

**AN END-TO-END UNIFIED METHODOLOGY FOR
PROTOTYPING OF GUI USING DEEP NEURAL NETURAL
NETWORK.**

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering Degree in Computer Science and Engineering

by

**KATTAMURI S S V SESA MANI PAVAN PRUDHVI (37110601)
PRATHI N N V RAMANJANEYULU (37110589)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119.**

MARCH - 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
Chennai - 600119
www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **KATTAMURI S S V SESA MANI PAVAN PRUDHVI (37110601), PRATHI N N V RAMANJANEYULU (37110589)** carried out the project entitled “**AN END-TO-END UNIFIED METHODOLOGY FOR PROTOTYPING OF GUI USING DEEP NEURAL NETWORK**” under my supervision from November 2020 to April 2021.

Internal Guide

DR. P. SARDAR MARAN.M.E.,Ph.D

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **KATTAMURI S S V SESA MANI PAVAN PRUDHVI (37110601)** and **PRATHI N N V RAMANJANEYULU (37110589)** hereby declare that the Project Report entitled “**A END-TO-END UNIFIED METHODOLOGY FOR PROTOTYPING OF GUI USING DEEP NEURAL NETURAL NETWORK**” done by us under the guidance of **DR. P. SARDAR MARAN.M.E.,Ph.D**, at Sathyabama Institute of Science and technology is submitted in partial fulfillment of the requirements of the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: Chennai-119

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to Board of management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

We convey our thanks to **Dr. T. SASIKALA M.E., Ph.D., Dean, School of Computing, Dr. S. VIGNESHWARI M.E., Ph.D. and Dr. L. LAKSHMANAN M.E., Ph.D., Heads of the Department,** Department of Computer Science and Engineering for providing us the necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr.P.SARDARMARAN,M.E.,PH.D.,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science And Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Before the release of desktop application products, a lot of repeated testing is often required. In the process of desktop application testing, the core problem is to locate the UI components on the desktop application screenshots. There are many methods to automatically identify UI components, but in some cases, such as crowdsourcing testing, it is difficult to use automatic methods to identify UI components.

In view of this, the desktop UI components recognition system based on image understanding provides new solutions and methods for application scenarios that are difficult to automatically locate components. We investigate desktop UI component information, use image understanding analysis to extract component images on screenshot, design and implement a convolutional neural networks, and then use trained CNN to classify these image.

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	vi
	LIST OF FIGURES	vi
1	INTRODUCTION	1
	1.1 OVERVIEW	1
	1.2 SCOPE OF THE PROJECT	1
	1.3 DOMAIN OVERVIEW	1
	1.3.1 DEEP LEARNING	2
2	LITERATURE SURVEY	13
3	SYSTEM ANALYSIS	18
	3.1 EXISTING SYSTEM	24
	3.2 DRAWBACKS OF EXISITING SYSTEM	24
	3.3PROPOSED SYSTEM	25
	3.4 ADVANTAGES OF PROPOSED SYSTEM	25
4	REQUIREMENT SPECIFICATION	26
	4.1 INTRODUCTION	26
	4.2 HARDWARE & SOFTWARE SPECIFICATION	26
	4.2.1 HARDWARE SPECIFICATION	26
	4.2.2 SOFTWARE SPECIFICATION	26
	4.3 SOFTWARE DESCRIPTION	27
	4.3.1 ABOUT PYTHON	27
	4.3.2 ABOUT IDE	38
5	SYSTEM DESIGN	43
	5.1 INTRODUCTION	43
	5.2 ARCHITECTURE DIAGRAM	43
6	SYSTEM IMPLEMENTATION	45
	6.1 LIST OF MODULES	45
7	SOFTWARE TESTING	48
	7.1 GENERAL	53
	7.2 TEST DRIVEN DEVELOPMENT	54
	7.3 UNIT TESTING	55

	7.4 BLACKBOX TESTING	56
	7.5 INTEGRATION TESTING	57
	7.6 SYSTEM TESTING	58
	7.7 SANITY TESTING	60
	7.8 REGRESSION TESTING	62
8	CONCLUSION & FEATURE ENCHANEMENT	67
	8.1 CONCLUSION	66
	8.2 FUTURE ENCHANCEMENT	66
	APPENDIX CODE	67
	APPENDIX SCREENSHOTS	83

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
1.3.1	DEEPLARNING ARCHITECTURE	8
1.3.2	CLASSIFICATION OF NEURAL NETWORKS	10
3.1.1	WATERFALL MODEL ARCHITECTURE	19
3.1.6	RAD MODEL	21
4.3.1	PYTHON CROSS PLATFORM	27
4.3.1	PYTHON LIBRARIES	28
5.2.1	ARCHITECTURE DIAGRAM	44
7.0.1	UNIT TESTING IN SDLC	52
7.1.1	GENERAL TESTING	54
7.5.1	INTEGRATION TESTING	58
7.6.1	SYSTEM TESTING	59
7.8.1	REGRESSION TESTING	64

CHAPTER 1

INTRODUCTION

Due to the rapid development of CNNs, there has been a huge breakthrough in large-scale image classification tasks. After UI components are identified, they need to be classified, such as button, textview, progress bar, etc. This is actually an image classification task. Relying on the rapid development of deep Convolution Neural Network, we use CNN to classify the image of UI components into specific types.

1.1 Overview

We segment the binary image, cut the image into blocks, detect whether the block contains UI components, and mark the components. We use the Flood Fill algorithm to mark the location of layout blocks in the image. The FloodFill algorithm is widely used, among which the "reverse color" of drawing is prominent. The principle is to start with a point and fill the nearby pixels with a new color until all pixels in the closed area are filled with the new color. In the image, it is likely that the layout blocks are nested or overlapped.

1.2 Scope of the project

The main contributions of this project therefore are:

- ▶ Data Analysis
- ▶ Dataset Preprocessing
- ▶ Training the Model
- ▶ Testing of Dataset

1.3 DOMAIN OVERVIEW

Deep learning has evolved hand-in-hand with the digital era, which has brought about an explosion of data in all forms and from every region of the world. This data, known simply as big data, is drawn from sources like social media, internet search engines, e-commerce platforms, and online cinemas, among others. This enormous amount of

data is readily accessible and can be shared through fintech applications like cloud computing.

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

However, the data, which normally is unstructured, is so vast that it could take decades for humans to comprehend it and extract relevant information. Companies realize the incredible potential that can result from unraveling this wealth of information and are increasingly adapting to AI systems for automated support.

1.3.1 DeepLearning

About Pandas

Pandas is a popular Python package for data science, and with good reason: it offers powerful, expressive and flexible data structures that make data manipulation and analysis easy, among many other things. The DataFrame is one of these structures.

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.

Pandas is built on top of the NumPy package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis

in SciPy, plotting functions from Matplotlib, and machine learning algorithms in Scikit-learn.

Jupyter Notebooks offer a good environment for using pandas to do data exploration and modeling, but pandas can also be used in text editors just as easily.

Jupyter Notebooks give us the ability to execute code in a particular cell as opposed to running the entire file. This saves a lot of time when working with large datasets and complex transformations. Notebooks also provide an easy way to visualize pandas' DataFrames and plots. As a matter of fact, this article was created entirely in a Jupyter Notebook.

There are two types of data structures in pandas: Series and DataFrames.

1. Series: a pandas Series is a one dimensional data structure (“a one dimensional ndarray”) that can store values — and for every value it holds a unique index, too.
2. DataFrame: a pandas DataFrame is a two (or more) dimensional data structure – basically a table with rows and columns. The columns have names and the rows have indexes.

Those who are familiar with R know the data frame as a way to store data in rectangular grids that can easily be overviewed. Each row of these grids corresponds to measurements or values of an instance, while each column is a vector containing data for a specific variable. This means that a data frame's rows do not need to contain, but can contain, the same type of values: they can be numeric, character, logical, etc.

Now, DataFrames in Python are very similar: they come with the Pandas library, and they are defined as two-dimensional labeled data structures with columns of potentially different types.

In general, you could say that the Pandas DataFrame consists of three main components: the data, the index, and the columns.

Firstly, the DataFrame can contain data that is:

- a Pandas DataFrame

- a Pandas Series: a one-dimensional labeled array capable of holding any data type with axis labels or index. An example of a Series object is one column from a DataFrame.
- a NumPy ndarray, which can be a record or structured
- a two-dimensional ndarray
- dictionaries of one-dimensional ndarray's, lists, dictionaries or Series.

Some of the key features of Python Pandas are as follows:

It provides DataFrame objects with default and customized indexing which is very fast and efficient.

- There are tools available for loading data of different file formats into in-memory data objects.
- It is easy to perform data alignment and integrated handling of missing data in Python Pandas.
- It is very simple to perform pivoting and reshaping of data sets in Pandas.
- It also provides indexing, label-based slicing, and sub-setting of large data sets.
- We can easily insert and delete columns from a data structure.
- Data aggregation and transformations can be done using group by.
- High-performance merging and joining of data can be done using Pandas.
- It also provides time series functionality.
- Inserting and deleting columns in data structures.
- Merging and joining data sets.
- Reshaping and pivoting data sets.
- Aligning data and dealing with missing data.
- Manipulating data using integrated indexing for DataFrame objects.
- Performing split-apply-combine on data sets using the group by engine.
- Manipulating high-dimensional data in a data structure with a lower dimension using hierarchical axis indexing.
- Subsetting, fancy indexing, and label-based slicing data sets that are large in size.

- Generating data range, converting frequency, date shifting, lagging, and other time-series functionality.
- Reading from files with CSV, XLSX, TXT, among other formats.
- Arranging data in an order ascending or descending.
- Filtering data around a condition.
- Analyzing time series.
- Iterating over a data set.

Numpy

Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays. If you are already familiar with MATLAB, you might find this tutorial useful to get started with Numpy.

A numpy array is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the rank of the array; the shape of an array is a tuple of integers giving the size of the array along each dimension.

NumPy is, just like SciPy, Scikit-Learn, Pandas, etc. one of the packages that you just can't miss when you're learning data science, mainly because this library provides you with an array data structure that holds some benefits over Python lists, such as: being more compact, faster access in reading and writing items, being more convenient and more efficient.

NumPy is a Python library that is the core library for scientific computing in Python. It contains a collection of tools and techniques that can be used to solve on a computer mathematical models of problems in Science and Engineering. One of these tools is a high-performance multidimensional array object that is a powerful data structure for efficient computation of arrays and matrices. To work with these arrays, there's a vast amount of high-level mathematical functions operate on these matrices and arrays.

an array is basically nothing but pointers. It's a combination of a memory address, a data type, a shape, and strides:

- The data pointer indicates the memory address of the first byte in the array,
- The data type or dtype pointer describes the kind of elements that are contained within the array,
- The shape indicates the shape of the array, and
- The strides are the number of bytes that should be skipped in memory to go to the next element. If your strides are (10,1), you need to proceed one byte to get to the next column and 10 bytes to locate the next row.

Or, in other words, an array contains information about the raw data, how to locate an element and how to interpret an element.

With NumPy, we work with multidimensional arrays. We'll dive into all of the possible types of multidimensional arrays later on, but for now, we'll focus on 2-dimensional arrays. A 2-dimensional array is also known as a matrix, and is something you should be familiar with. In fact, it's just a different way of thinking about a list of lists. A matrix has rows and columns. By specifying a row number and a column number, we're able to extract an element from a matrix.

We can create a NumPy array using the `numpy.array` function. If we pass in a list of lists, it will automatically create a NumPy array with the same number of rows and columns. Because we want all of the elements in the array to be float elements for easy computation, we'll leave off the header row, which contains strings. One of the limitations of NumPy is that all the elements in an array have to be of the same type, so if we include the header row, all the elements in the array will be read in as strings. Because we want to be able to do computations like find the average quality of the wines, we need the elements to all be floats.

NumPy has several advantages over using core Python mathematical functions, a few of which are outlined here:

1. NumPy is extremely fast when compared to core Python thanks to its heavy use of C extensions.
2. Many advanced Python libraries, such as Scikit-Learn, Scipy, and Keras, make extensive use of the NumPy library. Therefore, if you plan to pursue a career in

data science or machine learning, NumPy is a very good tool to master. NumPy comes with a variety of built-in functionalities, which in core Python would take a fair bit of custom code.

Matplotlib

Plotting of data can be extensively made possible in an interactive way by Matplotlib, which is a plotting library that can be demonstrated in Python scripts. Plotting of graphs is a part of data visualization, and this property can be achieved by making use of Matplotlib.

Matplotlib makes use of many general-purpose GUI toolkits, such as wxPython, Tkinter, QT, etc., in order to provide object-oriented APIs for embedding plots into applications. John D. Hunter was the person who originally wrote Matplotlib, and its lead developer was Michael Droettboom. One of the free and open-source Python library which is basically used for technical and scientific computing is Python SciPy. Matplotlib is widely used in SciPy as most scientific calculations require plotting of graphs and diagrams.

Matplotlib is a plotting library like GNUplot. The main advantage towards GNUplot is the fact that Matplotlib is a Python module. Due to the growing interest in python the popularity of matplotlib is continually rising as well.

Another reason for the attractiveness of Matplotlib lies in the fact that it is widely considered to be a perfect alternative to MATLAB, if it is used in combination with Numpy and Scipy. Whereas MATLAB is expensive and closed source, Matplotlib is free and open source code. It is also object-oriented and can be used in an object oriented way. Furthermore it can be used with general-purpose GUI toolkits like wxPython, Qt, and GTK+. There is also a procedural "pylab", which designed to closely resemble that of MATLAB. This can make it extremely easy for MATLAB users to migrate to matplotlib.

Matplotlib can be used to create publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

Another characteristic of matplotlib is its steep learning curve, which means that users usually make rapid progress after having started. The official website has to say the following about this: "matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc, with just a few lines of code."

Deep Learning

Deep learning is a computer software that mimics the network of neurons in a brain. It is a subset of machine learning and is called deep learning because it makes use of deep neural networks.

Deep learning algorithms are constructed with connected layers.

- The first layer is called the Input Layer
- The last layer is called the Output Layer
- All layers in between are called Hidden Layers. The word deep means the network join neurons in more than two layers.

A deep neural network provides state-of-the-art accuracy in many tasks, from object detection to speech recognition. They can learn automatically, without predefined knowledge explicitly coded by the programmers.

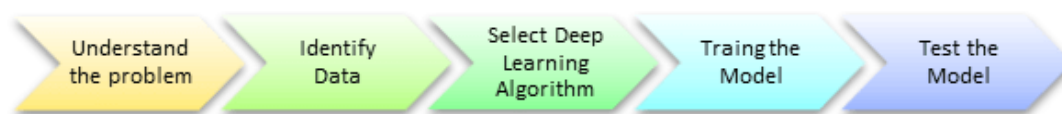


Fig:1.3.1: Deep Learning Architecture

To grasp the idea of deep learning, imagine a family, with an infant and parents. The toddler points objects with his little finger and always says the word 'cat.' As its parents are concerned about his education, they keep telling him 'Yes, that is a cat' or 'No, that is not a cat.' The infant persists in pointing objects but becomes more accurate with 'cats.' The little kid, deep down, does not know why he can say it is a cat or not. He has just learned how to hierarchies complex features coming up with a cat by looking at the pet overall and continue to focus on details such as the tails or the nose before to make up his mind.

A neural network works quite the same. Each layer represents a deeper level of knowledge, i.e., the hierarchy of knowledge. A neural network with four layers will learn more complex feature than with that with two layers.

The learning occurs in two phases.

1. The first phase consists of applying a nonlinear transformation of the input and create a statistical model as output.
2. The second phase aims at improving the model with a mathematical method known as derivative.

The neural network repeats these two phases hundreds to thousands of time until it has reached a tolerable level of accuracy. The repeat of this two-phase is called an iteration

Classification of Neural Networks

1. Shallow neural network: The Shallow neural network has only one hidden layer between the input and output.
2. Deep neural network: Deep neural networks have more than one layer. For instance, Google LeNet model for image recognition counts 22 layers.

The computational models in Deep Learning are loosely inspired by the human brain. The multiple layers of training are called Artificial Neural Networks (ANN).

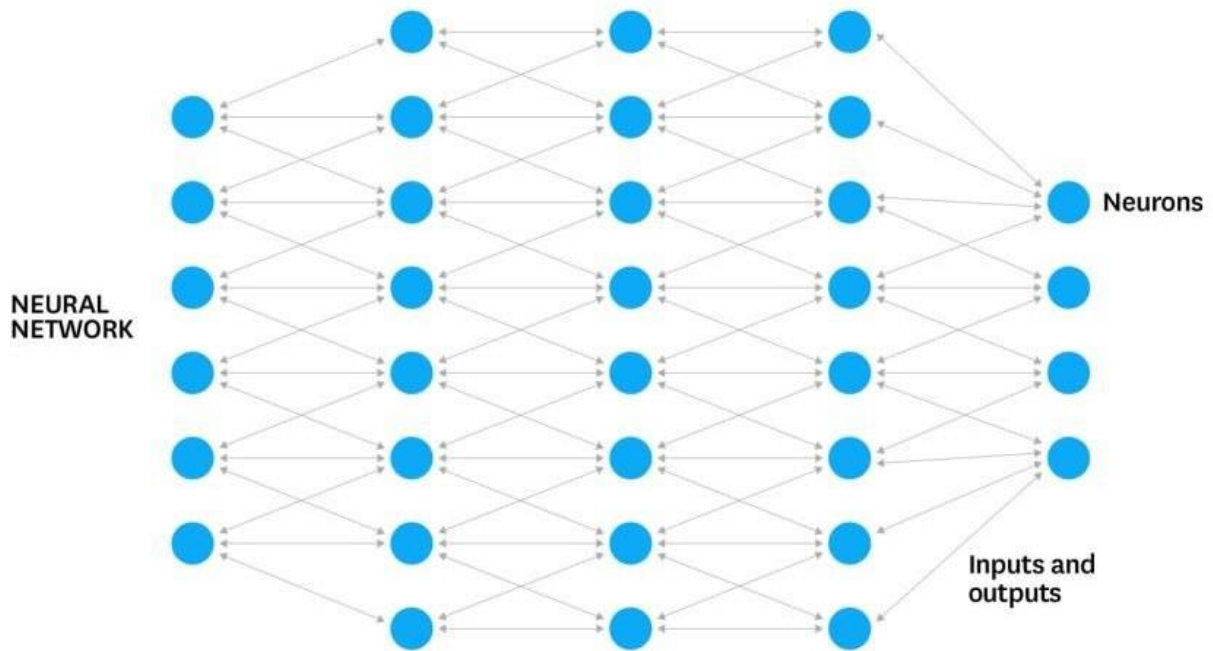


Fig:3.1.2: Classification Of Neural Networks

Neuron

Artificial Neural Networks contain layers of neurons. A neuron is a computational unit that calculates a piece of information based on weighted input parameters. Inputs accepted by the neuron are separately weighted.

Inputs are summed and passed through a non-linear function to produce output. Each layer of neurons detects some additional information, such as edges of things in a picture or tumors in a human body. Multiple layers of neurons can be used to detect additional information about input parameters.

Nodes

Artificial Neural Network is an interconnected group of nodes akin to the vast network of layers of neurons in a brain. Each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.

Inputs

Inputs are passed into the first layer. Individual neurons receive the inputs, with each of them receiving a specific value. After this, an output is produced based on these values.

Outputs

The outputs from the first layer are then passed into the second layer to be processed. This continues until the final output is produced. The assumption is that the correct output is predefined.

Each time data is passed through the network, the end result is compared with the correct one, and tweaks are made to their values until the network creates the correct final output each time.

Some of the commonly used neural networks are as follows:

1. Artificial Neural Network (ANN)
2. Convolutional Neural Network (CNN)
3. Recurrent Neural Network (RNN)
4. Deep Neural Network (DNN)
5. Deep Belief Network (DBN)

Artificial neural networks are one of the main tools used in machine learning. As the “neural” part of their name suggests, they are brain-inspired systems which are intended to replicate the way that we humans learn. Neural networks consist of input and output layers, as well as (in most cases) a hidden layer consisting of units that transform the input into something that the output layer can use. They are excellent tools for finding patterns which are far too complex or numerous for a human programmer to extract and teach the machine to recognize.

Generally, the working of a human brain by making the right connections is the idea behind ANNs. That was limited to use of silicon and wires as living neurons and dendrites.

Here, neurons, part of human brain. That was composed of 86 billion nerve cells. Also, connected to other thousands of cells by Axons. Although, there are various inputs from sensory organs. That was accepted by dendrites. As a result, it creates electric impulses. That is used to travel through the Artificial neural network. Thus, to handle the different issues, neuron send a message to another neuron.

As a result, we can say that ANNs are composed of multiple nodes. That imitate biological neurons of the human brain. Although, we connect these neurons by links.

Also, they interact with each other. Although, nodes are used to take input data. Further, perform simple operations on the data. As a result, these operations are passed to other neurons. Also, output at each node is called its activation or node value.

CHAPTER 2

LITERATURE SURVEY

User Interface Code Retrieval: A Novel Visual-Representation- Aware Approach

In this paper, Code reuse has been perceived to be an effective tool for user interface development that is known to be complex and messy. A fundamental problem in user interface code reuse is how to effectively retrieve working code from existing code repositories, which renders functional interfaces similar to a programmer's input sketch. Existing generic code retrieval techniques are normally inadequate for user interface code due to its associated visual information. In this paper, we put forward a novel approach that retrieves matching candidate code of an interface sketch based on their visual-representation trees. To better capture sub-region

similarities, we propose to decompose a visual-representation tree into an ordered list of paths (or sequences). We then devise a novel algorithm to measure the similarity between two sequences by leveraging various visual information and another algorithm to calculate the similarity of two ordered lists by considering the relative positions of different widgets. To evaluate our solution, we carefully design an experiment to generate the similarity scores of 6,750 user interface pairs based on real users' judgement. Based on this ground truth, we show that our method is able to generate accurate similarity scores, outperforming several state-of-the-art competitors.

Following the framework proposed, the VGG 16-layer net is taken as the backbone. The first 5 convolutional stages are retained while the following 2 fully-connected layers are removed. Thanks to the variable receptive field size of each convolutional stage, different stages can capture different level features with corresponding sizes. Specifically, shallower stages can acquire features of smaller objects while deeper stages are able to catch features of larger objects. Inspired by the property of the multi-stage convolutional network, a side output with upsampling to the same size as input image is added after each stage to get multi-scale output. We observe that this nested multi-scale framework is insensitive to input image sizes. With a fixed-size

input, our proposed method can detect signs of variable scales. The sideoutput prediction upsampling is simply implemented with a deconvolutional layer composed of a 1×1 convolutional layer and a upsampling layer, where the parameters of upsampling are fixed to perform bilinear interpolation. Then the hierarchical fusion maps are generated by concatenating these upsampled side outputs along the channel. Finally, a 1×1 convolutional layer is deployed on the top to aggregate the information over the channel and a sigmoid layer follows to make the pixel-level sign/non-sign prediction.

This paper consider the traffic sign detection task as a pixel-level sign/non-sign classification. So at the training stage, pixels within the bounding box of each traffic sign are treated as the positive region and vice versa. That is to say, we label the pixels within the traffic sign region to 1, and the pixels outside the traffic sign region to 0. FCN in this stage does not focus on where texts are, but where traffic signs are. Ideally, we can filter most non-sign regions in this stage, which greatly enhances the efficiency of the latter text detection. The binary cross-entropy loss function and stochastic gradient descent are used to train this part.

Combinatorial Optimization of Graphical User Interface Designs

In this paper, the graphical user interface (GUI) has become the prime means for interacting with computing systems. It leverages human perceptual and motor capabilities for elementary tasks such as command exploration and invocation, information search, and multitasking. For designing a GUI, numerous interconnected decisions must be made such that the outcome strikes a balance between human factors and technical objectives. Normally, design choices are specified manually and coded within the software by professional designers and developers. This article surveys

combinatorial optimization as a flexible and powerful tool for computational generation and adaptation of GUIs. As recently as 15 years ago, applications were limited to keyboards and widget layouts. The obstacle has been the mathematical definition of design tasks, on the one hand, and the lack of objective functions that

capture essential aspects of human behavior, on the other. This article presents definitions of layout design problems as integer programming tasks, a coherent formalism that permits identification of problem types, analysis of their complexity, and exploitation of known algorithmic solutions. It then surveys advances in formulating evaluative functions for common design-goal foci such as user performance and experience. The convergence of these two advances has expanded the range of solvable problems. Approaches to practical deployment are outlined with a wide spectrum of applications. This article concludes by discussing the position of this application area within optimization and human-computer interaction research and outlines challenges for future work.

In our work, we build the CNN block based on the SSD structure where predictions are directly performed across multiple feature levels. The main difference of our network with the previous detection networks is what it predicts as output: instead of predicting bounding box coordinates, our network performs pose estimation, which can be converted into the boundary estimation of corresponding traffic signs. In the CNN block, an input image is passed to a base network which extracts feature maps using a series of convolution, non-linear activation, and pooling operations. Then from the feature maps, 2D poses and shape class probabilities are estimated by two separated convolutional layers, namely, pose regression layer and shape classification layer, combined with successive operations that convert the convolution outputs to the 2D pose values and class probabilities, respectively (through Softmax and Decoder in Figure 2). Finally, we use the obtained 2D poses and shape class probabilities to compute boundary corners.

Given the 2D pose of a target sign, it is straightforward to determine the boundary of the sign given its template image. First, from the shape classification layer we predict the shape label of the target to.

Towards Real-Time Traffic Sign Detection and Classification

This paper aims to deal with real-time traffic sign recognition, i.e., localizing what type of traffic sign appears in which area of an input image at a fast processing time. To

achieve this goal, we first propose an extremely fast detection module, which is 20 times faster than the existing best detection module. Our detection module is based on traffic sign proposal extraction and classification built upon a color probability model and a color HOG. Then, we harvest from a convolutional neural network to further classify the detected signs into their subclasses within each superclass. Experimental results on both German and Chinese roads show that both our detection and classification methods achieve comparable performance with the state-of-the-art methods, with significantly improved computational efficiency

We propose a color probability model to deal with color information of traffic signs, so as to enhance the specific colors (e.g., red, blue and yellow) of traffic signs and suppress background colors as well as to reduce the search space of following algorithms and detection time. To further reduce detection time, we propose to extract traffic sign proposals instead of sliding window detection and combine some machine learning algorithms, i.e., SVM and CNN, to detect and classify traffic signs. To evaluate our methods on Chinese roads, we construct a Chinese Traffic Sign Dataset (CTSD) and make it available online for public use.¹

The goal of traffic sign classification is to classify the detected traffic signs into their specific sub-classes. Convolutional neural network is a good method for traffic sign classification. It is proved that the performance of CNN on traffic sign classification even outperforms the human performance. In [15], a CNN combined with a Multi-Layer Perception (MLP) trained on HOG features takes the 1st place of the preliminary phase of GTSRB competition. In [16], a multi-scale features CNN is presented for traffic sign classification by using layerskipping connection. Both of them outperform the human performance in the preliminary phase of GTSRB competition.

Fast Detection of Multiple Objects in Traffic Scenes With a Common Detection Framework

This paper e propose to detect all the three important objects in a single learning-based detection framework. The proposed framework consists of a dense feature extractor and detectors of three important classes. Once the dense features have been extracted, these features are shared with all detectors. The advantage of using one common framework is that the detection speed is much faster, since all dense features need only to be evaluated once in the testing phase. In contrast, most previous works have designed specific detectors using different features for each of these three classes. To enhance the feature robustness to noises and image deformations, we introduce spatially pooled features as a part of aggregated channel features. In order to further improve the generalization performance, we propose an object subcategorization method as a means of capturing the intraclass variation of objects. We experimentally demonstrate the effectiveness and efficiency of the proposed framework in three detection applications: traffic sign detection, car detection, and cyclist detection.

For object classes with a large intra-class variation like cars, the appearances and shapes of cars change significantly as viewpoints change. In order to deal with these variations that cannot be tackled by the conventional VJ framework, we present an object

subcategorization method which aims to cluster the object class into.

CHAPTER 3

SYSTEM ANALYSIS

One of the primary choices we face for our project implementations is “Which development methodology should we use?” There are various software development methodologies available in the ICT industry. They are

- Agile Software Development
- Crystal Methods
- Dynamic Systems Development Model (DSDM)
- Extreme Programming (XP)
- Feature Driven Development (FDD)
- Joint Application Development (JAD)
- Lean Development (LD)
- Rapid Application Development (RAD)
- Rational Unified Process (RUP)
- Scrum
- Spiral
- Systems Development Life Cycle (SDLC)
- Waterfall (a.k.a. Traditional)

Among all these methodologies, we have decided to compare the two most popular and widely used methodologies:

- Waterfall Model
- RAD Model

Waterfall Model

It is called as traditional approach. Waterfall is a linear method (sequential model) for any software application development. Here application development is segregated into a sequence of pre-defined phases.

Waterfall Model (Taken from Google.com)

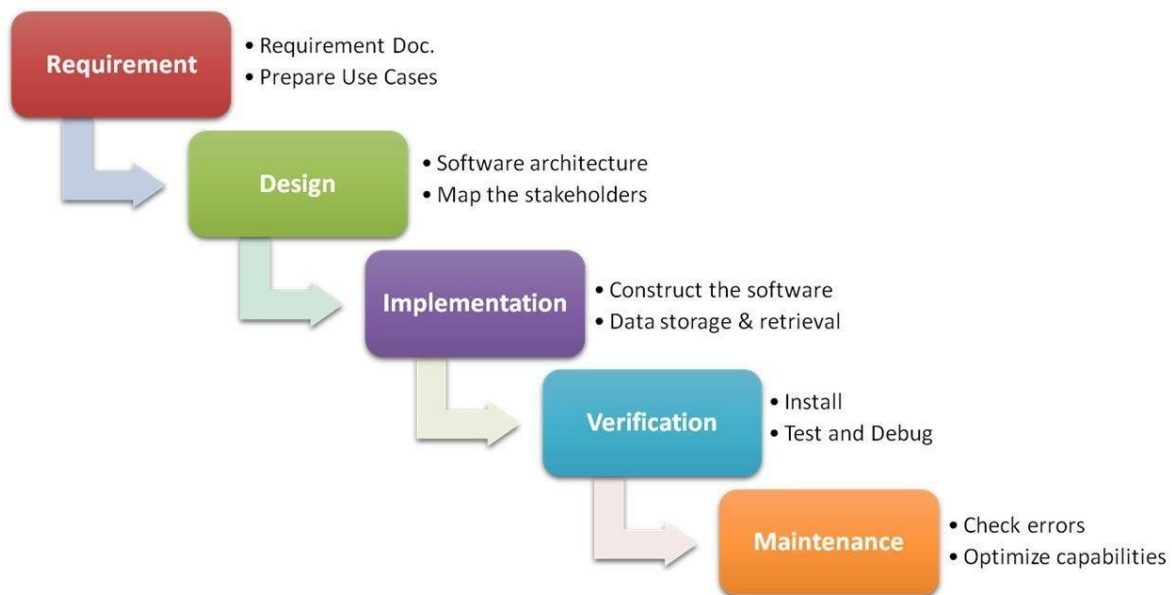


Fig:3.1.1: Waterfall Model Architecture

(Reference : <https://www.lucidchart.com/blog/waterfall-project-management-methodology>)

Requirement gathering and documentation

In this stage, you should gather comprehensive data approximately what this challenge requires. You may gather this data in a diffusion of ways, from interviews to questionnaires to interactive brainstorming. By means of the stop of this section, the task requirements have to be clean, and also you have to have a necessities file that has been allotted on your team.

2. System design

Using the well-known requirements, your team designs the solutions. During this phase, no development will be happening. But the project team starts specification such

as programming language or hardware requirements.Implementation

During this phase software development coding will be happening. Web application programmers take data from the previous stage and create a functional product. Web application programmers write source code in small pieces, which are integrated at the end of this phase or the beginning of the next.

3. Testing

Once all coding is done, testing of the product can begin. Testers methodically find and report any problems. If serious issues arise, your project may need to return to phase one for reevaluation.

4. Delivery/deployment

In this phase, the solution is complete, and your project team submits the deliverables to be deployed or released.

5. Maintenance

The final solution has been implemented to the client and is being used. As troubles arise, the project team might also want to create patches and updates might also to deal with them. Again, huge troubles may also necessitate a return to segment one.

RAD Model

Agile is an iterative, team-based approach to development. This method highlights the speedy delivery of an software application in comprehensive functional components.

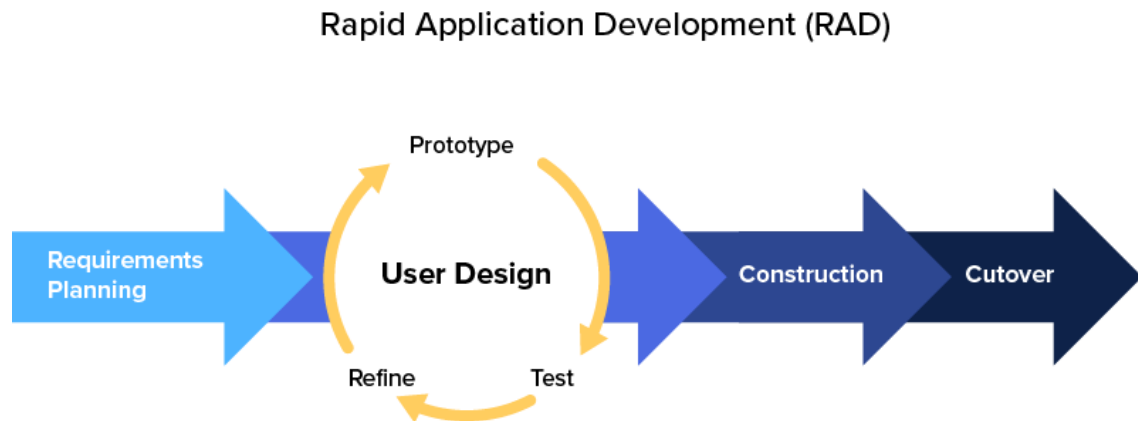


Fig:3.1.6:RAD Model

RAD Model (Taken from Google.com)

References (<https://www.lucidchart.com/blog/rapid-application-development-methodology>)

Phase 1: Requirements planning

This phase is equivalent to a project scoping meeting. Although the planning phase is condensed compared to other project management methodologies, this is a critical step for the ultimate success of the project.

During this stage, web application programmers, clients (software users), and team members communicate to determine the goals and expectations for the project as well as current and potential issues that would need to be addressed during the build.

A basic breakdown of this stage involves:

Researching the current problem

Defining the requirements for the project

Finalizing the requirements with each stakeholder's approval

It is important that everyone has the opportunity to evaluate the goals and expectations for the project and weigh in. By getting approval from each key stakeholder and web application programmer, teams can avoid miscommunications and costly change orders down the road.

Phase 2: User design

Once the project is scoped out, it's time to jump right into development, building out the user design through various prototype iterations.

This is the meat and potatoes of the RAD methodology—and what sets it apart from other project management strategies. During this phase, clients work hand in hand with web application programmers to ensure their needs are being met at every step in the design process. It's almost like customizable software development where the users can test each prototype of the product, at each stage, to ensure it meets their expectations.

All the bugs and kinks are worked out in an iterative process. The web application programmer designs a prototype, the client (user) tests it, and then they come together to communicate on what worked and what didn't.

This method gives web application programmers the opportunity to tweak the model as they go until they reach a satisfactory design.

Both the software web application programmers and the clients learn from the experience to make sure there is no potential for something to slip through the cracks.

Phase 3: Rapid construction

Phase 3 takes the prototypes and beta systems from the design phase and converts them into the working model.

Because the majority of the problems and changes were addressed during the thorough iterative design phase, web application programmers can construct the final

working model more quickly than they could by following a traditional project management approach.

The phase breaks down into several smaller steps:

Preparation for rapid construction

Program and application development

Coding

Unit, integration, and system testing

The software development team of programmers, coders, testers, and web application programmers work together during this stage to make sure everything is working smoothly and that the end result satisfies the client's expectations and objectives.

This third phase is important because the client still gets to give input throughout the process. They can suggest alterations, changes, or even new ideas that can solve problems as they arise.

Phase 4: Cutover

This is the implementation phase where the finished product goes to launch. It includes data conversion, testing, and changeover to the new system, as well as user training.

All final changes are made while the coders and clients continue to look for bugs in the system.

Benefits of RAD methodology

RAD is one of the most successful software development programs available today, with numerous benefits for both software development teams as well as their clients.

Here are just a few advantages:

- ▶ RAD lets you break the project down into smaller, more manageable tasks.
- ▶ The task-oriented structure allows project managers to optimize their team's efficiency by assigning tasks according to members' specialties and experience.
- ▶ Clients get a working product delivered in a shorter time frame.

- ▶ Regular communication and constant feedback between team members and stakeholders increases the efficiency of the design and build process.

3.1 EXISTING SYSTEM

In the existing system, the process takes place both at an application's inception and in an evolutionary context as GUI changes keep pace with evolving features. Unfortunately, this practice is challenging and time-consuming. In this paper, we present an approach that automates this process by enabling accurate prototyping of GUIs via three tasks: detection, classification, and assembly. First, logical components of a GUI are detected from a mock-up artifact using either computer vision techniques or mock-up metadata. Then, software repository mining, automated dynamic analysis, and deep convolutional neural networks are utilized to accurately classify GUI-components into domain-specific types (e.g., toggle-button). Finally, a data-driven, K-nearest-neighbors algorithm generates a suitable hierarchical GUI structure from which a prototype application can be automatically assembled.

In modern GUI-centric apps, GUI-components are rarely rendered on the screen using pre- defined coordinates. Instead, logical groupings of containers form hierarchical structures (or GUI-hierarchies). These hierarchies typically define spatial information about their constituent components, and in many cases react to changes in the size of the display area (i.e., reactive design) . For instance, a GUIcomponent that displays text may span the text according to the dimensions of its container.

3.2 DRAWBACKS OF EXISTING SYSTEM

- ✓ Takes lot of time to train the model.
- ✓ The class membership is not taken into consideration
- ✓ It takes mean value; therefore it is less efficient
- ✓ It encounters the problem of overfitting
- ✓ The structure is complex; therefore it is difficult to understand.

3.3 PROPOSED SYSTEM

In the proposed application, After obtaining the screenshot, we need to process the image to get the suspected UI component image. The image obtained by the screen capture tool is a color image, and we need to convert the color image into gray image. When mean filter is used to reduce image noise, it will bring side effects of image blur. Therefore, we take the gradient of the gray image to make the gray change of the contour edge more obvious.

We segment the binary image, cut the image into blocks, detect whether the block contains UI components, and mark the components. We use the Flood Fill algorithm to mark the location of layout blocks in the image. The algorithm is widely used, among which the "reverse color" of drawing is prominent. The principle is to start with a point and fill the nearby pixels with a new color until all pixels in the closed area are filled with the new color. In the image, it is likely that the layout blocks are nested or overlapped. Therefore, after detecting the layout block, we also need to calculate the level of the block to label it. For the calculation of the block level, we calculate the level relationship by comparing the relative sizes of two blocks.

We detect whether the divided blocks contain components. First, we check whether the block itself is a UI component based on its relative size. If it meets our standards, it is considered a UI component. Then, the binary image is processed and cut according to the relative size of the block to obtain the area containing the UI component in the block. Note that in this step of processing, due to the nesting relationship between the blocks, it is necessary to delete the blocks containing sub blocks before cutting.

3.4 Advantages of Proposed System.

- ✓ The speed of the classifier is high
- ✓ High Computation Rate
- ✓ Deals with noisy inputs efficiently
- ✓ It is data driven; therefore it learns from training data
- ✓ Does not face the problem of overfitting

CHAPTER 4

REQUIREMENT SPECIFICATION

4.1 INTRODUCTION

Clearly defined requirements are essential signs on the road that leads to a successful project. They establish a formal agreement between a client and a provider that they are both working to reach the same goal. High-quality, detailed requirements also help mitigate financial risks and keep the project on a schedule. According to the Business Analysis Body of Knowledge definition, requirements are a usable representation of a need.

Creating requirements is a complex task as it includes a set of processes such as elicitation, analysis, specification, validation, and management. In this article, we'll discuss the main types of requirements for software products and provide a number of recommendations for their use.

4.2 HARDWARE & SOFTWARE SPECIFICATION

4.2.1 HARDWARE SPECIFICATION

Hardware Requirement

Hardware	Minimum Requirement
----------	---------------------

Disk Space	32 GB or more, 10 GB or more for Foundation Edition
------------	---

Processor	1.4 GHz 64 bit
-----------	----------------

Memory	512 MB
--------	--------

Display	(800 × 600) Capable video adapter and monitor
---------	---

4.2.2 SOFTWARE SPECIFICATION

Backend Technologies

- ✓ Python
- ✓ Numpy
- ✓ Sci-learn

- ✓ Eclipse IDE

Frontend Technologies

- ✓ Web Technologies
- ✓ Bootstrap

4.3 SOFTWARE DESCRIPTION

4.3.1 ABOUT PYTHON

Python is a free, open-source programming language. Therefore, all you have to do is install Python once, and you can start working with it. Not to mention that you can contribute your own code to the community. Python is also a cross-platform compatible language. So, what does this mean? Well, you can install and run Python on several operating systems. Whether you have a Windows, Mac or Linux, you can rest assure that Python will work on all these operating systems.

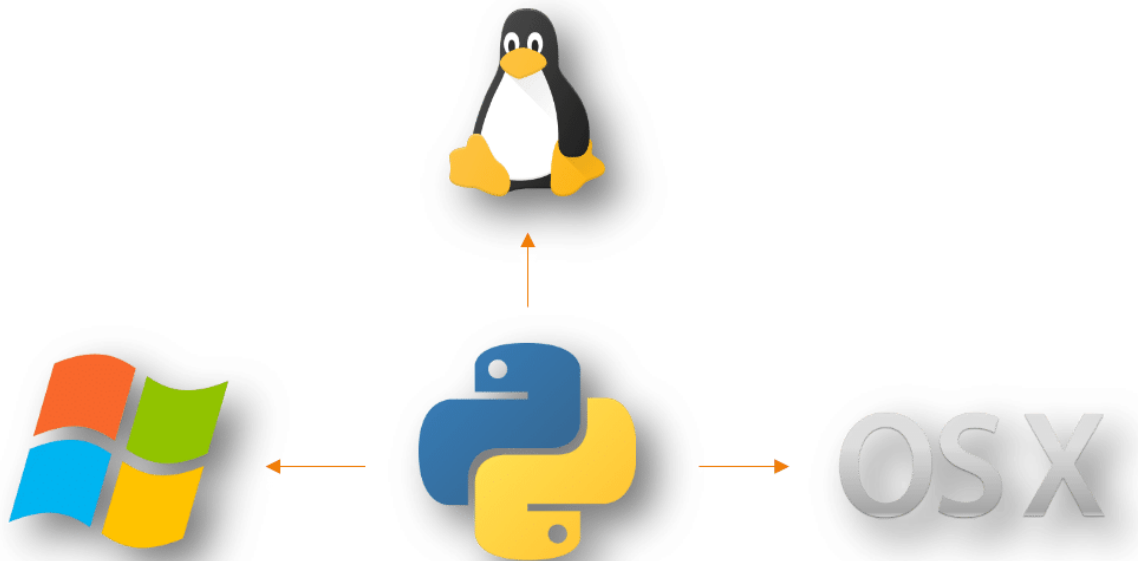


Fig: 4.3.1: Python Cross platform

Python is also a great visualization tool. It provides libraries such as Matplotlib, seaborn and bokeh to create stunning visualizations.

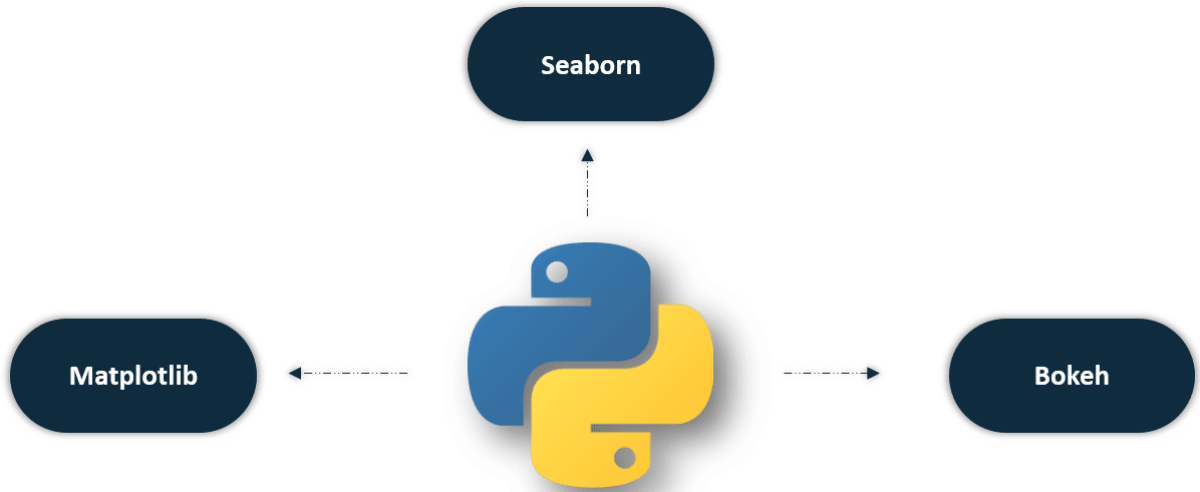


Fig: 4.3.1: Python Libraries

In addition, Python is the most popular language for machine learning and deep learning. As a matter of fact, today, all top organizations are investing in Python to implement machine learning in the back-end.

Python Line Structure

Python coding style comprises physical lines as well as logical lines or statements. A physical line in a Python program is a sequence of characters, and the end of the line terminates the line sequence as opposed to some other languages, such as C and C++ where a semi-colon is used to mark the end of the statement. A logical line, on the other hand, is composed of one or more physical lines. The use of a semi-colon is not prohibited in Python, although it's not mandatory. The NEWLINE token denotes the end of the logical line. A logical line that only contains spaces, comments, or tabs are called blank lines and they are ignored by the interpreter.

As we saw that in Python, a new line simply means that a new statement has started. Although, Python does provide a way to split a statement into a multiline statement or to join multiple statements into one logical line. This can be helpful to increase the readability of the statement. Following are the two ways to split a line into two or more lines:

Explicit Line Joining

In explicit line joining, we use a backward slash to split a statement into a multiline statement.

Implicit Line Joining

Statements that reside inside [], {}, or () parentheses can be broken down into two or more physical lines without using a back slash.

Multiple Statements on a Single Line

In Python, it is possible to club multiple statements in the same line using a semi-colon; however, most programmers do not consider this to be a good practice as it reduces the readability of the code.

Whitespaces and Indentation

Unlike most of the programming languages, Python uses indentation to mark a block of code. According to Python coding style guideline or PEP8, we should keep an indent size of four.

Most of the programming languages provide indentation for better code formatting and don't enforce to have it. But in Python it is mandatory. This is why indentation is so crucial in Python.

Comments in any programming language are used to increase the readability of the code. Similarly, in Python, when the program starts getting complicated, one of the best ways to maintain the readability of the code is to use Python comments. It is considered a good practice to include documentations and notes in the python syntax since it makes the code way more readable and understandable to other programmers as well, which comes in handy when multiple programmers are simultaneously working on the same project.

The code can only explain how it does something and not why it does that, but Python comments can do that. With Python comments, we can make documentations for various explanations in our code itself. Comments are nothing but tagged lines of codes which increase the readability of a code and make it self-explanatory. There are

different ways of creating comments depending on the type of comment we want to include in our code. Following are different kinds of comments that can be included in our Python program:

1. Single Line Comments
2. Docstring Comments
3. Multiline Comments

Single line Python comments are marked with `#` character. These comments end at the end of the physical line, which means that all characters starting after the `#` character (and lasts till the end of the line) are part of the comment.

Python has the documentation strings (or docstrings) feature which is usually the first statement included in functions and modules. Rather than being ignored by the Python Interpreter like regular comments, docstrings can actually be accessed at the run time using the dot operator.

It gives programmers an easy way of adding quick notes with every Python module, function, class, and method. To use this feature, we use triple quotes in the beginning of the documentation string or comment and the closing triple quotes at the end of the documentation comment. Docstrings can be one-liners as well as multi-liners.

Unlike some programming languages that support multiline comments, such as C, Java, and more, there is no specific feature for multiline comments in Python. But that does not mean that it is totally impossible to make multiline comments in Python. There are two ways we can include comments that can span across multiple lines in our Python code.

Python Block Comments: We can use several single line comments for a whole block. This type of comment is usually created to explain the block of code that follows the Block comment. Python Block comment is the only way of writing a real comment that can span across multiple lines. It is supported and preferred by Python's PEP8 style guide since Block comments are ignored by Python interpreter or parser. However, nothing is stopping programmers from using the second 'non-real' way of writing multiline comments in Python which is explained below.

Using Docstrings: Docstrings are largely used as multiline comments in Python by many programmers since it is the closest thing to having a multiline comment feature in Python. While it is not wrong to use docstrings when we need to make multiline comments, it is important to keep in mind that there is a significant difference between docstrings and comments. Comments in Python are totally ignored by the Python Interpreter, while docstrings, when used inside the Python function, can be accessed at the run time.

Python Data Types

One of the most crucial part of learning any programming language is to understand how data is stored and manipulated in that language. Users are often inclined toward Python because of its ease of use and the number of versatile features it provides. One of those features is dynamic typing.

In Python, unlike statically typed languages like C or Java, there is no need to specifically declare the data type of the variable. In dynamically typed languages such as Python, the interpreter itself predicts the data type of the Python Variable based on the type of value assigned to that variable.

Advantages of Python

- Universal Language Construct
- Support both High Level and Low Level Programming
- Language Interoperability
- Fastest Development life cycle therefore more productive coding environment
- Less memory used because a single container hold multiple data types and each type doesn't require its own function
- Learning Ease and open source development
- Speed and user-friendly data structure
- Extensive and extensible libraries.
- Simple & support IoT

- And many more

Based on the 3 languages that are described above, we decided to use Python as programming language for developing this e-Voting web based application. The important motives are

- Easy to learn, even non experienced programmers can use it. Ex: spacing and tabbing instead of extra syntax
- Interactive mode
- Large and comprehensive standard libraries
- Python programs resemble to that of pseudo-code. This makes it a basis and a must have for beginner programmers due to its extreme ease a difficulty when compared to C++ Java, Perl, and so forth

About Pandas

Pandas is a popular Python package for data science, and with good reason: it offers powerful, expressive and flexible data structures that make data manipulation and analysis easy, among many other things. The DataFrame is one of these structures.

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.

Pandas is built on top of the NumPy package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis in SciPy, plotting functions from Matplotlib, and machine learning algorithms in Scikit-learn.

Jupyter Notebooks offer a good environment for using pandas to do data exploration and modeling, but pandas can also be used in text editors just as easily.

Jupyter Notebooks give us the ability to execute code in a particular cell as opposed to running the entire file. This saves a lot of time when working with large datasets and complex transformations. Notebooks also provide an easy way to visualize pandas' DataFrames and plots. As a matter of fact, this article was created entirely in a Jupyter Notebook.

There are two types of data structures in pandas: Series and DataFrames.

1. **Series:** a pandas Series is a one dimensional data structure (“a one dimensional ndarray”) that can store values — and for every value it holds a unique index, too.
2. **DataFrame:** a pandas DataFrame is a two (or more) dimensional data structure – basically a table with rows and columns. The columns have names and the rows have indexes.

Those who are familiar with R know the data frame as a way to store data in rectangular grids that can easily be overviewed. Each row of these grids corresponds to measurements or values of an instance, while each column is a vector containing data for a specific variable. This means that a data frame's rows do not need to contain, but can contain, the same type of values: they can be numeric, character, logical, etc.

Now, DataFrames in Python are very similar: they come with the Pandas library, and they are defined as two-dimensional labeled data structures with columns of potentially different types.

In general, you could say that the Pandas DataFrame consists of three main components: the data, the index, and the columns.

Firstly, the DataFrame can contain data that is:

- a Pandas DataFrame
- a Pandas Series: a one-dimensional labeled array capable of holding any data type with axis labels or index. An example of a Series object is one column from a DataFrame.

- a NumPy ndarray, which can be a record or structured
- a two-dimensional ndarray
- dictionaries of one-dimensional ndarray's, lists, dictionaries or Series.

Some of the key features of Python Pandas are as follows:

It provides DataFrame objects with default and customized indexing which is very fast and efficient.

- There are tools available for loading data of different file formats into in-memory data objects.
- It is easy to perform data alignment and integrated handling of missing data in Python Pandas.
- It is very simple to perform pivoting and reshaping of data sets in Pandas.
- It also provides indexing, label-based slicing, and sub-setting of large data sets.
- We can easily insert and delete columns from a data structure.
- Data aggregation and transformations can be done using group by.
- High-performance merging and joining of data can be done using Pandas.
- It also provides time series functionality.
- Inserting and deleting columns in data structures.
- Merging and joining data sets.
- Reshaping and pivoting data sets.
- Aligning data and dealing with missing data.
- Manipulating data using integrated indexing for DataFrame objects.

- Performing split-apply-combine on data sets using the group by engine.
- Manipulating high-dimensional data in a data structure with a lower dimension using hierarchical axis indexing.
- Subsetting, fancy indexing, and label-based slicing data sets that are large in size.
- Generating data range, converting frequency, date shifting, lagging, and other time-series functionality.
- Reading from files with CSV, XLSX, TXT, among other formats.
- Arranging data in an order ascending or descending.
- Filtering data around a condition.
- Analyzing time series.
- Iterating over a data set.

Numpy

Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays. If you are already familiar with MATLAB, you might find this tutorial useful to get started with Numpy.

A numpy array is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the rank of the array; the shape of an array is a tuple of integers giving the size of the array along each dimension.

NumPy is, just like SciPy, Scikit-Learn, Pandas, etc. one of the packages that you just can't miss when you're learning data science, mainly because this library provides you with an array data structure that holds some benefits over Python lists, such as: being

more compact, faster access in reading and writing items, being more convenient and more efficient.

NumPy is a Python library that is the core library for scientific computing in Python. It contains a collection of tools and techniques that can be used to solve on a computer mathematical models of problems in Science and Engineering. One of these tools is a high-performance multidimensional array object that is a powerful data structure for efficient computation of arrays and matrices. To work with these arrays, there's a vast amount of high-level mathematical functions operate on these matrices and arrays.

an array is basically nothing but pointers. It's a combination of a memory address, a data type, a shape, and strides:

- The data pointer indicates the memory address of the first byte in the array,
- The data type or dtype pointer describes the kind of elements that are contained within the array,
- The shape indicates the shape of the array, and
- The strides are the number of bytes that should be skipped in memory to go to the next element. If your strides are (10,1), you need to proceed one byte to get to the next column and 10 bytes to locate the next row.

Or, in other words, an array contains information about the raw data, how to locate an element and how to interpret an element.

With NumPy, we work with multidimensional arrays. We'll dive into all of the possible types of multidimensional arrays later on, but for now, we'll focus on 2-dimensional arrays. A 2-dimensional array is also known as a matrix, and is something you should be familiar with. In fact, it's just a different way of thinking about a list of lists. A matrix has rows and columns. By specifying a row number and a column number, we're able to extract an element from a matrix.

We can create a NumPy array using the `numpy.array` function. If we pass in a list of lists, it will automatically create a NumPy array with the same number of rows and columns. Because we want all of the elements in the array to be float elements for easy computation, we'll leave off the header row, which contains strings. One of the limitations of NumPy is that all the elements in an array have to be of the same type, so if we include the header row, all the elements in the array will be read in as strings. Because we want to be able to do computations like find the average quality of the wines, we need the elements to all be floats.

NumPy has several advantages over using core Python mathematical functions, a few of which are outlined here:

1. NumPy is extremely fast when compared to core Python thanks to its heavy use of C extensions.
2. Many advanced Python libraries, such as Scikit-Learn, Scipy, and Keras, make extensive use of the NumPy library. Therefore, if you plan to pursue a career in data science or machine learning, NumPy is a very good tool to master.

NumPy comes with a variety of built-in functionalities, which in core Python would take a fair bit of custom code.

Matplotlib

Plotting of data can be extensively made possible in an interactive way by Matplotlib, which is a plotting library that can be demonstrated in Python scripts. Plotting of graphs is a part of data visualization, and this property can be achieved by making use of Matplotlib.

Matplotlib makes use of many general-purpose GUI toolkits, such as wxPython, Tkinter, QT, etc., in order to provide object-oriented APIs for embedding plots into applications. John D. Hunter was the person who originally wrote Matplotlib, and its lead developer was Michael Droettboom. One of the free and open-source Python library which is basically used for technical and scientific computing is Python SciPy. Matplotlib is

widely used in SciPy as most scientific calculations require plotting of graphs and diagrams.

Matplotlib is a plotting library like GNUplot. The main advantage towards GNUplot is the fact that Matplotlib is a Python module. Due to the growing interest in python the popularity of matplotlib is continually rising as well.

Another reason for the attractiveness of Matplotlib lies in the fact that it is widely considered to be a perfect alternative to MATLAB, if it is used in combination with Numpy and Scipy. Whereas MATLAB is expensive and closed source, Matplotlib is free and open source code. It is also object-oriented and can be used in an object oriented way. Furthermore it can be used with general-purpose GUI toolkits like wxPython, Qt, and GTK+. There is also a procedural "pylab", which designed to closely resemble that of MATLAB. This can make it extremely easy for MATLAB users to migrate to matplotlib.

Matplotlib can be used to create publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

Another characteristic of matplotlib is its steep learning curve, which means that users usually make rapid progress after having started. The official website has to say the following about this: "matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc, with just a few lines of code."

4.3.2 ABOUT IDE

IDE – Integrated Development Environment. It is software suite. It contains different instruments for web application software engineers to compose and test applications.

Programming Web application software engineers utilize wide scope of devices habitually all through application source code creation, building and application testing. Programming Development apparatuses frequently incorporate word processors for source code creation, code libraries for reuse, compilers and test stages.

An Integrated Development Environment (IDE) is a product application that gives a programming domain to streamline creating and troubleshooting programming. Instead of playing out every one of the means required to make an executable program as disconnected individual assignments, it brings every one of the devices required into one application and workspace. Every one of the devices has a familiarity with the earth, and they cooperate to display a consistent improvement set for the designer.

An IDE typically has an editor and a compiler window. There, we can write our code, run it, and compile it. Hence, it becomes much easier to develop applications using an IDE. The major reasons for using an IDE are given below:

- ▶ We can easily write our code in its text editor window.
- ▶ To store the resource files of a project, we have a project editor window.
- ▶ It makes debugging of the code easier.
- ▶ An IDE's autosuggest feature helps us in writing efficient code.
- ▶ We can import other projects for integrating with the existing one.

Without an IDE, a product web application software engineer must choose, send, incorporate and deal with these instruments independently. An IDE brings huge numbers of those improvement related instruments all together system, application or administration. The incorporated toolset is intended to streamline programming advancement and can distinguish and limit coding missteps and grammatical errors.

- ✓ Komodo
- ✓ NetBeans
- ✓ PyCharm
- ✓ Python for Visual Studio.NET
- ✓ PyDev
- ✓ PyStudio
- ✓ LiClipse (Eclipse)

LiClipse is at the core for all Python based application development activities. It's been available for many years. LiClipse is a set of plugins to enhance Eclipse. It improves the overall Eclipse experience. It is best IDE tool to develop Python Applications.

Advantages of LiClipse

- Perfect Debugger
- Error checking features
- An enhanced ability to analyze the code for quality and security concerns
- Auto complete feature

Based on the various IDE that are described above, we decided to use LiClipse as IDE (Integrated Development Environment) for developing this e-Voting web based application.

Important Features That Make LiClipse our IDE of Choice

- Rich Reusable Libraries are available
- Model Driven Development
- Enterprise Python Tooling
- Powerful Debugging options

IDE is a program that is used to write applications (other programs) in a variety of programming languages. It allows you to complete many distinct program development steps in one place (program window). A program which allows you to do this is known as an integrated development environment, or IDE. The program is free and easy to import onto your home computer.

LiClipse is used for writing programs in Python. There are several steps that must be completed to set up LiClipse and further steps to start up a new Python project (set of Python components which work together to make up an application).

PyCharm is an Integrated Development Environment (IDE) that provides tools for writing efficient code. It supports various frameworks such as Django, Flask, and Pyramid. PyCharm is majorly used by developers for application development with Python language.

Let's suppose, we are making a project using Python programming language. We know that Python is extensively popular among the developer's community, as well as in industries, due to its following features:

- ▶ Simple to code: Python is a very simple language compared to other programming languages. Also, it is a developer-friendly language.
- ▶ Object-oriented language: Python is an object-oriented language that uses concepts of classes and objects.
- ▶ GUI programming support: We can create graphical user interfaces using Python Tkinter library. Besides, we can use modules such as PyQt5, PyQt4, and wxPython to create a GUI application.
- ▶ High-level language: Python is a high-level programming language. We do not need to take care of managing memory or the system architecture while writing Python codes.
- ▶ Portability: Python is a portable programming language, i.e., we can run Python codes on any platform. For example, we can run a code written for Windows in Linux, UNIX, and Mac, without changing it.
- ▶ Integrated language: Python is an integrated programming language. We can easily integrate C, C++, etc. with Python.
- ▶ Interpreted language: Python is an interpreted programming language as well. Python's source code is changed to bytecode. Due to this, it is directly executed without compiling.
- ▶ Large set of standard libraries: Python consists of a rich set of modules, packages, and libraries. It helps in various stages of application development.

"notebook" or "notebook documents" denote documents that contain both code and rich text elements, such as figures, links, equations, ... Because of the mix of code and text elements, these documents are the ideal place to bring together an analysis

description, and its results, as well as, they can be executed perform the data analysis in real time.

The Jupyter Notebook App produces these documents.

As a server-client application, the Jupyter Notebook App allows you to edit and run your notebooks via a web browser. The application can be executed on a PC without Internet access, or it can be installed on a remote server, where you can access it through the Internet.

Its two main components are the kernels and a dashboard.

A kernel is a program that runs and introspects the user's code. The Jupyter Notebook App has a kernel for Python code, but there are also kernels available for other programming languages.

The dashboard of the application not only shows you the notebook documents that you have made and can reopen but can also be used to manage the kernels: you can which ones are running and shut them down if necessary.

CHAPTER 5

SYSTEM DESIGN

5.1 INTRODUCTION

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

Mainly, UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modeling workflows as well as a wider range of features to improve readability and efficiency.

5. 2 ARCHITECTURE DIAGRAM

An architecture diagram is a graphical representation of a set of concepts, that are part of an architecture, including their principles, elements and components.

What is a diagram? What are the types of diagrams for architecture? The Dragon1 open EA Method makes it very clear: if a diagram does not show a concept, principle or part of a principle, it is NOT an architecture diagram, because it does not show (a part of) the architecture.

There are many kinds of architecture diagrams, like a software architecture diagram, system architecture diagram, application architecture diagram, security architecture diagram, etc.

For system developers, they need system architecture diagrams to understand, clarify, and communicate ideas about the system structure and the user requirements that the

system must support. It's a basic framework can be used at the system planning phase helping partners understand the architecture, discuss changes, and communicate intentions clearly.

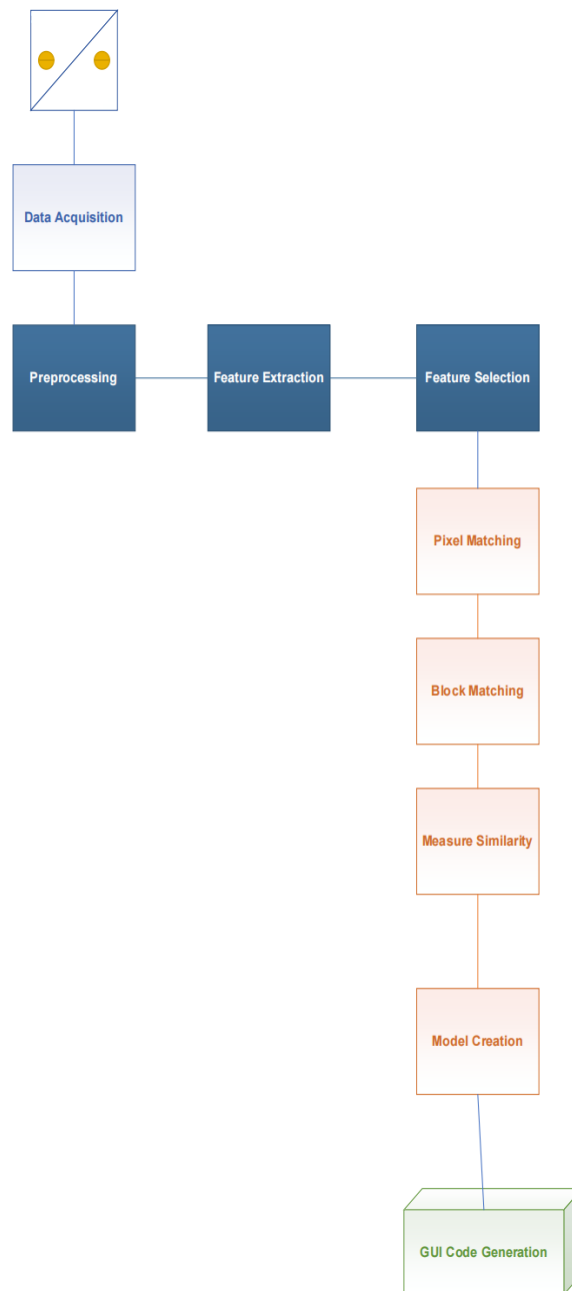


Fig: 5.2.1: Architecture Diagram

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 LIST OF MODULES

Image Preprocessing

Image pre-processing is the term for operations on images at the lowest level of abstraction. These operations do not increase image information content but they decrease it if entropy is an information measure. The aim of pre-processing is an improvement of the image data that suppresses undesired distortions or enhances some image features relevant for further processing and analysis task.

Brightness transformations modify pixel brightness and the transformation depends on the properties of a pixel itself. In PBT, output pixel's value depends only on the corresponding input pixel value. Examples of such operators include brightness and contrast adjustments as well as colour correction and transformations.

Contrast enhancement is an important area in image processing for both human and computer vision. It is widely used for medical image processing and as a pre-processing step in speech recognition, texture synthesis, and many other image/video processing applications.

Image augmentation was performed to the training dataset. Specifically, augmentations were performed in memory and the results were not saved, the so-called just-in-time augmentation that is now the standard way for using the approach.

The first type of augmentation performed was horizontal flips of a smaller cropped square image that was expanded to the required size using horizontal reflections within the image.

Feature Extraction

The main idea behind the HOG method is to express the image as a group of local histograms. This histogram consists of gradient orientation and gradient magnitude. This technique calculates the gradient value in a particular area of an image. Each image has the characteristics indicated by the gradient distribution. This characteristic

is obtained by dividing the image into a small area called a cell. Each cell is arranged a histogram of a gradient. The combination of this histogram is used as a descriptor that represents an object.

The initial stage of HOG is calculating the gradient value of the input image. The most common method for calculating the magnitude of a gradient is to use a sobel filter in one or two directions, either horizontally or vertically. Next is to create parts called cells. Each pixel in a cell has its own histogram value based on the value generated in the gradient calculation.

Cells have a size of 4×4 pixels in an image while the block has a size of 2×2 cell or 8×8 pixels. Gabor filter is one filter that can simulate the characteristics of the human visual system in isolating the frequency and specific orientation of the image. This characteristic makes Gabor Filter appropriate for texture recognition applications in computer vision. Spatially, a Gabor function is a sinusoid that is modulated by the Gauss function. We also choose our base networks among these networks with a notable consideration: since we aim at using our method for real-time vehicle applications, the most important criterion to choose our base network is the accuracy/speed trade-off.

Prediction

Given an input image I , a channel C of I is a feature map, where the output pixels are computed from corresponding pixels of the input image. Aggregated channel features are extracted from multiple image channels using pixel lookup method. Many image channels are available for extracting features. For example, a trivial channel of a grayscale image is the image itself. For a color image, each color channel can be used as a channel. Other channels can be computed using various transformations of I . In order to accelerate the speed of feature extraction, all transformations are required to be translational invariant. It means that the transformation needs only to be evaluated once on the entire image rather than separately for each overlapping detection window.

Spatial pooling is used to combine multiple visual descriptors obtained at nearby locations into a lower dimensional descriptor over the pooling region. We follow the work of which is shown that pooling can enhance the robustness of two hand-crafted low-level features, covariance features. : A covariance matrix is a positive semidefinite matrix which provides a measure of the relationship between multiple sets of variates. The diagonal elements of a covariance matrix represent the variance of each feature and non-diagonal elements represent the correlation between different features.

The spatial invariance and robustness of the covariance descriptors can be improved by applying pooling method. There are two common pooling methods in this context: average pooling and max pooling. Max pooling is used in our framework as it has been shown to outperform average pooling in image classification. Max pooling uses the maximum value of a pooling region to represent the pooled features in the region. It aims to retain the most salient information and discard irrelevant details and noises over the pooling region. The image window is divided into multiple dense patches. Covariance features are computed over pixels within each patch. Then, we perform max pooling over a fixed-size pooling region and use the pooled features to represent the covariance features in the pooling region. In fact, multiple covariance matrices within each pooling region are summarized into a single matrix which has better invariance to image deformation and translation. The pooled features extracted from each pooling region is called the spatially pooled covariance.

CHAPTER 7

SOFTWARE TESTING

Testing documentation is the documentation of artifacts that are created during or before the testing of a software application. Documentation reflects the importance of processes for the customer, individual and organization. Projects which contain all documents have a high level of maturity. Careful documentation can save the time, efforts and wealth of the organization.

If the testing or development team gets software that is not working correctly and developed by someone else, so to find the error, the team will first need a document. Now, if the documents are available then the team will quickly find out the cause of the error by examining documentation. But, if the documents are not available then the tester need to do black box and white box testing again, which will waste the time and money of the organization. More than that, Lack of documentation becomes a problem for acceptance.

Benefits of using Documentation

- Documentation clarifies the quality of methods and objectives.
- It ensures internal coordination when a customer uses software application.
- It ensures clarity about the stability of tasks and performance.
- It provides feedback on preventive tasks.
- It provides feedback for your planning cycle.
- It creates objective evidence for the performance of the quality management system.

The test scenario is a detailed document of test cases that cover end to end functionality of a software application in liner statements. The liner statement is considered as a scenario. The test scenario is a high-level classification of testable requirements. These requirements are grouped on the basis of the functionality of a module and obtained from the use cases.

In the test scenario, there is a detailed testing process due to many associated test cases. Before performing the test scenario, the tester has to consider the test cases for each scenario.

In the test scenario, testers need to put themselves in the place of the user because they test the software application under the user's point of view. Preparation of scenarios is the most critical part, and it is necessary to seek advice or help from customers, stakeholders or developers to prepare the scenario.

As per the IEEE Documentation describing plans for, or results of, the testing of a system or component, Types include test case specification, test incident report, test log, test plan, test procedure, test report. Hence the testing of all the above mentioned documents is known as documentation testing.

This is one of the most cost effective approaches to testing. If the documentation is not right: there will be major and costly problems. The documentation can be tested in a number of different ways to many different degrees of complexity. These range from running the documents through a spelling and grammar checking device, to manually reviewing the documentation to remove any ambiguity or inconsistency.

Documentation testing can start at the very beginning of the software process and hence save large amounts of money, since the earlier a defect is found the less it will cost to be fixed.

the most popular testing documentation files are test reports, plans, and checklists. These documents are used to outline the team's workload and keep track of the process. Let's take a look at the key requirements for these files and see how they contribute to the process.

Test strategy. An outline of the full approach to product testing. As the project moves along, developers, designers, product owners can come back to the document and see if the actual performance corresponds to the planned activities.

Test data. The data that testers enter into the software to verify certain features and their outputs. Examples of such data can be fake user profiles, statistics, media content, similar to files that would be uploaded by an end-user in a ready solution.

Test plans. A file that describes the strategy, resources, environment, limitations, and schedule of the testing process. It's the fullest testing document, essential for informed planning. Such a document is distributed between team members and shared with all stakeholders.

Test scenarios. In scenarios, testers break down the product's functionality and interface by modules and provide real-time status updates at all testing stages. A module can be described by a single statement, or require hundreds of statuses, depending on its size and scope.

Test cases. If the test scenario describes the object of testing (what), a scenario describes a procedure (how). These files cover step-by-step guidance, detailed conditions, and current inputs of a testing task. Test cases have their own kinds that depend on the type of testing — functional, UI, physical, logical cases, etc. Test cases compare available resources and current conditions with desired outcomes and determine if the functionality can be released or not.

Traceability Matrix. This software testing documentation maps test cases and their requirements. All entries have their custom IDs — team members and stakeholders can track the progress of any tasks by simply entering its ID to the search.

External documentation collects information from inner documentation but also emphasize on providing a visual data representation — graphs, diagrams, etc.

External reports — these documents collect information on test results and can describe an entire project or a particular piece of functionality.

Test summary report — the file with final test results and findings, presented to stakeholders.

Bug reports — such files keep track of newly encountered bugs and their fixes. We prefer to keep our bug documentation numbered, so it's easier to mention them in

further documentation. Reports are concise and focus on offering tangible solutions. Sometimes, bug reports can only include issue description, if the team hasn't yet found the best approach to fixing the problem.

The combination of internal and external documentation is the key to a deep understanding of all testing processes. Although stakeholders typically have access to the majority of documentation, they mostly work with external files, since they are more concise and tackle tangible issues and results. Internal files, on the other hand, are used by team members to optimize the testing process.

Black Box testers don't care about Unit Testing. Their main goal is to validate the application against the requirements without going into the implementation details.

But as a curiosity or Out of the box thinking, have you ever wondered how developers test their own code? What method do they use to test before releasing code for testing? How is dev-testing important in an agile process? The answer to all this is Unit Testing. I want to educate you on the importance of Unit Testing so that development and testing teams can work more collaboratively to design, test and release an excellent application.

Unit Testing is not a new concept. It's been there since the early days of programming. Usually, developers and sometimes White box testers write Unit tests to improve code quality by verifying each and every unit of the code used to implement functional requirements (aka test drove development TDD or test-first development).

Most of us might know the classic definition of Unit Testing

“Unit Testing is the method of verifying the smallest piece of testable code against its purpose.” If the purpose or requirement failed then the unit test has failed.

In simple words, Unit Testing means – writing a piece of code (unit test) to verify the code (unit) written for implementing requirements.



Fig: 7.0.1 : Unit Testing in SDLC

Unit Testing is used to design robust software components that help maintain code and eliminate the issues in code units. We all know the importance of finding and fixing defects in the early stage of the software development cycle. Unit Testing serves the same purpose.

Unit Testing is an integral part of the agile software development process. When a nightly build run unit test suite should run and report should be generated. If any of the unit tests have failed then the QA team should not accept that build for verification.

If we set this as a standard process, many defects would be caught in the early development cycle, saving much testing time.

I know many developers hate to write unit tests. They either ignore or write bad unit test cases due to tight scheduled or lack of seriousness (yea they write empty unit tests, so 100% of them pass successfully ;-)). It's important to write good unit tests or don't write them at all. It's even more important to provide sufficient time and a supportive environment for real benefits.

- Testing can be done in the early phases of the software development lifecycle when other modules may not be available for integration
- Fixing an issue in Unit Testing can fix many other issues occurring in later development and testing stages

- Cost of fixing a defect found in Unit Testing is very less than the one found in the system or acceptance testing
- Code coverage can be measured
- Fewer bugs in the System and Acceptance testing
- Code completeness can be demonstrated using unit tests. This is more useful in the agile process. Testers don't get the functional builds to test until integration is completed. Code completion cannot be justified by showing that you have written and checked in the code. But running Unit tests can demonstrate code completeness.
- Expect robust design and development as developers write test cases by understanding the specifications first.
- Easily identify who broke the build
- Saves development time: Code completion may take more time but due to decreased defect count overall development time can be saved.

7.1 GENERAL

Unit Testing frameworks are mostly used to help write unit tests quickly and easily. Most of the programming languages do not support unit testing with the inbuilt compiler. Third-party open source and commercial tools can be used to make unit testing even more fun.

List of popular Unit Testing tools for different programming languages:

- ▶ Java framework – JUnit
- ▶ PHP framework – PHPUnit
- ▶ C++ frameworks – UnitTest++ and Google C++
- ▶ .NET framework – NUnit

Python framework – py.test.

Functional Testing types include:

1. Unit Testing
2. Integration Testing
3. System Testing
4. Sanity Testing
5. Smoke Testing
6. Interface Testing
7. Regression Testing
8. Beta/Acceptance Testing
9. Non-functional Testing types include:

Performance Testing

- Load Testing
- Stress Testing
- Volume Testing
- Security Testing
- Compatibility Testing
- Install Testing
- Recovery Testing
- Reliability Testing
- Usability Testing
- Compliance Testing
- Localization Testing

7.3 UNIT TESTING

UNIT TESTING is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or

derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

A unit can be almost anything you want it to be -- a line of code, a method, or a class. Generally though, smaller is better. Smaller tests give you a much more granular view of how your code is performing. There is also the practical aspect that when you test very small units, your tests can be run fast; like a thousand tests in a second fast.

Black Box testers don't care about Unit Testing. Their main goal is to validate the application against the requirements without going into the implementation details.

Unit Testing is not a new concept. It's been there since the early days of programming. Usually, developers and sometimes White box testers write Unit tests to improve code quality by verifying each and every unit of the code used to implement functional requirements (aka test drove development TDD or test-first development).

Most of us might know the classic definition of Unit Testing –

“Unit Testing is the method of verifying the smallest piece of testable code against its purpose.” If the purpose or requirement failed then the unit test has failed.

In simple words, Unit Testing means – writing a piece of code (unit test) to verify the code (unit) written for implementing requirements.

7.4 BLACKBOX TESTING

During functional testing, testers verify the app features against the user specifications. This is completely different from testing done by developers which is unit testing. It checks whether the code works as expected. Because unit testing focuses on the internal structure of the code, it is called the white box testing. On the other hand, functional testing checks app's functionalities without looking at the internal structure of the code, hence it is called black box testing. Despite how flawless the various individual code components may be, it is essential to check that the app is functioning

as expected, when all components are combined. Here you can find a detailed comparison between functional testing vs unit testing.

7.5 INTEGRATION TESTING

INTEGRATION TESTING is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

integration testing: Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems. See also component integration testing, system integration testing.

component integration testing: Testing performed to expose defects in the interfaces and interaction between integrated components. System integration testing: Testing the integration of systems and packages; testing interfaces to external organizations (e.g. Electronic Data Interchange, Internet).

Integration tests determine if independently developed units of software work correctly when they are connected to each other. The term has become blurred even by the diffuse standards of the software industry, so I've been wary of using it in my writing. In particular, many people assume integration tests are necessarily broad in scope, while they can be more effectively done with a narrower scope.

As often with these things, it's best to start with a bit of history. When I first learned about integration testing, it was in the 1980's and the waterfall was the dominant influence of software development thinking. In a larger project, we would have a design phase that would specify the interface and behavior of the various modules in the system. Modules would then be assigned to developers to program. It was not unusual for one programmer to be responsible for a single module, but this would be big enough that it could take months to build it. All this work was done in isolation, and when the programmer believed it was finished they would hand it over to QA for testing.

Integration testing tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems. Integration testing is a key aspect of software testing.

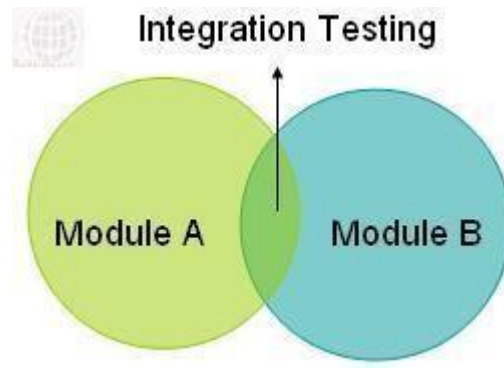


Fig: 7.5.1: Integration Testing

7.6 SYSTEM TESTING

SYSTEM TESTING is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements. System Testing means testing the system as a whole. All the modules/components are integrated in order to verify if the system works as expected or not.

System Testing is done after Integration Testing. This plays an important role in delivering a high-quality product.

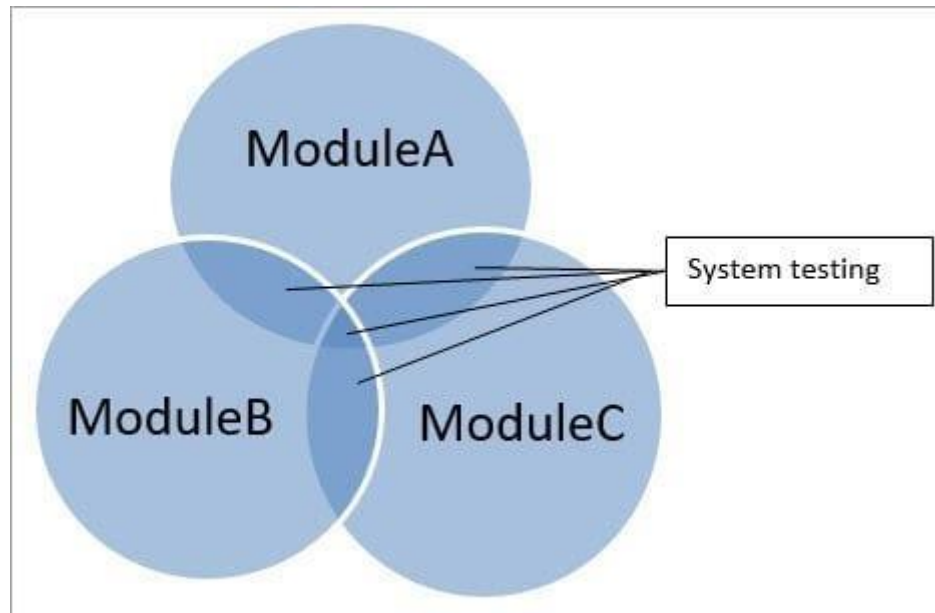


Fig: 7.6.1: System Testing

System testing is a method of monitoring and assessing the behaviour of the complete and fully-integrated software product or system, on the basis of pre-decided specifications and functional requirements. It is a solution to the question "whether the complete system functions in accordance to its pre-defined requirements?"

It's comes under black box testing i.e. only external working features of the software are evaluated during this testing. It does not requires any internal knowledge of the coding, programming, design, etc., and is completely based on users-perspective.

A black box testing type, system testing is the first testing technique that carries out the task of testing a software product as a whole. This System testing tests the integrated system and validates whether it meets the specified requirements of the client.

System testing is a process of testing the entire system that is fully functional, in order to ensure the system is bound to all the requirements provided by the client in the form of the functional specification or system specification documentation. In most cases, it is done next to the Integration testing, as this testing should be covering the end-to-end system's actual routine. This type of testing requires a dedicated Test Plan and other

test documentation derived from the system specification document that should cover both software and hardware requirements. By this test, we uncover the errors. It ensures that all the system works as expected. We check System performance and functionality to get a quality product. System testing is nothing but testing the system as a whole. This testing checks complete end-to-end scenario as per the customer's point of view. Functional and Non-Functional tests also done by System testing. All things are done to maintain trust within the development that the system is defect-free and bug-free. System testing is also intended to test hardware/software requirements specifications. System testing is more of a limited type of testing; it seeks to detect both defects within the "inter-assemblages".

7.7 SANITY TESTING

Sanity Testing is done when as a QA we do not have sufficient time to run all the test cases, be it Functional Testing, UI, OS or Browser Testing. Sanity testing is a subset of regression testing. After receiving the software build, sanity testing is performed to ensure that the code changes introduced are working as expected. This testing is a checkpoint to determine if testing for the build can proceed or not.

The main purpose of this testing is to determine that the changes or the proposed functionality are working as expected. If the sanity test fails, the build is rejected by the testing team to save time and money. It is performed only after the build has cleared the smoke test and been accepted by the Quality Assurance team for further testing. The focus of the team during this testing process is to validate the functionality of the application and not detailed testing. Let's assume that in an e-commerce project, there are mainly five modules- like login page, home page, user profile page, new user creation, etc. There is a defect in the login page when the password field accepts less than four alpha numeric characters and the requirement mentions that this password field should not be below eight characters. So, the bug is reported by the testing team to the development team to resolve it. When the development team fixes the defect and passes it to the testing team. Then the testing team checks if the changes done are working fine. It also checks if it does not have an impact on other related functionalities. Now there is a functionality to update the password in the user profile

page. As part of the sanity check, we would need to validate the login page as well as the profile page to ensure that the checks are working fine at both the places. Sanity tests are generally performed on build where the production deployment is required immediately - like a critical bug fix.

Smoke Testing is done to make sure if the build we received from the development team is testable or not. It is also called as “Day 0” check. It is done at the “build level”.

It helps not to waste the testing time to simply testing the whole application when the key features don't work or the key bugs have not been fixed yet. Here our focus will be on primary and core application work flow.

To conduct smoke testing, we don't write test cases. We just pick the necessary test cases from already written test cases.

Do we really write test cases for all testing types? Here in this article, we have given clear idea on choosing testing types to write test cases.

As mentioned earlier, here in Smoke Testing, our main focus will be on core application work flow. So we pick the test cases from our test suite which cover major functionality of the application. In general, we pick minimal number of test cases that won't take more than half an hour to execute.

After receiving a Software build with the minor issues fixes in code or functionality, Sanity testing is carry out to check whether the bugs reported in previous build are fixed & there is regression introduced due to these fixes i.e. not breaking any previously working functionality. The main aim of Sanity testing to check the planned functionality is working as expected. Instead of doing whole regression testing the Sanity testing is perform.

Sanity tests helps to avoid wasting time and cost involved in testing if the build is failed. Tester should reject the build upon build failure.

After completion of regression testing the Sanity testing is started to check the defect fixes & changes done in the software application is not breaking the core functionality of the software. Typically this is done nearing end of SDLC i.e. while releasing the

software. You can say that sanity testing is a subset of acceptance testing. We can also say Tester Acceptance Testing for Sanity testing.

7.8 REGRESSION TESTING

Regression Testing is a type of testing that is done to verify that a code change in the software does not impact the existing functionality of the product. This is to make sure the product works fine with new functionality, bug fixes or any change in the existing feature. Previously executed test cases are re-executed in order to verify the impact of change.

Regression Testing is a Software Testing type in which test cases are re-executed in order to check whether the previous functionality of the application is working fine and the new changes have not introduced any new bugs.

This test can be performed on a new build when there is a significant change in the original functionality that too even in a single bug fix.

For regression testing to be effective, it needs to be seen as one part of a comprehensive testing methodology that is cost-effective and efficient while still incorporating enough variety—such as well-designed frontend UI automated tests alongside targeted unit testing, based on smart risk prioritization—to prevent any aspects of your software applications from going unchecked. These days, many Agile work environments employing workflow practices such as XP (Extreme Programming), RUP (Rational Unified Process), or Scrum appreciate regression testing as an essential aspect of a dynamic, iterative development and deployment schedule.

But no matter what software development and quality-assurance process your organization uses, if you take the time to put in enough careful planning up front, crafting a clear and diverse testing strategy with automated regression testing at its core, you can help prevent projects from going over budget, keep your team on track, and, most importantly, prevent unexpected bugs from damaging your products and your company's bottom line.

Performance testing is the practice of evaluating how a system performs in terms of responsiveness and stability under a particular workload. Performance tests are typically executed to examine speed, robustness, reliability, and application size. The process incorporates “performance” indicators such as:

- Browser, page, and network response times
- Server request processing times
- Acceptable concurrent user volumes
- Processor memory consumption; number and type of errors that might be encountered with app

Performance testing gathers all the tests that verify an application’s speed, robustness, reliability, and correct sizing. It examines several indicators such as a browser, page and network response times, server query processing time, number of acceptable concurrent users architected, CPU memory consumption, and number/type of errors which may be encountered when using an application.

Performance testing is the testing that is performed to ascertain how the components of a system are performing under a certain given situation.

Resource usage, scalability, and reliability of the product are also validated under this testing. This testing is the subset of performance engineering, which is focused on addressing performance issues in the design and architecture of a software product.



Fig: 7.8.1: Regression Testing

Software Performance testing is type of testing perform to determine the performance of system to major the measure, validate or verify quality attributes of the system like responsiveness, Speed, Scalability, Stability under variety of load conditions. The system is tested under a mixture of load conditions and check the time required responding by the system under varying workloads. Software performance testing involves the testing of application under test to ensure that application is working as expected under variety of load conditions. The goal of performance testing is not only find the bugs in the system but also eliminate the performance bottlenecks from the system.

Load Testing is type of performance testing to check system with constantly increasing the load on the system until the time load is reaches to its threshold value. Here Increasing load means increasing number of concurrent users, transactions & check the behavior of application under test. It is normally carried out underneath controlled environment in order to distinguish between two different systems. It is also called as “Endurance testing” and “Volume testing”. The main purpose of load testing is to monitor the response time and staying power of application when system is performing

well under heavy load. Load testing comes under the Non Functional Testing & it is designed to test the non-functional requirements of a software application.

Load testing is perform to make sure that what amount of load can be withstand the application under test. The successfully executed load testing is only if the specified test cases are executed without any error in allocated time.

Simple examples of load testing:

- Testing printer by sending large job.
- Editing a very large document for testing of word processor
- Continuously reading and writing data into hard disk.
- Running multiple applications simultaneously on server.
- Testing of mail server by accessing thousands of mailboxes
- In case of zero-volume testing & system fed with zero load.

CHAPTER 8

CONCLUSION & FEATURE ENCHANCEMENT

8.1 CONCLUSION

is more often used for evaluating patients with suspected acute stroke which is represents high cost and low value endeavor. We have developed an algorithm for helping clinicians determine which patients evaluated for acute stroke need a classifiers. This algorithm has superior diagnostic accuracy in comparison to the algorithm, which is currently the most accurate and widely used algorithm for determining which acute stroke patient needs a classifiers.

Using readily available clinical information, our algorithm is capable of helping clinicians avoid nearly two times the number of acute stroke that can be avoided when the algorithm is utilized.

Clinical implementation of this algorithm can help decrease financial costs associated with evaluations for traumatic brain injury, while decreasing patient exposure to avoidable ionizing radiation. Although the algorithm currently utilizes only easily obtainable clinical data.

8.1 FUTURE ENHANCEMENT

In the complex screenshot, the accuracy of component recognition in our system is not very ideal. To further strengthen the image processing part, and the segmentation of screenshot needs to be more accurate, so as to improve the accuracy of control recognition

APPENDIX CODE

```
from flask import Flask, request, render_template, redirect, url_for, session
import pypyodbc
```

```
import json
from os.path import isfile, join
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
from skimage.io import imread
from os.path import basename
from glob import glob
import string
import random
import os
import sys
import hashlib
import shutil
```

```
from keras.models import Model
from keras.layers import LSTM, Bidirectional, Dense, Input, Embedding, concatenate,
Dropout
from keras.preprocessing.text import Tokenizer
from sklearn.preprocessing import LabelEncoder
from PIL import Image as PImage
from keras.models import Sequential
from keras.layers import RepeatVector, Input, Dropout, Conv2D, MaxPooling2D,
Flatten, Conv2DTranspose, UpSampling2D, Reshape, Dense
from sklearn.model_selection import train_test_split
from keras.callbacks import ModelCheckpoint, LearningRateScheduler,
```

```
EarlyStopping, ReduceLROnPlateau
from keras.backend import clear_session
from keras.models import model_from_json
from keras.optimizers import RMSprop
from keras import *
```

```
from RoleModel import RoleModel
from UserModel import UserModel
from Constants import connString
```

```
app = Flask(__name__)
app.secret_key = "MySecret"
ctx = app.app_context()
ctx.push()
```

```
with ctx:
    pass
```

```
userName = ""
roleObject = None
message = ""
msgType = ""
```

```
def initialize():
    global message, msgType
    message = ""
    msgType=""
```

```
def processRole(optionID):
```

```

if optionID == 10 :
    if roleObject.canRole == False :
        return False
if optionID == 20 :
    if roleObject.canUser == False :
        return False
if optionID == 30 :
    if roleObject.CL111 == False :
        return False
if optionID == 40 :
    if roleObject.CL222 == False :
        return False
if optionID == 50 :
    if roleObject.CL333 == False :
        return False
return True

```

```
@app.route('/')

```

```
def index():

```

```
    global userID, userName

```

```
    return render_template('Login.html') # when the home page is called Index.html
will be triggered.

```

```
@app.route('/processLogin', methods=['POST'])

```

```
def processLogin():

```

```
    global userID, userName, roleObject

```

```
    userName= request.form['userName']

```

```
    password= request.form['password']

```

```
    conn1 = pypyodbc.connect(connString, autocommit=True)

```

```
    cur1 = conn1.cursor()

```

```
sqlcmd1 = "SELECT * FROM UserTable WHERE userName = '"+userName+"'
AND password = '"+password+"' AND isActive = 1";
```

```
cur1.execute(sqlcmd1)
row = cur1.fetchone()
```

```
cur1.commit()
if not row:
    return render_template('Login.html', processResult="Invalid Credentials")
userID = row[0]
userName = row[3]
```

```
cur2 = conn1.cursor()
sqlcmd2 = "SELECT * FROM Role WHERE RoleID = '"+str(row[6])+"'";
cur2.execute(sqlcmd2)
row2 = cur2.fetchone()
```

```
if not row2:
    return render_template('Login.html', processResult="Invalid Role")
```

```
roleObject = RoleModel(row2[0], row2[1],row2[2],row2[3],row2[4],row2[5])
```

```
return render_template('Dashboard.html')
```

```
@app.route("/ChangePassword")
```

```
def changePassword():
```

```
    global userID, userName
```

```
    return render_template('ChangePassword.html')
```

```
@app.route("/ProcessChangePassword", methods=['POST'])
```

```
def processChangePassword():
```

```

global userID, userName
oldPassword= request.form['oldPassword']
newPassword= request.form['newPassword']
confirmPassword= request.form['confirmPassword']
conn1 = pypyodbc.connect(connString, autocommit=True)
cur1 = conn1.cursor()
sqlcmd1 = "SELECT * FROM UserTable WHERE userName = '"+userName+"'
AND password = '"+oldPassword+"'";
cur1.execute(sqlcmd1)
row = cur1.fetchone()
cur1.commit()
if not row:
    return render_template('ChangePassword.html', msg="Invalid Old Password")

if newPassword.strip() != confirmPassword.strip() :
    return render_template('ChangePassword.html', msg="New Password and
Confirm Password are NOT same")

conn2 = pypyodbc.connect(connString, autocommit=True)
cur2 = conn2.cursor()
sqlcmd2 = "UPDATE UserTable SET password = '"+newPassword+"' WHERE
userName = '"+userName+"'";
cur1.execute(sqlcmd2)
cur2.commit()
return render_template('ChangePassword.html', msg="Password Changed
Successfully")

@app.route("/Dashboard")
def Dashboard():
    global userID, userName

```

```
return render_template('Dashboard.html')
```

```
@app.route("/Information")
```

```
def Information():
```

```
    global message, msgType
```

```
    return render_template('Information.html', msgType=msgType, message =  
message)
```

```
@app.route("/UserListing")
```

```
def UserListing():
```

```
    global userID, userName
```

```
    global message, msgType, roleObject
```

```
    if roleObject == None:
```

```
        message = "Application Error Occurred"
```

```
        msgType="Error"
```

```
        return redirect(url_for('Information'))
```

```
    canRole = processRole(10)
```

```
    if canRole == False:
```

```
        message = "You Don't Have Permission to Access User"
```

```
        msgType="Error"
```

```
        return redirect(url_for('Information'))
```

```
conn2 = pypyodbc.connect(connString, autocommit=True)
```

```
cursor = conn2.cursor()
```

```

sqlcmd1 = "SELECT * FROM UserTable ORDER BY userName"
cursor.execute(sqlcmd1)
records = []

while True:
    dbrow = cursor.fetchone()
    if not dbrow:
        break

    conn3 = pypyodbc.connect(connString, autocommit=True)
    cursor3 = conn3.cursor()
    temp = str(dbrow[6])
    sqlcmd3 = "SELECT * FROM Role WHERE RoleID = '"+temp+"'"
    cursor3.execute(sqlcmd3)
    rolerow = cursor3.fetchone()
    roleModel = RoleModel(0)
    if rolerow:
        roleModel = RoleModel(rolerow[0],rolerow[1])
    else:
        print("Role Row is Not Available")

    row = UserModel(dbrow[0], dbrow[1], dbrow[2], dbrow[3], dbrow[4], dbrow[5],
dbrow[6], roleModel=roleModel)
    records.append(row)
    return render_template('UserListing.html', records=records)

@app.route("/UserOperation")
def UserOperation():

    global userID, userName

```



```
print("roleDDDraw[1]>>>>>>>>>>>>>>>>>>>>>>", roleDDDraw[1])  
roleDDObj = RoleModel(roleDDDraw[0], roleDDDraw[1])  
rolesDDLlist.append(roleDDObj)
```

```
row = UserModel(0)
```

```
if operation != "Create" :
```

```
unqid = request.args.get('unqid').strip()
conn2 = pypyodbc.connect(connString, autocommit=True)
cursor = conn2.cursor()
sqlcmd1 = "SELECT * FROM UserTable WHERE UserID = '"+unqid+"'"
cursor.execute(sqlcmd1)
dbrow = cursor.fetchone()
if dbrow:
```

```
conn3 = pypyodbc.connect(connString, autocommit=True)
cursor3 = conn3.cursor()
temp = str(dbrow[6])
sqlcmd3 = "SELECT * FROM Role WHERE RoleID = '"+temp+"'"
cursor3.execute(sqlcmd3)
rolerow = cursor3.fetchone()
roleModel = RoleModel(0)
if rolerow:
    roleModel = RoleModel(rolerow[0],rolerow[1])
else:
    print("Role Row is Not Available")
row = UserModel(dbrow[0], dbrow[1], dbrow[2], dbrow[3], dbrow[4], dbrow[5],
dbrow[6], roleModel=roleModel)
```

```
return render template('UserOperation.html', row = row, operation=operation,
```

```
rolesDDLList=rolesDDLList )
```

```
@app.route("/ProcessUserOperation",methods = ['POST'])
```

```
def processUserOperation():
```

```
    global userName, userID
```

```
    operation = request.form['operation']
```

```
    unqid = request.form['unqid'].strip()
```

```
    userName= request.form['userName']
```

```
    emailid= request.form['emailid']
```

```
    password=request.form['password']
```

```
    contactNo= request.form['contactNo']
```

```
    isActive = 0
```

```
    if request.form.get("isActive") != None :
```

```
        isActive = 1
```

```
    roleID= request.form['roleID']
```

```
conn1 = pypyodbc.connect(connString, autocommit=True)
```

```
cur1 = conn1.cursor()
```

```
if operation == "Create" :
```

```
    sqlcmd = "INSERT INTO UserTable( userName,emailid, password,contactNo,isActive, roleID) VALUES('"+userName+"','"+emailid+"', '"+password+"', '"+contactNo+"', '"+str(isActive)+"', '"+str(roleID)+"')"
```

```
if operation == "Edit" :
```

```
    sqlcmd = "UPDATE UserTable SET userName = '"+userName+"', emailid = '"+emailid+"', password = '"+password+"',contactNo='"+contactNo+"', isActive = "
```

```
""+str(isActive)+"", roleID = ""+str(roleID)+" WHERE UserID = ""+unqid+""
```

```
if operation == "Delete" :
```

```
sqlcmd = "DELETE FROM UserTable WHERE UserID = '"+unqid+'"'
```

```
if sqlcmd == "" :
```

```
return redirect(url_for('Information'))
```

```
cur1.execute(sqlcmd)
```

```
cur1.commit()
```

```
conn1.close()
```

```
return redirect(url_for("UserListing"))
```

■■■

Role Operation Start

...

```
@app.route("/RoleListing")
```

```
def RoleListing():
```

global message, msgType

[illegible]

```
if roleObject == None:
```

```
message = "Application Error Occurred"
```

```
msgType="Error"
```

```
return redirect(url_for('Information'))
```

```

canRole = processRole(20)

if canRole == False:
    message = "You Don't Have Permission to Access Role"
    msgType="Error"
    return redirect(url_for('Information'))

searchData = request.args.get('searchData')
print(searchData)
if searchData == None:
    searchData = "";
conn2 = pypyodbc.connect(connString, autocommit=True)
cursor = conn2.cursor()
sqlcmd1 = "SELECT * FROM Role WHERE roleName LIKE '"+searchData+"%"
print(sqlcmd1)
cursor.execute(sqlcmd1)
records = []

while True:
    dbrow = cursor.fetchone()
    if not dbrow:
        break

    row =
    RoleModel(dbrow[0],dbrow[1],dbrow[2],dbrow[3],dbrow[4],dbrow[5],dbrow[6])

    records.append(row)

    return render_template('RoleListing.html', records=records,
searchData=searchData)

```

```

@app.route("/RoleOperation")
def RoleOperation():

    global message, msgType
    if roleObject == None:
        message = "Application Error Occurred"
        msgType="Error"
        return redirect(url_for('/'))
    canRole = processRole(120)

    if canRole == False:
        message = "You Don't Have Permission to Access Role"
        msgType="Error"
        return redirect(url_for('Information'))

    operation = request.args.get('operation')
    unqid = ""
    row = RoleModel(0, "",0,0,0,0)
    if operation != "Create" :
        unqid = request.args.get('unqid').strip()

    conn2 = pypyodbc.connect(connString, autocommit=True)
    cursor = conn2.cursor()
    sqlcmd1 = "SELECT * FROM Role WHERE RoleID = '"+unqid+"'"
    cursor.execute(sqlcmd1)
    while True:
        dbrow = cursor.fetchone()
        if not dbrow:
            break
        row

```

=

```
RoleModel(dbrow[0],dbrow[1],dbrow[2],dbrow[3],dbrow[4],dbrow[5],dbrow[6])
```

```
return render_template('RoleOperation.html', row = row, operation=operation )
```

```
@app.route("/ProcessRoleOperation", methods=['POST'])
```

```
def ProcessRoleOperation():
```

```
    global message, msgType
```

```
    if roleObject == None:
```

```
        message = "Application Error Occurred"
```

```
        msgType="Error"
```

```
        return redirect(url_for('/'))
```

```
    canRole = processRole(120)
```

```
    if canRole == False:
```

```
        message = "You Don't Have Permission to Access Role"
```

```
        msgType="Error"
```

```
        return redirect(url_for('Information'))
```

```
print("ProcessRole")
```

```
operation = request.form['operation']
```

```
if operation != "Delete" :
```

```
    roleName = request.form['roleName']
```

```
    canRole = 0
```

```
    canUser = 0
```

```
    CL111 = 0
```

```
    CL222 = 0
```

```
    CL333 = 0
```

```

if request.form.get("canRole") != None :
    canRole = 1
if request.form.get("canUser") != None :
    canUser = 1
if request.form.get("CL111") != None :
    CL111 = 1
if request.form.get("CL222") != None :
    CL222 = 1
if request.form.get("CL333") != None :
    CL333 = 1

```

```

print(1)
unqid = request.form['unqid'].strip()
print(operation)
conn3 = pypyodbc.connect(connString, autocommit=True)
cur3 = conn3.cursor()

```

```

sqlcmd = ""
if operation == "Create" :
    sqlcmd = "INSERT INTO Role (roleName, canRole, canUser, CL111, CL222,
CL333) VALUES ('"+roleName+"', '"+str(canRole)+"', '"+str(canUser)+"',
 '"+str(CL111)+"', '"+str(CL222)+"', '"+str(CL333)+"')
if operation == "Edit" :
    print("edit inside")
    sqlcmd = "UPDATE Role SET roleName = '"+roleName+"', canRole =
 '"+str(canRole)+"', canUser = '"+str(canUser)+"', CL111 = '"+str(CL111)+"', CL222 =

```

```

""+str(CL222)+"", CL333 = ""+str(CL333)+" WHERE RoleID = ""+unqid+""
if operation == "Delete" :
    conn4 = pypyodbc.connect(connString, autocommit=True)
    cur4 = conn4.cursor()
    sqlcmd4 = "SELECT roleID FROM UserTable WHERE roleID = ""+unqid+""
    cur4.execute(sqlcmd4)
    dbrow4 = cur4.fetchone()
    if dbrow4:
        message = "You can't Delete this Role Since it Available in Users Table"
        msgType="Error"
        return redirect(url_for('Information'))

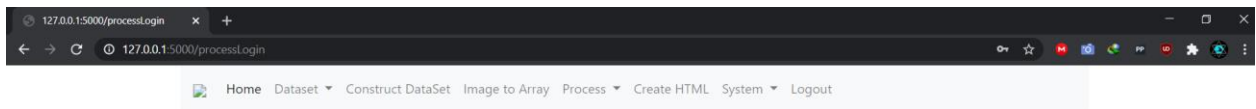
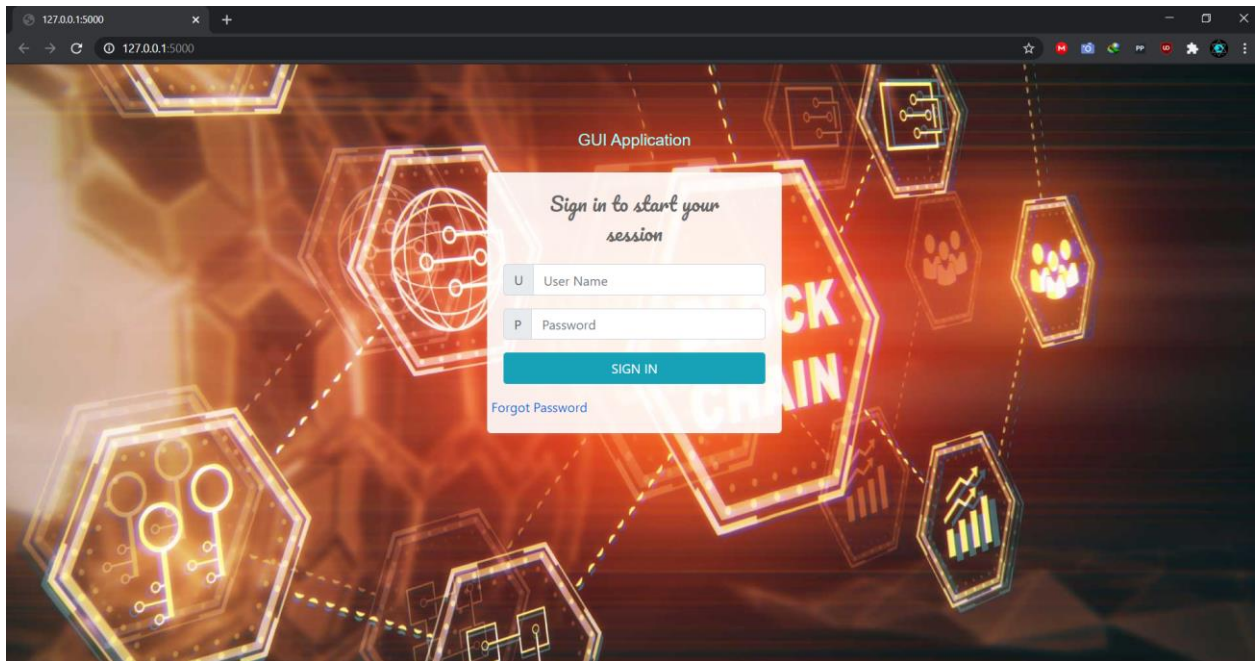
    sqlcmd = "DELETE FROM Role WHERE RoleID = ""+unqid+""
    print(operation, sqlcmd)
    if sqlcmd == "" :
        return redirect(url_for('Information'))
    cur3.execute(sqlcmd)
    cur3.commit()

    return redirect(url_for('RoleListing'))

"""
Role Operation End
"""

```

APPENDIX SCREENSHOTS



127.0.0.1:5000/First50FilesInfoResult			
Home Dataset Construct DataSet Image to Array Process Create HTML System Logout			
First 50 Dataset Files Info Result			
Path	Source	File Type	Field
static\datasets\web\all_data\training_features\00150311-A7AE-4804-8B7D-9273687B4FC0.gui	training_features	gui	00150311-A7AE-4804-8B7D-9273687B4FC0
static\datasets\web\all_data\training_features\00190F39-0DE9-47EB-B0C2-856FDD3ACE62.gui	training_features	gui	00190F39-0DE9-47EB-B0C2-856FDD3ACE62
static\datasets\web\all_data\training_features\007798BE-DD64-4909-9909-24F5C8044A7B.gui	training_features	gui	007798BE-DD64-4909-9909-24F5C8044A7B
static\datasets\web\all_data\training_features\00CDC9A8-3D73-4291-90EF-49178E408797.gui	training_features	gui	00CDC9A8-3D73-4291-90EF-49178E408797
static\datasets\web\all_data\training_features\00E15BB2-5568-4466-BA18-A8A8D34FC61C.gui	training_features	gui	00E15BB2-5568-4466-BA18-A8A8D34FC61C
static\datasets\web\all_data\training_features\011CEB4E-C603-4359-AEA8-D3112A97E9C5.gui	training_features	gui	011CEB4E-C603-4359-AEA8-D3112A97E9C5
static\datasets\web\all_data\training_features\011D9C15-AF65-4660-B7C9-ECA8676185FE.gui	training_features	gui	011D9C15-AF65-4660-B7C9-ECA8676185FE
static\datasets\web\all_data\training_features\01357379-D6C2-4B01-AA66-	training_features	gui	01357379-D6C2-4B01-AA66-

127.0.0.1:5000/ConstructDataSet

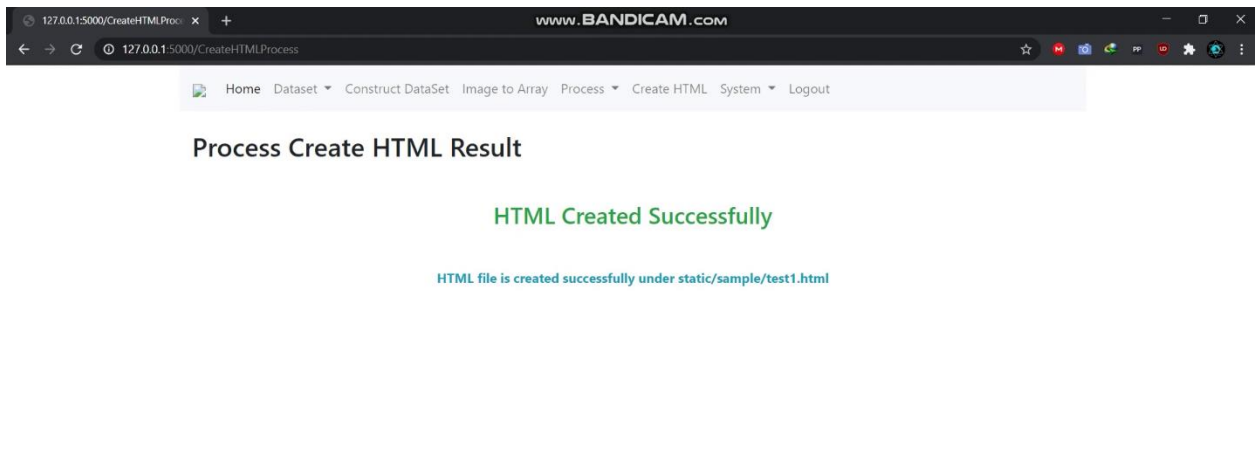
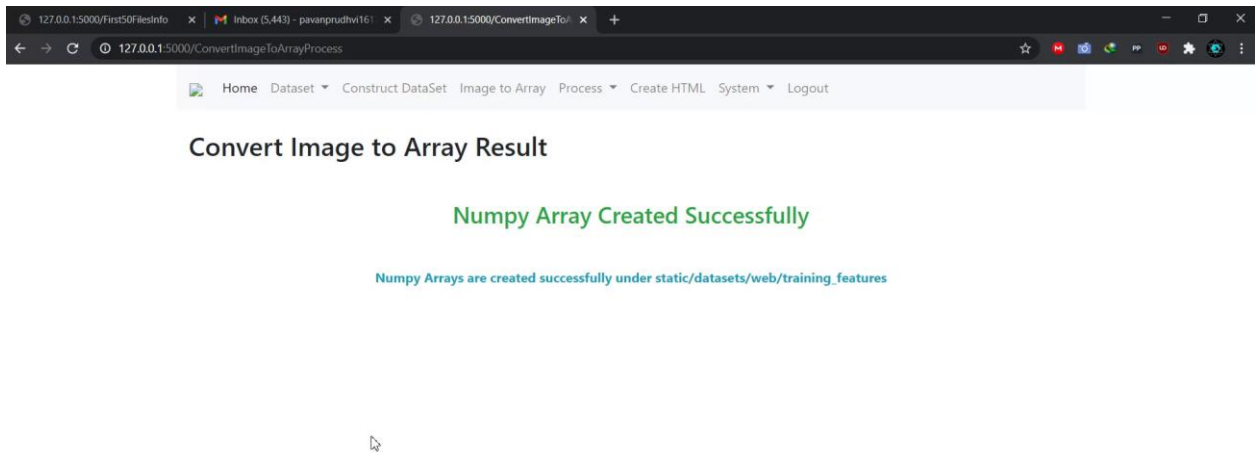
127.0.0.1:5000/ConstructDataSetProcess

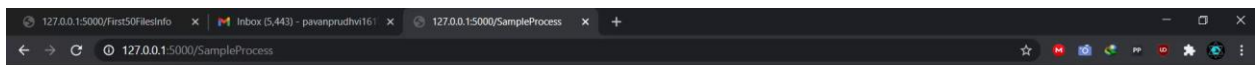
Home Dataset Construct DataSet Image to Array Process Create HTML System Logout

Process Construct Dataset Result

Dataset Constructed Successfully

Two folders (training & eval) are created successfully under static/datasets/web





Home Dataset Construct DataSet Image to Array Process Create HTML System Logout

Process Sample GUI Result

Sample GUI Created Successfully

Sample GUI are created successfully under static/sample/test1.gui

