.9	DFD Diagram	35
Fig.4.2.1	Project Architecture	36
Fig.4.3.1	Python Website	37
Fig.4.3.2	Pycharm Wizard	38
Fig.4.3.3	Pycharm logo	39
Fig.4.3.4	Numpy Installation	39
Fig.5.1.1	Convolution Operation	40
Fig.5.1.2	Convolution Layer	41
Fig.5.1.3	Relu Layer	42
Fig 5.1.4	CNN	43
Fig.7.1	Run time Environment	50
Fig.7.2.1	Home Page	51
Fig.7.2.2	Upload Page	52
Fig.7.2.3	Output	53

LIST OF ABBREVIATIONS

CSV	Comma Separated Values
CNN	Convolutional Neural Networks
CV	Computer Vision
UML	Unified Modelling Language
ML	Machine Learning
DL	Deep Learning
DFD	Data Flow Diagram
ER	Entity Relationship
HTML	Hyper Text Markup Language
XML	Extensible Markup Language
OOP	Object Oriented Programming

ABSTRACT:

In this project we present an innovative method for handwritten character detection using deep neural networks. In today world it has become easier to train deep neural networks because of availability of huge amount of data and various Algorithmic innovations which are taking place. Now-a-days the amount of computational power needed to train a neural network has increased due to the availability of GPU's and other cloud-based services like Google Cloud platform and Amazon Web Services which provide resources to train a Neural network on the cloud. We have designed a image segmentation based Handwritten character recognition system. In our system we have made use of OpenCV for performing Image processing and have used TensorFlow for training a neural Network. We have developed this system using python programming language.

KEYWORDS: Handwritten, Character, segmentation, Neural Networks,

Deep Learning.

CHAPTER – 1

INTRODUCTION

The purpose of this project is to take handwritten English characters as input, process the character, train the neural network algorithm, to recognize the pattern or structure of characters, and matches the character or the output with the desired input. Pattern recognition is perhaps the most common use of neural networks. The neural network is presented with a input which contains the pattern information, this could be an image, and hand written data. The neural network then attempts to determine if the input data matches a pattern that the neural network has mentioned. A neural network trained for classification is designed to take input samples and classify them into groups. These input groups may be fuzzy, without clearly defined boundaries. This project concerns detecting free handwritten characters.

In this project we present an innovative method for offline handwritten character detection using deep neural networks. In today world it has become easier to train deep neural networks because of availability of huge amount of data and various Algorithmic innovations which are taking place. Now-a-days the amount of computational power needed to train a neural network has increased due to the availability of GPU's and other cloud-based services like Google Cloud platform and Amazon Web Services which provide resources to train a neural network on the cloud. We have designed a image segmentation based Handwritten character recognition system. In our system we have made use of OpenCV for performing Image processing and have used Tensor Flow for training a neural Network. We have developed this system using python programming language.

CHAPTER – 2

LITERATURE SURVEY

- [1] Yuauf Perwej, Ashish Chaturvedi, “Neural networks for Handwritten English Alphabet Recognition, International Journal of Computer Application 0975-8887) Volume-20No.7, April 2011”.**

This project demonstrates the use of neural networks for developing a system that can recognize hand-written English alphabets. In this system, each English alphabet is represented by binary values that are used as input to a simple feature extraction system, whose output is fed to our neural network system

- [2] Savitha Attigeri,” Neural networks based Handwritten Character Recognition System, International Journal of Engineering and Computer Science, p.No.:23761-23768”**

Handwritten character recognition has been one of the active and challenging research areas in the field of image processing and pattern recognition. It has numerous applications which include, reading aid for blind, bank cheques and conversion of any hand written document into structural text form. In this paper an attempt is made to recognize handwritten characters for English alphabets without feature extraction using multilayer Feed Forward neural network. Each character data set contains 26 alphabets. Fifty different character data sets are used for training the neural network. The trained network is used for classification and recognition. In the proposed system, each character is resized into 30x20 pixels, which is directly subjected to training. That is, each resized character has 600 pixels and these pixels are taken as features for training the neural network. The results show that the proposed system yields good recognition rates which are comparable to that of feature extraction based schemes for handwritten character recognition.

[3] Chirag I Patel, Ripal Patel, Palak Patel, “Handwritten character recognition using neural network”, International Journal of Scientific and Research Volume 2, Issue may 2011”.

Objective is this paper is recognize the characters in a given scanned documents and study the effects of changing the Models of ANN. Today Neural Networks are mostly used for Pattern Recognition task. The paper describes the behaviors of different Models of Neural Network used in OCR. OCR is widespread use of Neural Network. We have considered parameters like number of Hidden Layer, size of Hidden Layer and epochs. We have used Multilayer Feed Forward network with Back propagation. In Preprocessing we have applied some basic algorithms for segmentation of characters, normalizing of characters and De-skewing. We have used different Models of Neural Network and applied the test set on each to find the accuracy of the respective Neural Network.

[4] Anita Pal & Dayashankar Singh, “Handwritten English Character Recognition using Neural Network, International Journal of Computer Science & Communication, Volume-1, No.2 , july-December 2010, pp.141-144.”

The main aim of this project is to design expert system for , “HCR (English) using Neural Network”. that can effectively recognize a particular character of type format using the Artificial Neural Network approach. Neural computing Is comparatively new field, and design components are therefore less well specified than those of other architectures. Neural computers implement data parallelism. Neural computer are operated in way which is completely different from the operation of normal computers. Neural computer are trained (not Programmed) so that given a certain starting state (data input); they either classify the input data into one of the number of classes or cause the original data to evolve in such a way that a certain desirable property is optimized.

2.1 EXISTING SYSTEM

There are many are many fields in Handwriting Recognition is one of the active areas of research where deep neural networks are being utilized. Recognizing handwriting is an easy task for humans but a daunting task for computers. Handwriting recognition systems are of two types: Online and Offline. In online handwriting recognition system the handwriting of the user is recognized as the user is writing. The information like the order in which the user has made the strokes is also available. But in offline handwriting recognition system, the handwriting of user is available as an image. Handwriting recognition is a challenging task because of many reasons. The primary reason is that different people have different styles of writing. The secondary reason is there are lot of characters like Capital letters, Small letters, Digits and Special symbols. However even the most modern and commercially available systems have not been able to achieve such a high accuracy loss.

2.2 DISADVANTAGES:

- Failure for the data maintained.
- Expensive.
- Inefficient.
- Requires planning and time.

CHAPTER-3

PROPOSED SYSTEM & ANALYSIS

3.1 Proposed System

To overcome the limitation of the existing system, we develop a hand writing character recognition system which uses deep learning models in conjunction with computer vision to achieve good accuracies in characters detection and recognition. Convolutional Neural Networks (CNN) are employed as a deep learning model for character recognition and classification. Computer Vision using Open CV is used for character detection and image pre-processing purposes.

ADVANTAGES:

- High efficiency.
- High Scalable.
- Inexpensive.
- Efficient.

APPLICATIONS:

- Used by technology companies for hand writing recognition.
- Used to convert physical documents into digital by OCR.

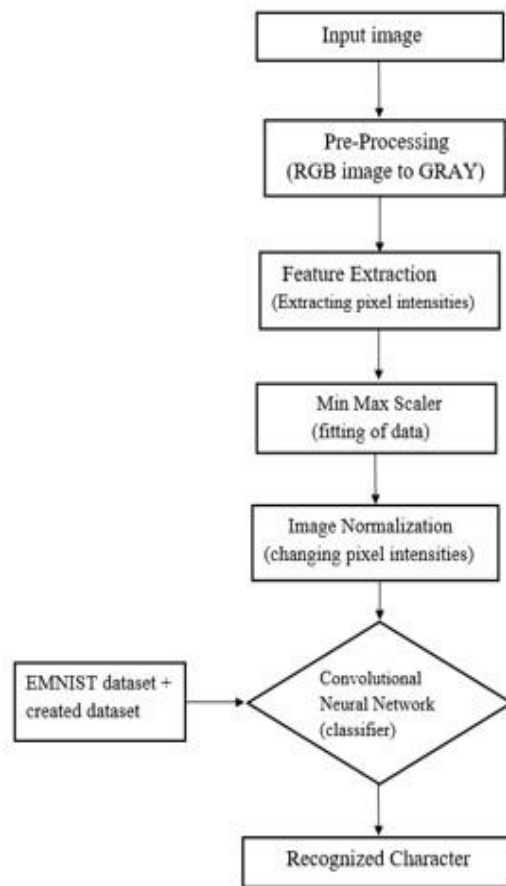


Fig. 3.1 Flow of the project

3.2 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY

□ SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.2.1 Functional and non-functional requirements:

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types:

Functional Requirements:

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Non-functional requirements:

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

3.2.2 Hardware and Software Specifications:

H/W Specifications:

- Processor : I3/Intel Processor
- RAM : 8GB (min)
- Hard Disk : 128 GB

S/W Specifications:

- Operating System : Windows 10
- Server-side Script : Python 3.6
- IDE : PyCharm
- Libraries Used : Numpy, Flask, Keras, Tensor Flow.

3.3 INTRODUCTION TO PYTHON

□ Python

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)

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

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

- Great interactive environment

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

Python's features include –

- **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 vs. 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.

Python, however, doesn’t require this. You simply give your variables names and assign values to them. The interpreter takes care of keeping track of what kinds of objects your program is using. This also means 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 and interactive.

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

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

20 Python libraries

- 1.** Requests. The most famous http library written by Kenneth remits. It's a must have for every python developer.
- 2.** Scrappy. 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.
- 3.** Python. A guy toolkit for python. I have primarily used it in place of tinder. You will really love it.
- 4.** 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.
- 5.** SQLAlchemy. A database library. Many love it and many hate it. The choice is yours.
- 6.** Beautiful Soup. I know it's slow but this xml and html parsing library is very useful for beginners.
- 7.** Twisted. The most important tool for any network application developer.
It has a very beautiful ape and is used by a lot of famous python developers.
- 8.** Numbly. How can we leave this very important library? It provides some advance math functionalities to python.
- 9.** Skippy. When we talk about numbly then we have to talk about spicy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.
- 10.** Matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyzer.
- 11.** Pygmy. Which developer does not like to play games and develop them? This library will help you achieve your goal of 2d game development.

12. Piglet. A 3d animation and game creation engine. This is the engine in which the famous python port of mine craft was made
13. Pit. A GUI toolkit for python. It is my second choice after python for developing GUI's for my python scripts.
14. Pit. Another python GUI library. It is the same library in which the famous Bit torrent client is created.
15. Scaly. A packet sniffer and analyzer for python made in python.
16. Pywin32. A python library which provides some useful methods and classes for interacting with windows.
17. Notch. 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 its capacity is beyond that. Do check it out.
18. 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.
19. Simply. Simply can do algebraic evaluation, differentiation, expansion, complex numbers, etc. It is contained in a pure Python distribution.
20. I Python. I 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 numblly 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 class and objects

These are the building blocks of OOP. Class creates a new object. This object can be anything, whether an abstract data concept or a model of a physical object, e.g. a chair. Each class has individual characteristics unique to that class, including variables and methods. Classes are very powerful and currently “the big thing” in most programming languages. Hence, there are several chapters dedicated to OOP later in the book. The class is the most basic component of object-oriented programming. Previously, you learned how to use functions to make your program do something. Now will move into the big, scary world of Object-Oriented Programming (OOP). To be honest, it took me several months to get a handle on objects. When I first learned C and C++, I did great; functions just made sense for me.

Having messed around with BASIC in the early '90s, I realized functions were just like subroutines so there wasn't much new to learn. However, when my C++ course started talking about objects, classes, and all the new features of OOP, my grades definitely suffered. Once you learn OOP, you'll realize that it's actually a pretty powerful tool. Plus many Python libraries and APIs use classes, so you should at least be able to understand what the code is doing.

One thing to note about Python and OOP: it's not mandatory to use objects in your code in a way that works best; maybe you don't need to have a full-blown class with initialization code and methods to just return a calculation. With Python, you can get as technical as you want. As you've already seen, Python can do just fine with functions. Unlike languages such as Java, you aren't tied down to a single way of doing things; you can mix functions and classes as necessary in the same program. This lets you build the code

Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.

Here's a brief list of Python OOP ideas:

- The class statement creates a class object and gives it a name. This creates a new namespace.
- Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax:

ClassName.Attribute.

- Class attributes export the state of an object and its associated behavior.

These attributes are shared by all instances of a class.

- Calling a class (just like a function) creates a new instance of the class.

This is where the multiple copies part comes in.

- Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.
- Using the term self identifies a particular instance, allowing for per instance attributes. This allows items such as variables to be associated with a particular instance.

Inheritance

First off, classes allow you to modify a program without really making changes to it. To elaborate, by subclassing a class, you can change the behavior of the program by simply adding new components to it rather than rewriting the existing components. As we've seen, an instance of a class inherits the attributes of that class.

However, classes can also inherit attributes from other classes. Hence, a subclass inherits from a superclass allowing you to make a generic superclass that is specialized via subclasses.

The subclasses can override the logic in a superclass, allowing you to change the behavior of your classes without changing the superclass at all.

Operator Overloads

Operator overloading simply means that objects that you create from classes can respond to actions (operations) that are already defined within Python, such as addition, slicing, printing, etc.

Even though these actions can be implemented via class methods, using overloading ties the behavior closer to Python's object model and the object interfaces are more consistent to Python's built-in objects, hence overloading is easier to learn and use.

User-made classes can override nearly all of Python's built-in operation methods

Exceptions

I've talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program's flow, either intentionally or due to errors.

They are special events that can occur due to an error, e.g. trying to open a file that doesn't exist, or when the program reaches a marker, such as the completion of a loop.

Exceptions, by definition, don't occur very often; hence, they are the "exception to the rule" and a special class has been created for them. Exceptions are everywhere in Python.

Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

Here are just a few examples:

- Accessing a non-existent dictionary key will raise a Key Error exception.
- Searching a list for a non-existent value will raise a Value Error exception

- Calling a non-existent method will raise an Attribute Error exception.
- Referencing a non-existent variable will raise a Name Error exception.
- Mixing data types without coercion will raise a Type Error exception.

One use of exceptions is to catch a fault and allow the program to continue working; we have seen this before when we talked about files. This is the most common way to use exceptions. When programming with the Python command line interpreter, you don't need to worry about catching exceptions. Your program is usually short enough to not be hurt too much if an exception occurs. Plus, having the exception occur at the command line is a quick and easy way to tell if your code logic has a problem. However, if the same error occurred in your real program, it will fail and stop working. Exceptions can be created manually in the code by raising an exception.

It operates exactly as a system-caused exceptions, except that the programmer is doing it on purpose. This can be for a number of reasons. One of the benefits of using exceptions is that, by their nature, they don't put any overhead on the code processing. Because exceptions aren't supposed to happen very often, they aren't processed until they occur. Exceptions can be thought of as a special form of the if/elif statements. You can realistically do the same thing with if blocks as you can with exceptions.

However, as already mentioned, exceptions aren't processed until they occur; if blocks are processed all the time. Proper use of exceptions can help the performance of your program. The more infrequent the error might occur, the better off you are to use exceptions; using if blocks requires Python to always test extra conditions before continuing. Exceptions also make code management easier: if your programming logic is mixed in with error-handling if statements, it can be difficult to read, modify, and debug your program.

User-Defined Exceptions

I won't spend too much time talking about this, but Python does allow for a programmer to create his own exceptions.

You probably won't have to do this very often but it's nice to have the option when necessary. However, before making your own exceptions, make sure there isn't one of the built-in exceptions that will work for you. They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free.

Making your own exceptions involves object-oriented programming, which will be covered in the next chapter. To make a custom exception, the programmer determines which base exception to use as the class to inherit from, e.g. making an exception for negative numbers or one for imaginary numbers would probably fall under the Arithmetic Error exception class. To make a custom exception, simply inherit the base exception and define what it will do.

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. This 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 carved 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.

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()`, `camp()`, ...) 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.

Python XML Parser

XML is a portable, open source language that allows programmers to develop applications that can be read by other applications, regardless of operating system and/or developmental language.

What is XML? The Extensible Markup Language XML is a markup language much like HTML or SGML.

This is recommended by the World Wide Web Consortium and available as an open standard.

XML is extremely useful for keeping track of small to medium amounts of data without requiring a SQL-based backbone.

XML Parser Architectures and APIs the Python standard library provides a minimal but useful set of interfaces to work with XML. The two most basic and broadly used APIs to XML data are the SAX and DOM interfaces. Simple API for XML SAX: Here, you register callbacks for events of interest and then let the parser proceed through the document.

This is useful when your documents are large or you have memory limitations, it parses the file as it reads it from disk and the entire file is never stored in memory. Document Object Model DOM API : This is a World Wide Web Consortium recommendation wherein the entire file is read into memory and stored in a hierarchical tree – based form to represent all the features of an XML document.

SAX obviously cannot process information as fast as DOM can when working with large files. On the other hand, using DOM exclusively can really kill your resources, especially if used on a lot of small files. SAX is read-only, while DOM allows changes to the XML file. Since these two different APIs literally complement each other, there is no reason why you cannot use them both for large projects.

Python Web Frameworks

A web framework is a code library that makes a developer's life easier when building reliable, scalable and maintainable web applications.

Why are web frameworks useful?

Web frameworks encapsulate what developers have learned over the past twenty years while programming sites and applications for the web. Frameworks make it easier to reuse code for common HTTP operations and to structure projects so other developers with knowledge of the framework can quickly build and maintain the application.

Common web framework functionality

Frameworks provide functionality in their code or through extensions to perform common operations required to run web applications. These common operations include:

1. URL routing
2. HTML, XML, JSON, and other output format templating
3. Database manipulation
4. Security against Cross-site request forgery (CSRF) and other attacks
5. Session storage and retrieval

Not all web frameworks include code for all of the above functionality. Frameworks fall on the spectrum from executing a single use case to providing every known web framework feature to every developer. Some frameworks take the "batteries-included" approach where everything possible comes bundled with the framework while others have a minimal core package that is amenable to extensions provided by other packages.

Comparing web frameworks

There is also a repository called [compare-python-web-frameworks](#) where the same web application is being coded with varying Python web frameworks, templating engines and object.

Web framework resources

- When you are learning how to use one or more web frameworks it's helpful to have an idea of what the code under the covers is doing.
- Frameworks is a really well done short video that explains how to choose between web frameworks. The author has some particular opinions about what should be in a framework. For the most part I agree although I've found sessions and database ORMs to be a helpful part of a framework when done well.
- What is a web framework? Is an in-depth explanation of what web frameworks are and their relation to web servers?
- Jingo vs. Flask vs. Pyramid: Choosing a Python web framework contains background information and code comparisons for similar web applications built in these three big Python frameworks.

- This fascinating blog post takes a look at the code complexity of several Python web frameworks by providing visualizations based on their code bases.
- Python's web frameworks benchmarks is a test of the responsiveness of a framework with encoding an object to JSON and returning it as a response as well as retrieving data from the database and rendering it in a template. There were no conclusive results but the output is fun to read about nonetheless.
- What web frameworks do you use and why are they awesome? Is a language agnostic Reedit discussion on web frameworks? It's interesting to see what programmers in other languages like and dislike about their suite of web frameworks compared to the main Python frameworks.
- This user-voted question & answer site asked "What are the best general purpose Python web frameworks usable in production?" The votes aren't as important as the list of the many frameworks that are available to Python developers.

Web frameworks learning checklist

2. Choose a major Python web framework (Jingo or Flask are recommended) and stick with it. When you're just starting it's best to learn one framework first instead of bouncing around trying to understand every framework.
3. Work through a detailed tutorial found within the resources links on the framework's page.
4. Study open source examples built with your framework of choice so you can take parts of those projects and reuse the code in your application.
5. Build the first simple iteration of your web application then go to the deployment section to make it accessible on the web.

CHAPTER-4

DESIGN

4.1 UML DIAGRAMS

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. The UML represents a collection of best engineering practices that have proven successful insists the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.

2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

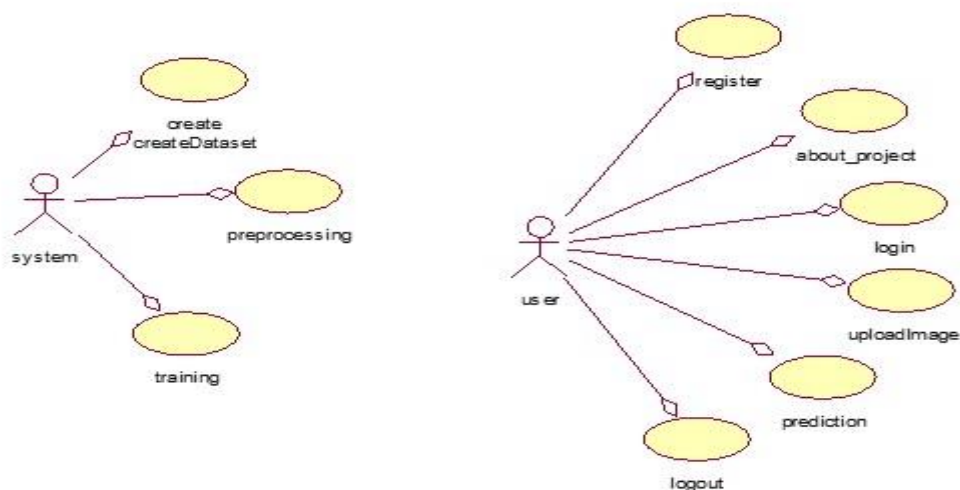


Fig. 4.1.1 .Use Case Diagram

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.

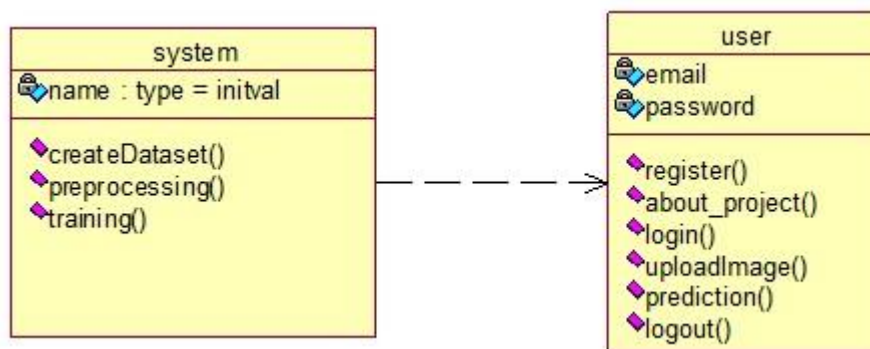


Fig.4.1.2. Class Diagram

Deployment Diagram:

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.



Fig.4.1.3. Deployment 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. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

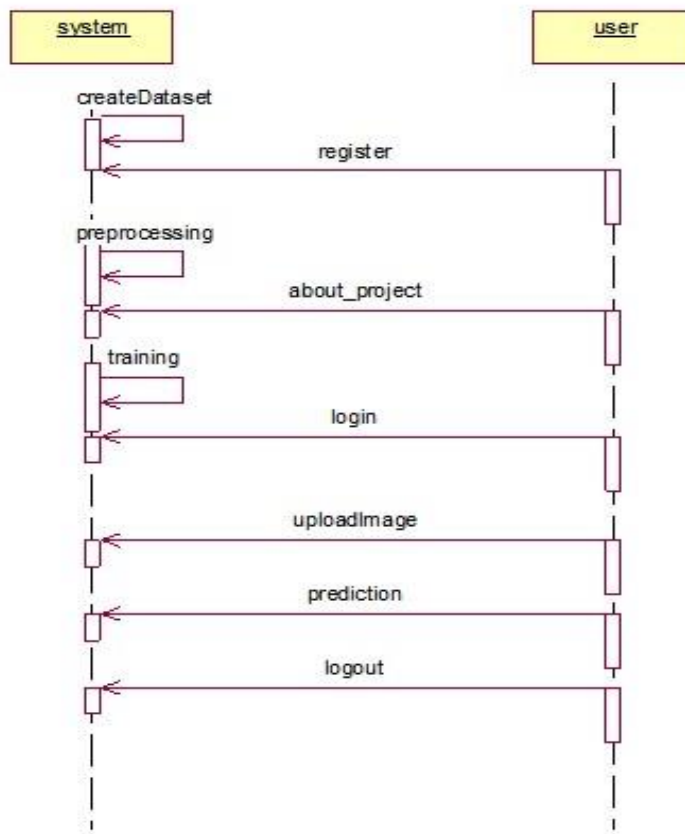


Fig .4.1.4 Sequence Diagram

Collaboration Diagram:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.

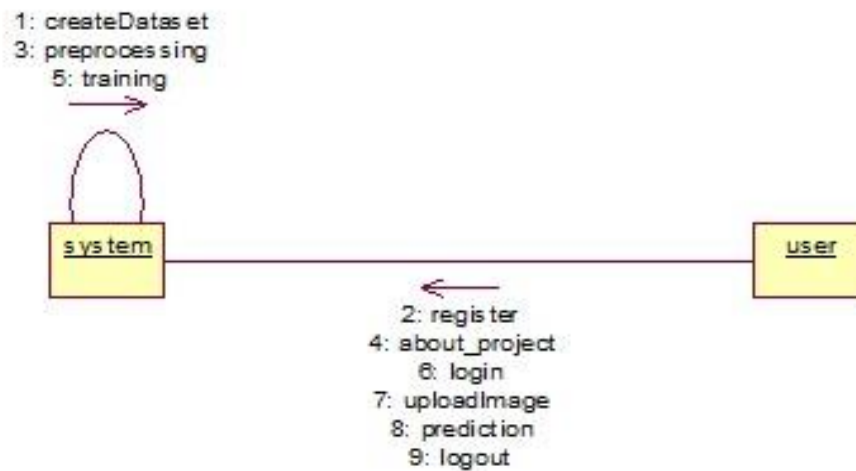


Fig.4.1.5. Collaboration Diagram

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.

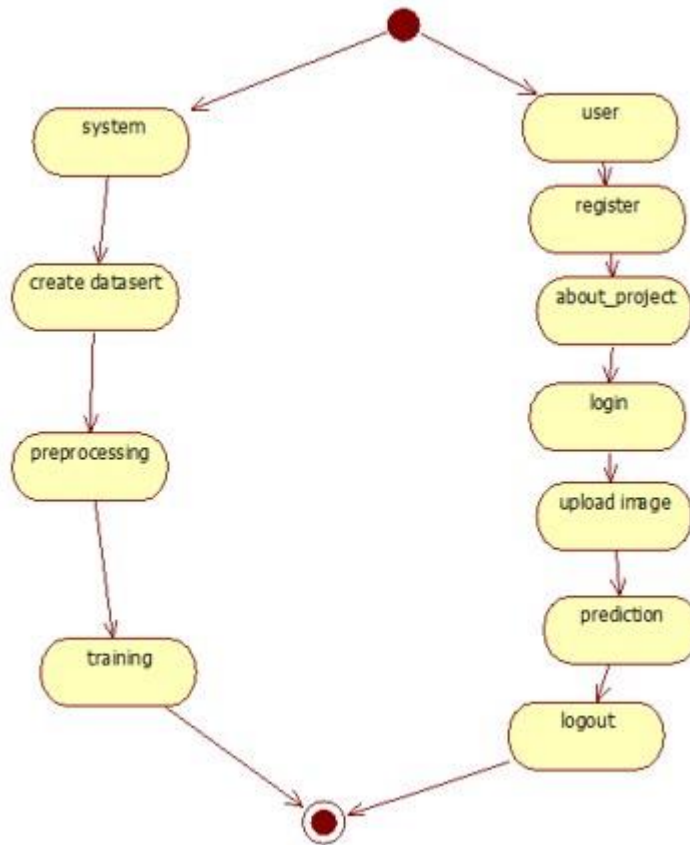


Fig.4.1.6.Activity Diagram

Component Diagram:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.



Fig.4.1.7. Component Diagram

ER Diagram:

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of ER model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

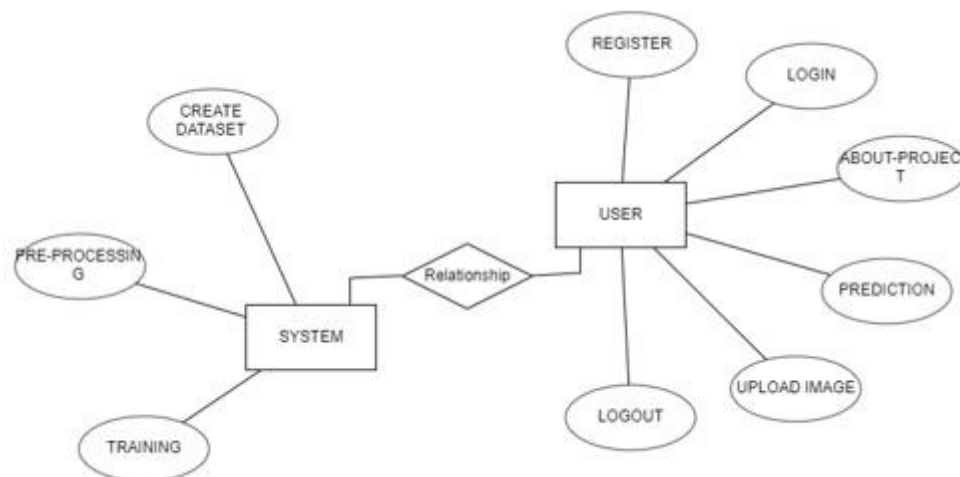


Fig.4.1.8. ER Diagram

DFD Diagram:

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information

and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

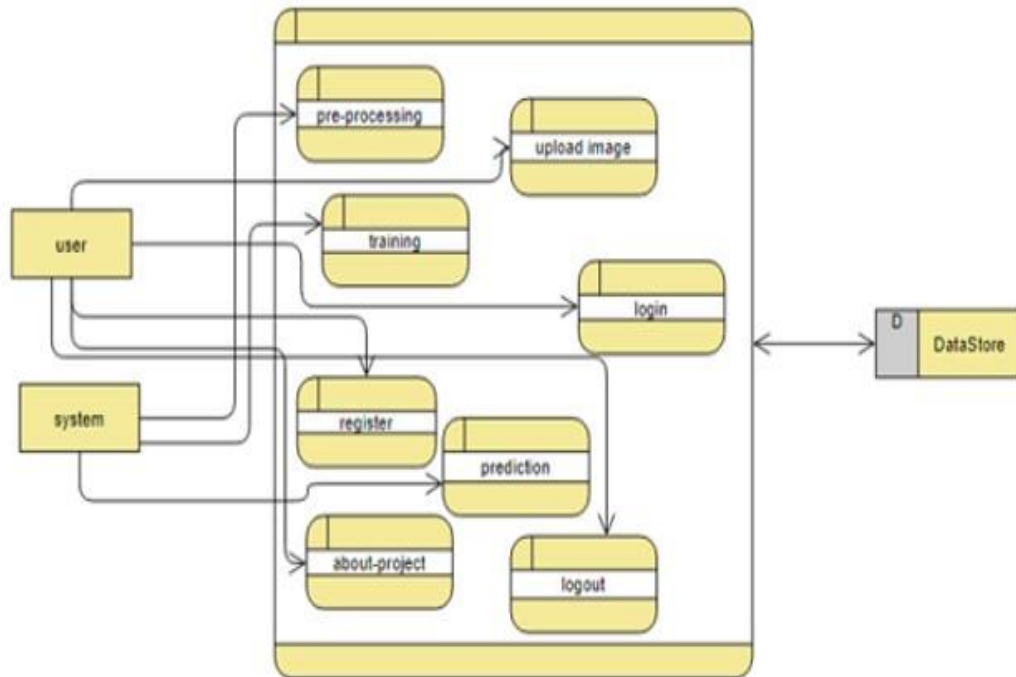


Fig.4.1.9.DFD Diagram

4.2. Architecture of the Project

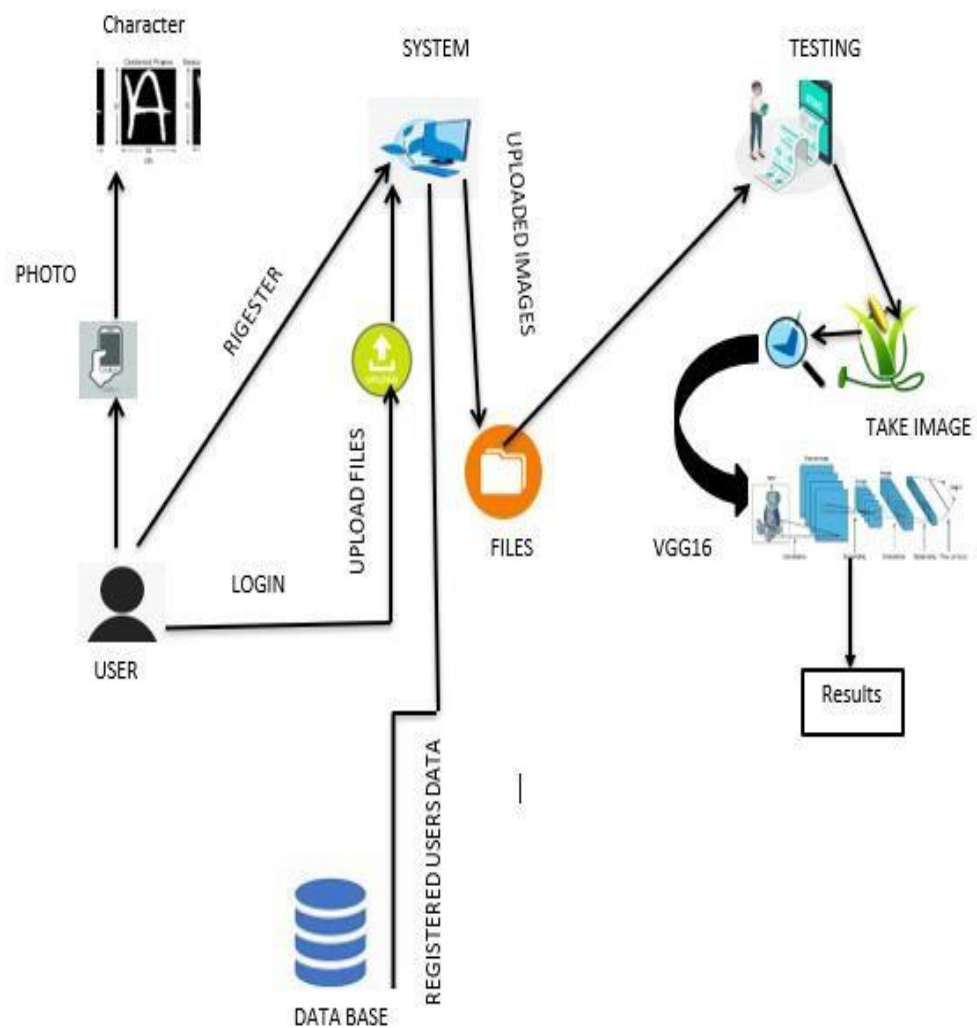


Fig.4.2.1. Project Architecture

SOFTWARE INSTALLATION FOR THIS PROJECTS:

Installing Python:

1. To download and install Python visit the official website of Python <https://www.python.org/downloads/> and choose your version.
2. Once the download is complete, run the exe for install Python. Now click on Install Now.
3. You can see Python installing at this point.
4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Installing PyCharm:

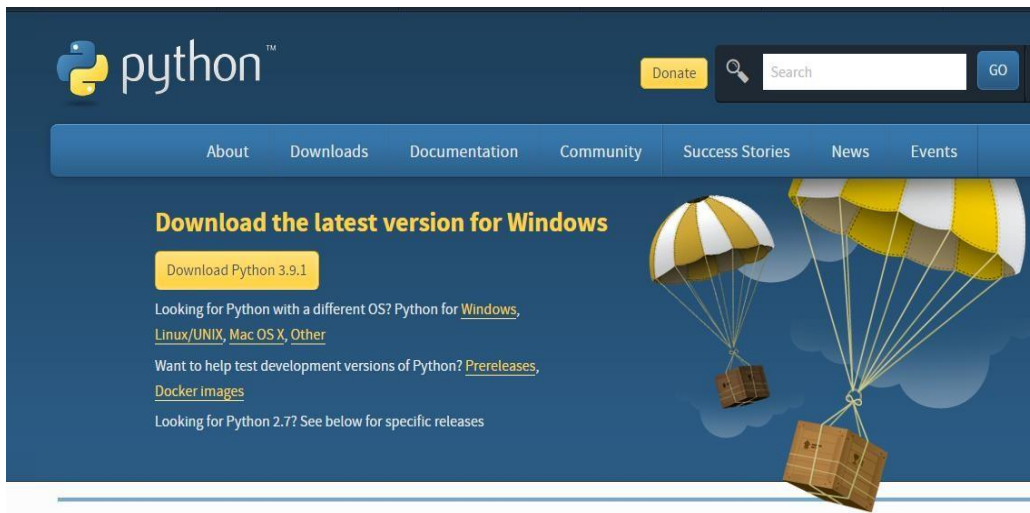


Fig.4.3.1.Python website

1. To download PyCharm visit the website <https://www.jetbrains.com/pycharm/download/> and Click the "DOWNLOAD" link under the Community Section.

Download PyCharm

[Windows](#) [Mac](#) [Linux](#)

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

Download

Free trial

Community

For pure Python development

Download

Free, open-source

Fig.4.3.2 Pycharm wizard

1. Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.
2. On the next screen, Change the installation path if required. Click “Next”.
3. On the next screen, you can create a desktop shortcut if you want and click on “Next”.
4. Choose the start menu folder. Keep selected Jet Brains and click on “Install”.
5. Wait for the installation to finish.
6. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
7. After you click on "Finish," the Following screen will appear.
8. 8.You need to install some packages to execute your project in a proper way.

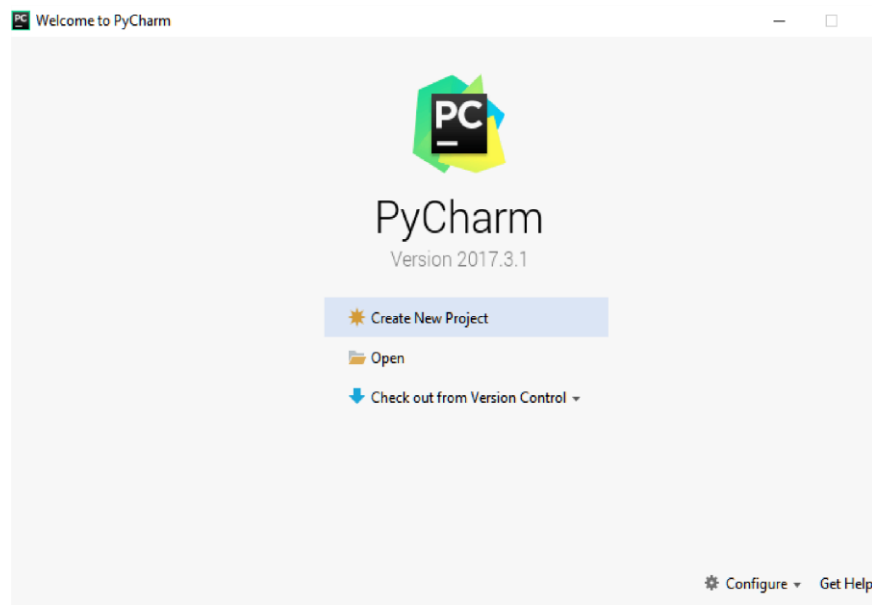


Fig .4.3.3 Pycharm logo

9. Open the command prompt/ anaconda prompt or terminal as administrator.

10.The prompt will get open, with specified path, type “pip install package name” which you want to install (like numpy, pandas, seaborn, scikit-learn, matplotlib.pyplot)

Ex: pip install numpy

```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
    | 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, b
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

Fig 4.3.4. Numpy installation

CHAPTER-5

IMPLEMENTATION

5.1 ALGORITHM

1. Convolutional Neural Network

Step1: convolutional 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.

The Convolution Operation

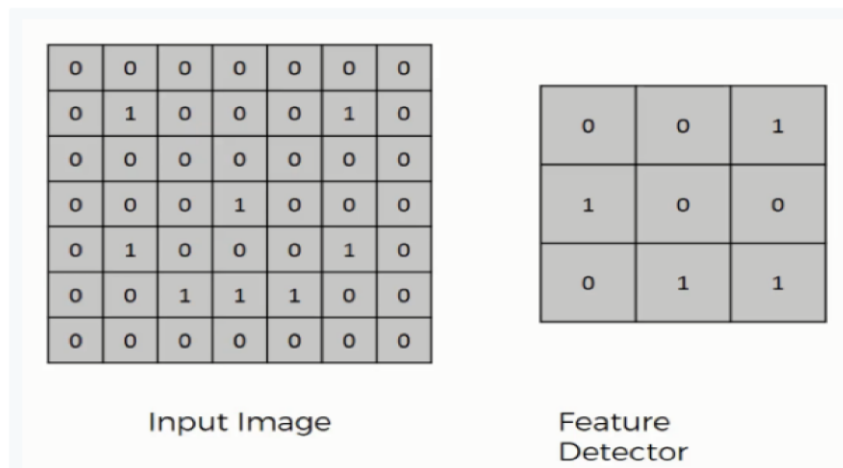


Fig 5.1.1 Convolution Operation

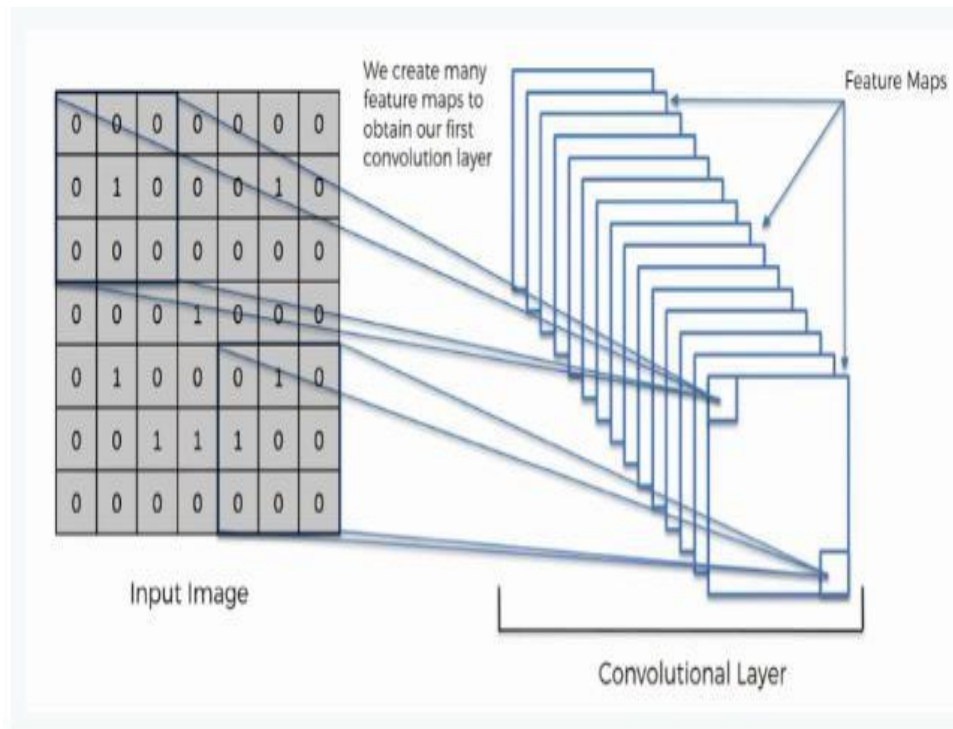


Fig 5.1.2 Convolution layer

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 Convolutional Neural Networks.

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

Convolutional Neural Networks Scan Images

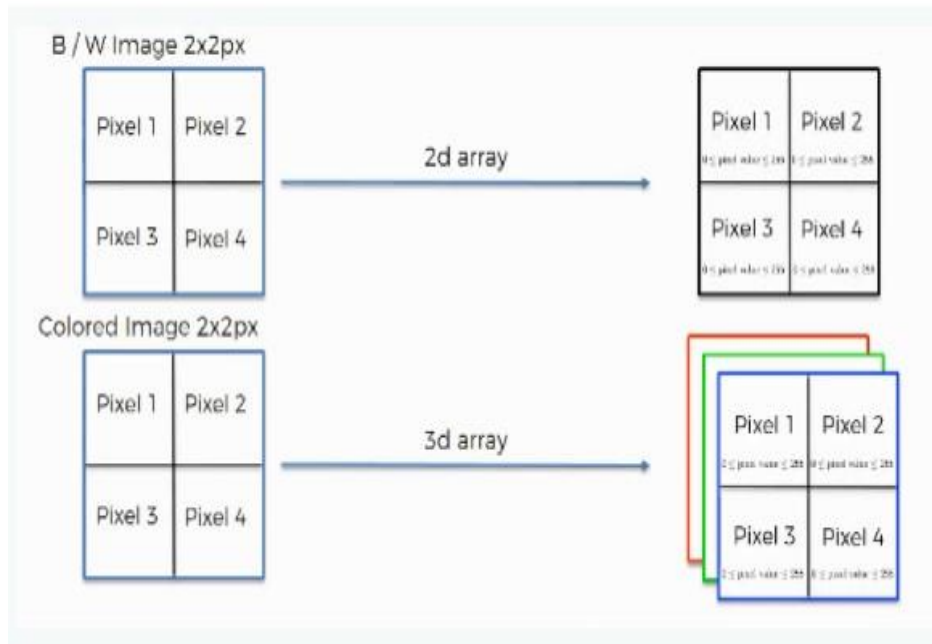


Fig . 5.1.3 Relu Layer

Step 2: Pooling Layer

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

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

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 how Convolutional Neural Networks operate and how the "neurons" that are finally produced learn the classification of images.

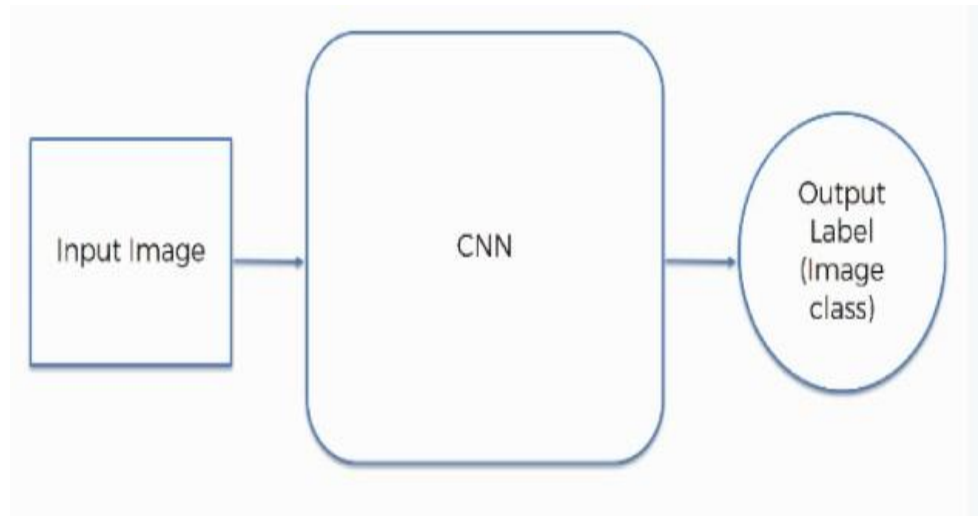


Fig 5.1.4 CNN

5.2 MODULES

5.2.1. System:

Create Dataset:

The dataset containing images of the desired objects to be recognize is split into training and testing dataset with the test size of 20-30%.

Pre-processing:

Resizing and reshaping the images into appropriate format to train our model.

Training:

Use the pre-processed training dataset is used to train our model using CNN algorithm.

5.2.2 User:

Register

The user needs to register and the data stored in MySQL database.

Login

A registered user can login using the valid credentials to the website to use a application.

About-Project

In this application, we have successfully created an application which takes to classify the images.

Upload Image

The user has to upload an image which needs to be classify the images.

Prediction

The results of our model will display the correct character which we have assigned to it.

Logout

Once the prediction is over, the user can logout of the application.

5.3 STEPS FOR EXECUTING THE PROJECTS

1. Install the required packages
2. Defining the custom model.
3. Loading the dataset.
4. Pre-Processing the dataset.
5. Training the custom model.
6. Loading the pre-trained CNN models.
7. Training the pre-trained model with our own dataset.
8. Performing prediction.
9. Create a Flask based User Interface.

CHAPTER-6

TESTING

SYSTEM TESTING:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

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.

System testing ensures that the entire integrated software system meets requirements. 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.

TYPES OF TESTINGS:

6.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. 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.

6.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to

identify Business process flows; 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.

6.4 White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

6.5 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.6 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results:

All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER-7

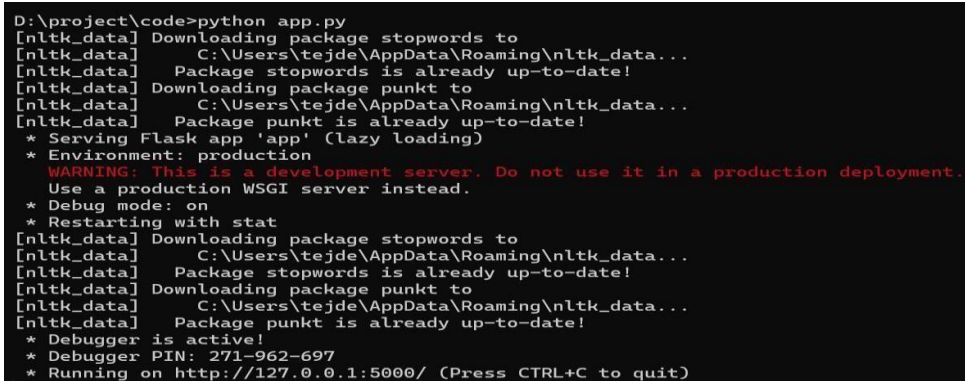
EXECUTION and RESULTS

7.1 Execution

To execute the code in PyCharm:

>python app.py

Run Time Environment:



```
D:\project\code>python app.py
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\tejde\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\tejde\AppData\Roaming\nltk_data...
[nltk_data] Package punkt is already up-to-date!
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\tejde\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\tejde\AppData\Roaming\nltk_data...
[nltk_data] Package punkt is already up-to-date!
* Debugger is active!
* Debugger PIN: 271-962-697
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Fig.7.1: Run Time Environment

7.2 Results

7.2.1 Home Page

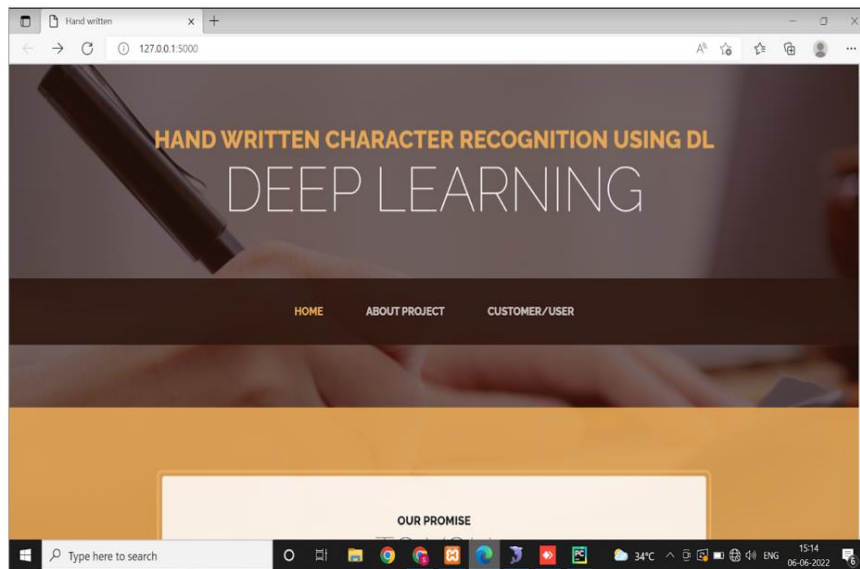


Fig 7.2.1 Home page

7.2.2 Upload Page

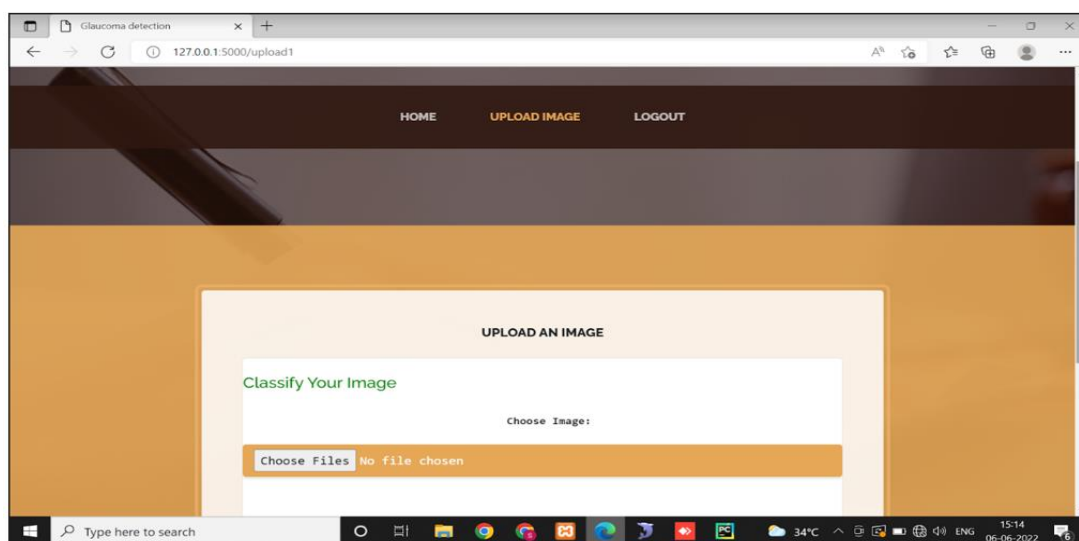


Fig 7.2.2 Upload Page

7.2.3 Output:

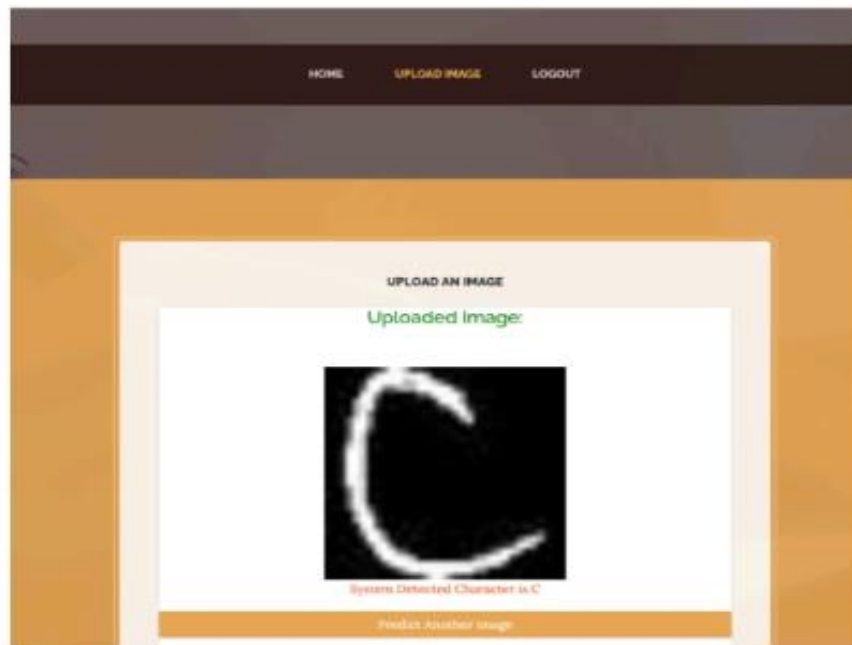


Fig 7.2.3 Output

FUTURE WORK

In future work, we would like to collect more characters and we have ability to add more characters further. We would like to add cross validation process in future in order to validate our results. We would also like to use better deep learning models and other state-of-the-art works and compare it with the results obtained. The developed model can be used in future to classify the remaining characters that includes both upper case and lower case characters.

CONCLUSION

In this project we have proposed a deep learning architecture with training character images and testing made on different characters and that correctly classifies our test images. The number of epochs used was stopped at particular number because we had received a cut point after which the accuracy was not improving and the loss was not decreasing on both training and validation data.

REFERENCES

- [1] Yuauf Perwej, Ashish Chaturvedi, “Neural networks for Handwritten English Alphabet Recognition, International Journal of Computer Application 0975-8887) Volume-20No.7, April 2011”.
- [2] Savitha Attigeri,” Neural networks based Handwritten Character Recognition System, International Journal of Engineering and Computer Science, p.No.:23761-23768”
- [3] Chirag I Patel, Ripal Patel, Palak Patel, “Handwritten character recognition using neural network”, International Journal of Scientific and Research Volume 2, Issue may 2011”.
- [4] Anita Pal & Dayashankar Singh, “Handwritten English Character Recognition using Neural Network, International Journal of Computer Science & Communication, Volume-1, No.2 , july-December 2010, pp.141144.”
- [5] N.Prameela , P.Anusha and R.Karthik, “Offline Handwritten Character Recognition”, IEEE Paper.
- [6] Aiquan Yuan , Gang Bai, Lijing jiao, Yajie Liy “Offline Handwritten Character Recognition based on Convolutional Neural Network” IEEE Paper.