

A
Technical Report
on
**GENERATING HTML CODE AUTOMATICALLY FROM
MOCK UP IMAGES USING DEEP LEARNING**

Submitted to CMR Institute of Technology in the partial fulfillment of the requirement of

Technology Exploration and Social Innovation-3

Of

III B.Tech I- Semester

in

COMPUTER SCIENCE ENGINEERING

Submitted by

**T.SIRICHANDANA
K.OMESHWAR**

**(19R01A05B5)
(19R01A04M6)**

Under the esteemed guidance of

*Mr. P. Nagaraja Kumar
Assistant Professor*

*Mrs. M. Nagageetha
Assistant Professor*



**CMR INSTITUTE OF TECHNOLOGY
(UGC-AUTONOMOUS)**

(Approved by AICTE, Permanently Affiliated to JNTU Hyderabad, Accredited by NBA, Accredited by NAAC with 'A' Grade)

Kandlakoya (V), Medchal Road, Hyderabad – 501 401

2021-2022

Department of Freshman Engineering

Certificate

This is to certify that the technical report entitled “**GENERATING HTML CODE AUTOMATICALLY FROM MOCK UP IMAGES USING DEEP LEARNING**” is the bonafide work done and submitted by

T.SIRICHANDANA

(19R01A05B5)

K.OMESHWAR

(19R01A04J6)

towards the partial fulfillment of the requirement of Technology Exploration and Social Innovation (TESI) Laboratory of **III B. Tech I-Semester in Freshman Engineering** is a record of bonafide work carried out by them during the period **Aug 2021 to Dec 2021**.

Co- Ordinator 1

Mr. P. Nagaraja Kumar

Co- Ordinator 2

Mrs.M. Nagageetha

Head of Department

Prof. P. Pavan Kumar

ACKNOWLEDGEMENT

We are deeply indebted to **Mr. Nagaraja Kumar Pateti**, Assistant Professor, & **Mrs. M. Nagageetha**, Assistant Professor, Department of Electronics and Communication Engineering, the guiding force behind this project, we want to thank them for giving us the opportunity to work under their guidance. They were always available to share with us their deep insights, wide knowledge and extensive experience. Their advices have value lasting much beyond this project.

We would like to express our deep gratitude to **Dr. B. Sridhar Babu**, *Dean* IIIC CMRIT for providing us an opportunity to innovate, He has been great sources of inspiration to us and we thank him from the bottom of our heart.

We express my respects to **Prof P. Pavan Kumar**, Head of Department, Fresh man Engineering Department for encouraging us throughout our project and for his support.

We are very thankful to **Dr. B. Satyanarayana**, Principal of CMR Institute of Technology for providing us with the opportunity and facilities required to accomplish our project.

We express our gratitude to **Dr. M. Janga Reddy**, Director of CMR Institute of Technology for providing us with the opportunity and facilities required to accomplish our project.

Also we would like to thank all teaching and non-teaching members of Freshman Department for their generous help in various ways for the completion of this report.

Last but not least we would like to thank my parents. They are my first teachers when we came into this world, who taught me the value of hard work by their own example and to our friends whose support was very valuable in completion of the work.

T.SIRICHANDANA

(19R01A05B5)

K.OMESHWAR

(19R01A04M6)

ABSTRACT

The designing of a website starts with building mock-ups for a particular site or web pages by manually or using visual computerization and specified mock-up development tools. This method is normally repeated on various more occasions until the perfect design is made. It's common practice for developers to transform a web page mock-up into code.

This process is repetitive, expensive and time consuming. Hence, proposed system aims to automate this process. Hand drawn image of form is given as an input and it is processed, and various components are detected. Once the components are detected they are cropped, and these components are recognized using deep learning CNN methods. On recognition of the component equivalent HTML code is constructed using HTML builder algorithm.

This approach to automate the code generation from mock-ups will reduce time and it is cost-effective.

Keywords:

HTML, CNN.

CONTENTS

ACKNOWLEDGMENTS	VII
ABSTRACT	VIII
CONTENTS	IX
LIST OF FIGURES	XI
LIST OF TABLES	XIII
CHAPTER-1 INTRODUCTION	1
1.1 Overview	1
1.1.1 Aim	2
1.1.2 Existing System	2
1.1.3 Proposed System	3
1.2 Problem Statement	3
1.3 Objectives	4
1.4 Organization of the project	4
CHAPTER-2 LITURATURE SURVEY	5
CHAPTER-3 DOMAIN SPECIFICATIONS	7
3.1 System Components	8
3.2 Modules	9
3.2.1 Upload Datasets Files	9
3.2.2 Object Detection and Cropping	9
3.2.3 Object Recognition	9
3.2.4 HTML Builder	10
3.3 Model Diagram	10

Data Flow Diagram	11
UML Diagrams	12
Data Dictionary	18
CHAPTER-4 METHODOLOGY	20
System Requirements	20
Functional Requirements	20
Non-Functional Requirements	21
Hardware Requirements	22
Software Requirements	22
Implementation	24
Source Code	26
CHAPTER-5 RESULTS	33
CHAPTER-6 ADVANTAGES AND DISADVANTAGES	38
Advantages	38
Disadvantages	38
Applications	38
CHAPTER-7 CONCLUSIONS	39
Present Work	39
Future Scope	39
REFERENCES	40

List of Figures

Fig. No	Fig. Name	Page.No
3.1	Structure of Automatic Generating of HTML Code	10
3.2	Data Flow Diagram	11
3.3	Use Case Diagram	12
3.4	Class Diagram	13
3.5	Activity Diagram	14
3.6	Sequence Diagram	15
3.7	State Chart Diagram	16
3.8	ER Diagram	17
3.9	Component Diagram	18
4.1	Convolution Layer	25
4.2	Applying RELU	25
4.3	Convolutional Neural Networks	26
5.1	Login Page	33
5.2	Registration Page	34
5.3	Upload Page	34

Upload Image	35
Image Processing Page	35
HTML Conversation	36
HTML Code Generation	36
Web Page Created from Mockup Image	37

List of Tables

Table. No	Table Name	Page. No
3.1	Login authentication	19
3.2	Upload image	19

Acronyms Used

CNN	Convolution Neural network
REMAUI	Reverse Engineering Mobile Application User Interfaces
HTML	Hyper Text Markup Language
XML	Extensible Markup Language
GUI	Graphical User Interface
OCR	Optical Character Recognition
KNN	K-nearest neighbor
UML	Unified Modeling Language
CSV	Comma Separated Value
SQL	Sequence Query Language
ReLU	Rectified Linear Unit
ERD	Entity Relationship Diagram
NFR	Non-Functional Requirement
OS	Operating System

CHAPTER-1 INTRODUCTION

OVERVIEW

The importance of the Internet web sites has increased considerably due to the progress made in today's technology. Nowadays web sites reflect the face of the states, institutions, communities, people, etc. Web sites are available in almost every field, from knowledge to social work, from games to training and many more. Web sites created by companies come to the fore for financial reasons for product marketing or advertising purposes. On the other hand, official institutions aim to provide more efficient services.

At the front-end of every web site is a web page which is the part of the web site that interacts with the user. It is very important to serve a page that attracts the attention of the end-user, is easy to use and has tough functional features. However, developing web pages that respond efficiently to these needs involves a very burdensome process.

At present programmer perform this task manually which is costly and error prone. In the preparation of web pages, graphic designers, software specialists, end-users, corporate authorities and people employed in many different areas are required to work together. Usually, the process starts with the mock-up design of the user interface by the graphic designers or mock-up artists, either on paper or a graphic editing software, in line with the needs of the institution. Software experts write code for the web pages based on these drafts. The resulting web pages may change based on feedback received by the end users. There process involves a lot of repetitive tasks. Rewriting the code for components with similar functions and page structures changing over time makes the process tedious. This reveals the need to explore more efficient solutions in web page design. There are a lot of routine tasks in their operation. The process of rewriting code for components with identical functions and page configurations that change over time is tedious.

The process of automatic code generation based on the hand drawn mockup images of Graphical User Interface created by designers is expensive in practice. There are various real time software applications that provide facility to generate code for user interfaces such as Android Studio, Xcode and Eclipse IDE. Proposed system presents an approach that constructs HTML code for hand drawn GUI image. It involves processes namely detection, dilation, erosion, classification and assembly of an image. The initial process involves detecting borders of various GUI components such as text boxes, labels,

drop down list, check box, button, etc. Once the GUI components are detected, they need to be classified. Thus, second task is image classification. The proposed system is implemented using CNN model. CNN can detect multiple regions of image (up to 2000 different regions). Once object detection and image classification are performed, HTML code will be generated for specific component.

Aim

The Main aim of the project is to generate HTML code automatically from mock up images using Deep learning.

Existing system

The importance of Internet websites has increased due to progress in technology. Websites reflects states, institutions, communities and people which are used for product marketing or advertising purpose. Any website is the interface from which user interacts with the system. Hence, it's very important to design interactive web pages which attract attention of users. In existing system, software experts and designers along with the end-users work together for developing design for web page. Draft of design of user interface is created by designers either on paper or graphics tools. Then based on this draft the code for the web pages is developed by software experts. There may be changes in web pages according to the feedback received by the end users and this is repetitive task. Rewriting code for similar components is tedious task. Hence, to automate the production of web pages is all about Code Generation using CNN Algorithm.

An algorithm named Reverse Engineering Mobile Application User Interfaces (REMAUI) finds the elements of the UI a mobile application for example buttons, text-boxes and pictures, it makes the code for them from the screenshots of an application window. It transforms to the code from the screen pictures or drawings for mobile platforms, PC vision and optical character acknowledgment techniques are utilized. Despite the fact that the REMAUI strategy works effectively, it doesn't assist cross-page change and animations inside the page. Creators built up the P2A algorithm to cure the lacks of the REMAUI calculation. Creators built up the pix2code algorithm which expects to change over the graphical interface for a website page to structured code using deep learning with convolution and repetitive neural networks.

The algorithm Redraw takes mock-ups of mobile application screens and makes an organized XML code for it. In the first phase of their implementation PC vision procedures

are utilized to distinguish singular GUI components. The second phase includes the order of the recognized components as indicated by their function, for example, toggle button, text area, etc. Right now, convolution neural networks are utilized. In the last phase, the XML code is produced by joining.

Proposed system

Producing computer code written in a given programming language from a mock-up image can be differ with the undertaking of creating English printed sketch from a scene photography. In the two situations, we need to deliver a variable-length series of tokens from pixel value. We would thus be able to partition our concern into three sub-issues

Initial, a computer vision issue of understanding the given scene (i.e. HTML picture) and inducing the item present, their personalities, positions, and poses (for example buttons, labels, component). Second, a language displaying issue of getting content (for example code) and producing grammatically and semantically right examples. The last test is to utilize the answers for both past sub-issues by exploit the inactive factors derived from scene comprehension to produce relating textual description (for example PC code as opposed to English) of the articles characterized by these factors. In this stage deep convolutional neural networks are used. In the last stage, the XML code is generated by combining with the CNN algorithm according to web programming hierarchy. Nowadays open-source code libraries such as Github are used very frequently to share code and applications. It is a common practice to investigate this repository and reuse code when starting or improving software projects. The shared codes in these libraries reduce the same code being written over and over by different people. This interface is then searched in existing libraries to obtain similar interfaces. These interfaces are turned into operable codes and returned to the end user to select the most suitable interface.

PROBLEM STATEMENT

Rewriting the code for components with similar functions and page structures changing over time makes the process tedious. This reveals the need to explore more efficient solutions in web page design. Instead of wasting time and money on repeatedly designing and coding UI, a precise computerized approach would almost certainly be required.

OBJECTIVES

1. Reduce the time taken and wasted in rewriting the code for components with similar functions and page structures that change over time.
2. To manage the time and money used for repeatedly designing and coding UI, a precise computerized approach would almost certainly be required.
3. It will be easier for the programmers and designers to work smartly by utilizing the automatic system for generating webpage from images which doesn't need any rework.

ORGANIZATION OF THE PROJECT

Chapter-1 Explains about the Introduction of the project and describes about its aim.

Chapter-2 Explains about the description of the components and it contains about the literature survey of the project.

Chapter-3 Explains about the project Domain Specifications of the project.

Chapter-4 Explains about the Methodology involved the project and it contains the code.

Chapter-5 Explains about the outcome of the project.

CHAPTER-2 LITERATURE SURVEY

A. Generating Code from GUI

A system is used to transform GUI screenshots into computer code. Deep learning methods such as Convolutional and Recurrent Neural Networks were used for automatic generation of code from an input image with accuracy of 77% for platforms like iOS, Android and web-based technologies.

B. Reverse Engineering Mobile Application

Computer vision and optical character recognition (OCR) techniques are used in REMAUI to identify various elements such as images, texts, containers and lists on a given input. 448 screenshots of iOS and Android applications were used in this system. User interfaces generated by REMAUI were similar to the original screenshots.

C. Machine Learning Based Prototyping of Graphical User Interfaces for Mobile Apps

This system is used to transform a GUI into code. Initially, components of a GUI were detected from an artifact. Then, GUI components were accurately classified into domain specific types (e.g., toggle-button) using automated dynamic analysis, software repository mining and deep convolutional neural networks. Finally, prototype application can be combined from a GUI structure generated using K-nearest neighbors algorithm. This was implemented for Android system.

An algorithm named Reverse Engineering Mobile Application User Interfaces (REMAUI) finds the elements of the UI a mobile application for example buttons, text-boxes and pictures, it makes the code for them from the screenshots of an application window. It transforms to the code from the screen pictures or drawings for mobile platforms, PC vision and optical character acknowledgment techniques are utilized. Despite the fact that the REMAUI strategy works effectively, it doesn't assist cross-page change and animations inside the page. Creators built up the P2A algorithm to cure the lacks of the REMAUI calculation. Creators built up the pix2code algorithm which expects to change over the graphical interface for a website page to structured code using deep learning with convolution and repetitive neural networks.

The algorithm Redraw takes mock-ups of mobile application screens and makes an organized XML code for it. In the first phase of their implementation PC vision procedures are utilized to distinguish singular GUI components. The second phase includes the order of the recognized components as indicated by their function, for example, toggle button, text area, etc. Right now, convolution neural networks are utilized. In the last phase, the XML code is produced by joining the K-Nearest Neighbor's (KNN) algorithm as indicated by the web programming hierarchy.

These days open-source code libraries, for example, GitHub are utilized very common to share code and applications. It is a typical practice to explore this repository and reuse code when beginning or improving programming ventures. The common codes in these libraries lessen a similar code being composed again and again by various individuals.

The creators utilize an inquiry program called SUISE in which the clients characterize a graphical interface with straightforward drawings and keywords. This interface is then finding in existing libraries to get comparative interfaces. These interfaces are transformed into operable codes and came back to the end client to choose the most reasonable interface

The project involved analyzing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigations from one screen to the other well-ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the browser version had to be chosen so that it is compatible with most of the Browsers. Machine Learning is being used in almost every type of industry Machines can do Predictive Analysis such as classification & regression (predicting numerical values) and tasks like driving car which require some kind of intelligence.

Machine Learning is part of Artificial Intelligence, in which we provide data to the machine so that it can learn pattern from the data and it will be able to predict solution for similar future problems. **Neural Network (NN)** is inspired by neural network of the human brain. **Computer Vision** is a field of Artificial Intelligence which focuses on problems related to images. CNN combined with Computer Vision is capable of performing complex operations ranging from classifying images to solving scientific problems of astronomy and building self-driving cars.

CHAPTER-3 DOMAIN SPECIFICATIONS.

INPUT DESIGN

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

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow.

OUTPUT DESIGN

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

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

2. Select methods for presenting information.

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

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

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

SYSTEM COMPONENTS

Input: Upload data files: Hand drawn mock-up images resembling the web page are upload in the form of files.

Output: HTML code is generated by training the images then a webpage is created accordingly.

Key Technology: Programming Languages: Python

UML Diagrams: Star UML

Implementation: Image files are uploaded in the form of csv (comma separated value) files. After reading the input file, it is converted to gray scale format. In this way, the components in the input image have been detected then cropped.CNN is applied on the cropped images for classification of images. Recognized components were successfully translated into HTML code via the bootstrap framework.

MODULES

Upload Datasets Files

Users are allowed to upload the image. Once the file is uploaded, in SQL Alchemy ORM, the Object Relational Mapper is introduced and fully described. If you want to work with higher-level SQL which is constructed automatically create engine stored the database.

Object Detection and Cropping

After reading the input file, it is converted to gray scale format. Then, Gaussian Blur was applied 2 times to them with 3x3 rectangle kernel. After the threshold process was carried out, rectangle was drawn by applying the contour detection algorithm to determine the objects by applying morphological transformations. In this way, the components in the input image have been detected. The detected components were cropped to be transferred to the CNN model.

In the stages of morphological transformations, 8 iteration dilation was performed with 4x4 rectangle kernel. Then erosion process was applied with 3x3 ellipse kernel for 4 iterations and dilation process was performed with 1x10 rectangle shaped kernel for 2 iterations. Finally, a 10x1 rectangle shaped kernel was used for dilation process.

Object Recognition

The model was trained with the elements in our component dataset. As mentioned before, it consists of four different types of components such as textbox, dropdown, button and checkbox. After the stage of training the model, the loss function was trained for 200 epochs using Binary Crossentropy and RMSProp algorithms by setting the batch size to 64. Afterwards, component recognition process was carried out by giving the cropped components that came from the previous stage as input.

As seen in our CNN Model we put several convolution layers with 4x4 kernels and then we applied max pooling processes with 2x2 kernels for the feature extraction purpose. After the process that we call as vectorization of the features we put BiLSTM layer for catching correlation of the extracted features. After all, we put Full Connected Layers and Dropout layers with the ratio of 20% in order to achieve the objective of classification.

HTML Builder

Recognized components were successfully translated into HTML code via the bootstrap framework. It was performed with the help of the coordinates from the result of the contour finding algorithms. Demonstrates the latest output that obtained from a browser when the input image is the first one of the images.

As seen in the HTML builder algorithm first we created the templates for a header and a footer. Second, we detected how many items there are on each of the rows with coordinates of the components. Then we mapped the labels of the components to their template codes. At the end of this process the body section of the HTML code was successfully obtained. Finally, the header, body and footer sections were combined as well. So, the final HTML code was composed.

MODEL DIAGRAM

A model is an abstraction that contains all the elements needed to describe the intention of the thing being modeled. This can include all aspects concerning the business, systems, relationships, and more. A diagram is a specific view into what you are trying to understand in a specific context.

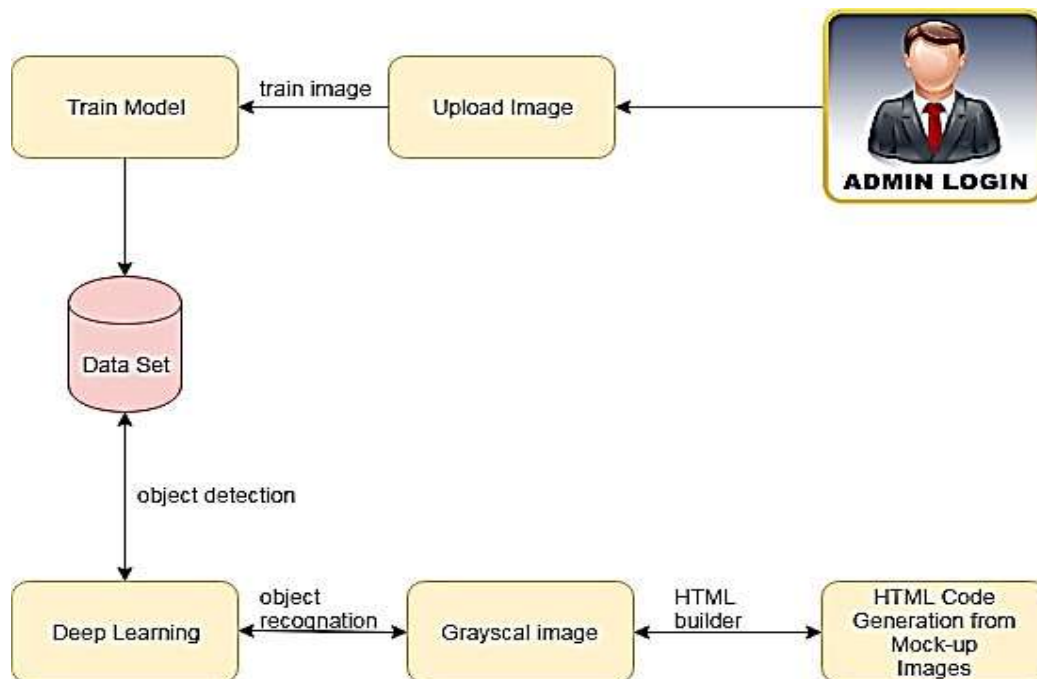


Fig 3.1 Structure of automatic generating of HTML code

The user can upload the hand drawn mock image of the webpage that he is desired

into the system through GUI. Then the uploaded image will be inserted into the database and it is trained accordingly. The trained model is searched in the dataset by testing against the test set that is on which training set the model is trained. Thus the object detection is done using Convolution neural networks. These images are converted to different forms using grayscale conversion and then all these images are cropped and using CNN the components are detected.

The detected components are inserted to an HTML builder as inputs and the corresponding code is generated so as the webpage

DATA FLOW DIAGRAM

Data flow diagram is graphical tool used to describe and analyze the movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data diagrams show the actual implements and movement of data between people departments and workstations. A full description of a system actually consists of a set of data flow diagrams.

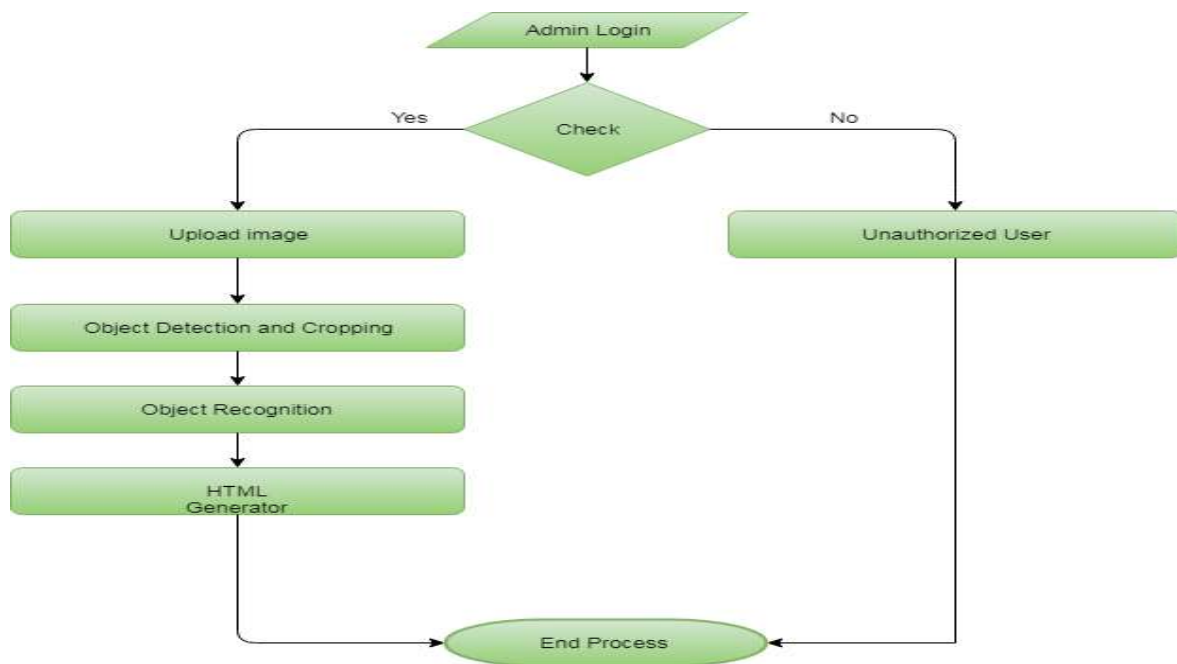


Fig 3.2: Data Flow Diagram of Admin

UML DIAGRAMS

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML stands for Unified Modeling Language. UML is a pictorial language used to make software blueprints. It can be described as a general-purpose visual modeling language to visualize, specify, construct, and document software system. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design. Some of the most important UML diagrams are as follows:

Use case diagram:

Use case diagrams are a set of use cases, actors, and their relationships. They represent the use case view of a system. It represents a particular functionality of a system. Hence, use case diagram is used to describe the relationships among the functionalities and their internal/external controllers. These controllers are known as actors. The use case diagram is used to represent all the functional use cases that are involved in the project.

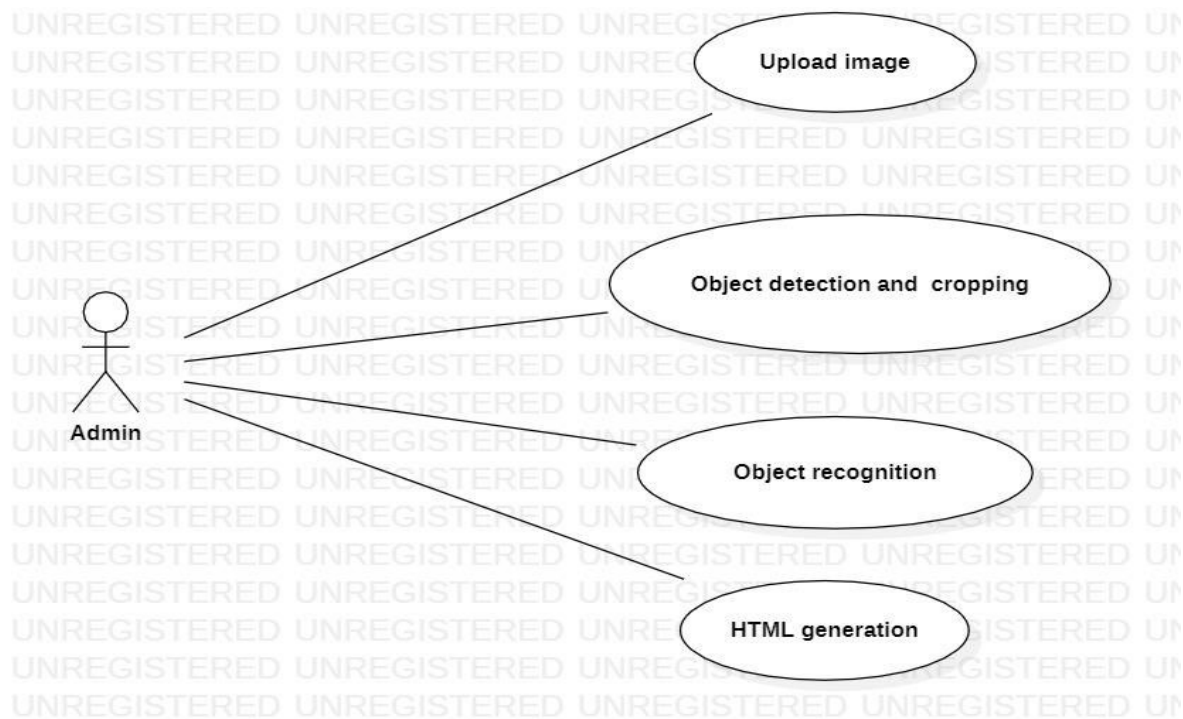


Fig 3.3: Use Case Diagram

Fig 3.3 shows the use case diagram, the actor is admin and he can upload image, detect the objects and crop them for further object detection. Then after detection of the components, HTML code can be generated using HTML builder.

Class diagrams:

Class diagrams are the most common diagrams used in UML. Class diagram consists of classes, interfaces, associations, and collaboration. Class diagrams basically represent the object-oriented view of a system, which is static in nature. Active class is used in a class diagram to represent the concurrency of the system. Class diagram represents the object orientation of a system. Hence, it is generally used for development purpose. This is the most widely used diagram at the time of system construction.

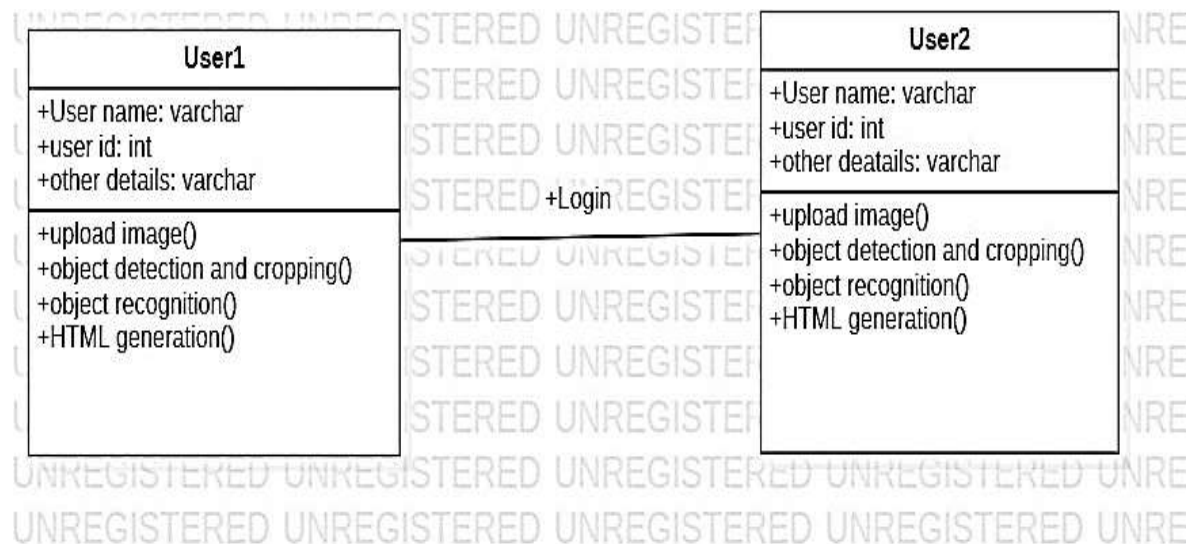


Fig 3.4: Class Diagram

Fig 3.4 shows the class diagram, we have user classes which can login to the system and possess the attributes as username, userid and other details. The operations will be upload image, object detection and cropping, object recognition and HTML generation.

Activity Diagram:

Activity diagram describes the flow of control in a system. It consists of activities and links. The flow can be sequential, concurrent, or branched. Activities are nothing but the functions of a system. Numbers of activity diagrams are prepared to capture the entire flow in a system.

Activity diagrams are used to visualize the flow of controls in a system. This is prepared to have an idea of how the system will work when executed.

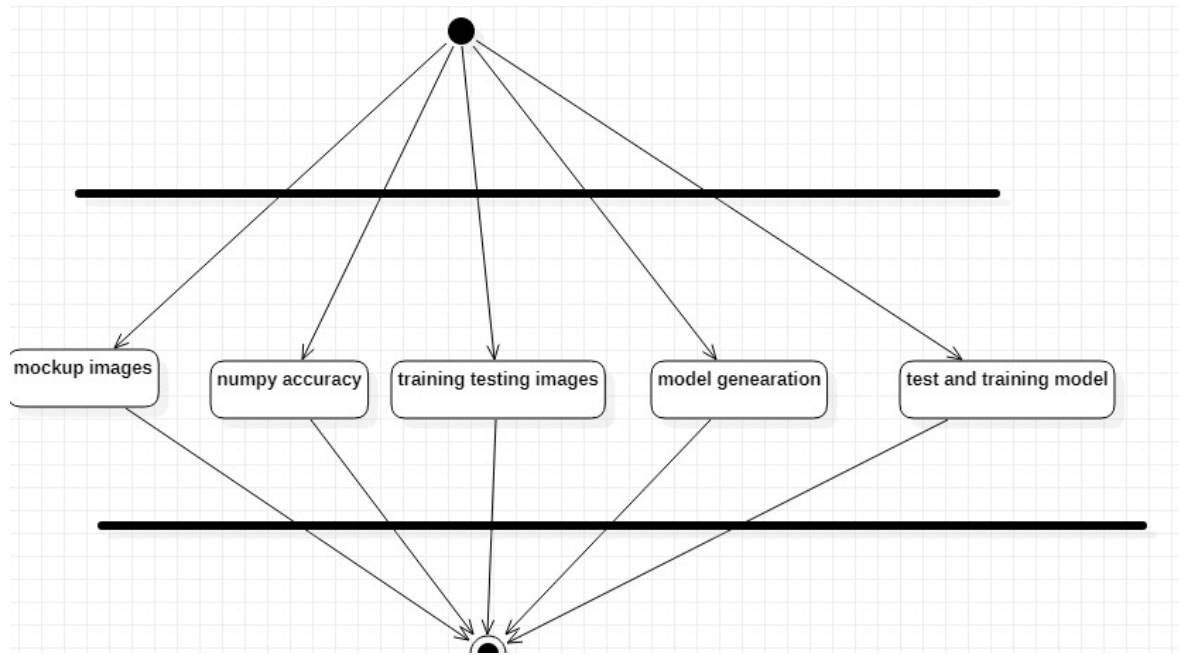


Fig 3.5: Activity Diagram

Fig 3.5 shows the activity diagram, at any instance whether there is any error occurred then the fork is applied that is the user will get stopped of his execution process.

Sequence diagram:

A sequence diagram is an interaction diagram. From the name, it is clear that the diagram deals with some sequences, which are the sequence of messages flowing from one object to another. Interaction among the components of a system is very important from implementation and execution perspective. Sequence diagram is used to visualize the sequence of calls in a system to perform a specific functionality.

state change of a class, interface, etc.

State chart diagram is used to visualize the reaction of a system by internal/external factors.

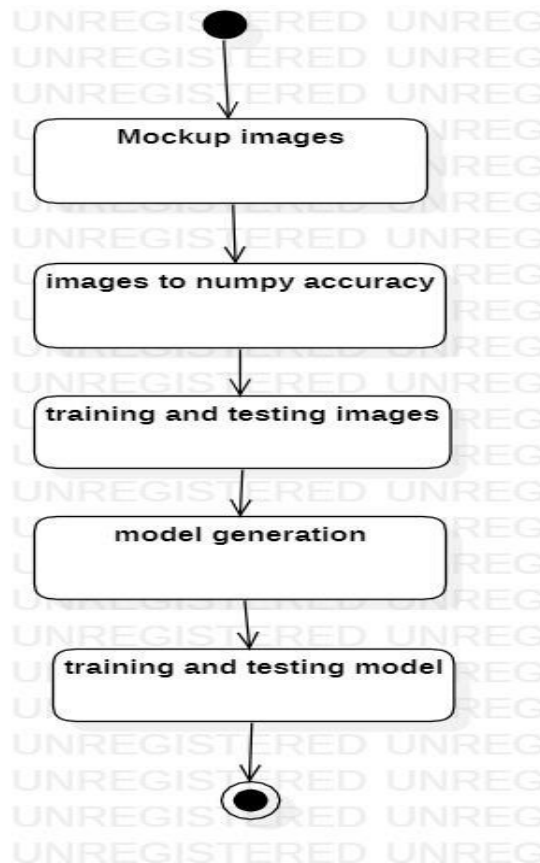


Fig 3.7: State chart diagram

Fig 3.7 shows the changes in the states taking place. Here the changes are first the image is uploaded then the image are tested for accuracy then trained against the training set. The model is generated and then tested against train set.

ER diagram

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.

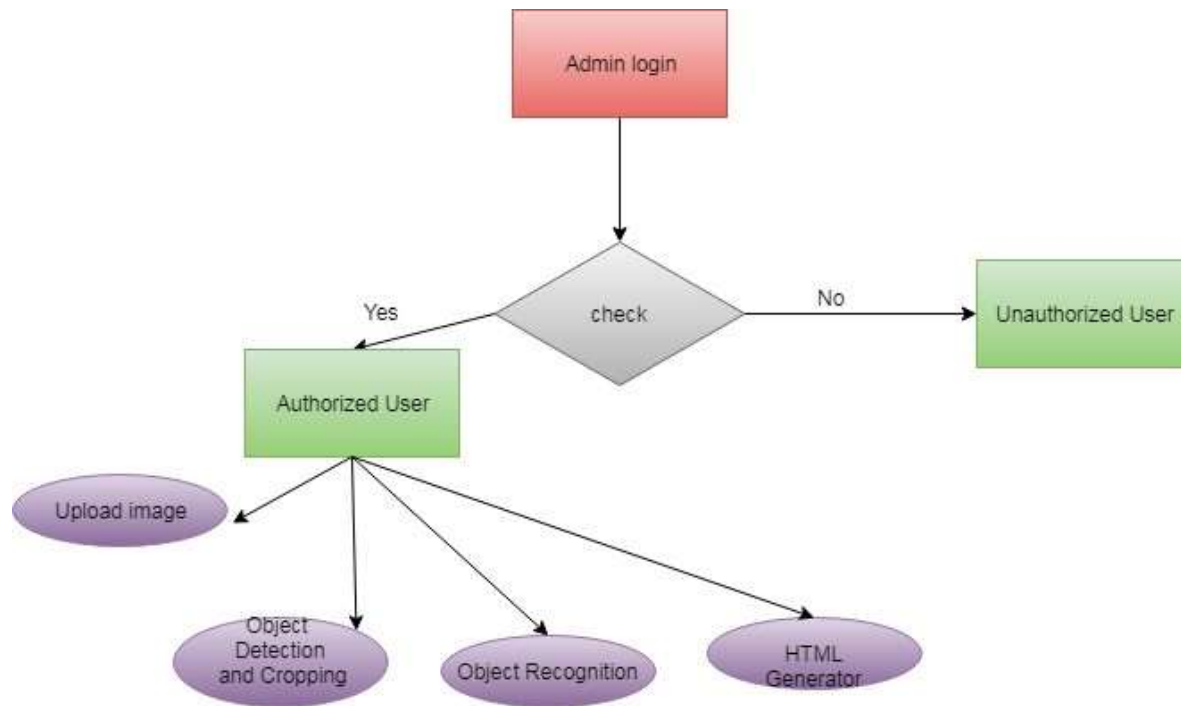


Fig 3.8: ER diagram

Fig 3.8 shows the ER diagram, the admin logs in to the system and it will be checked whether the user is an authorized user or not. If he is an authorized user then he would be able to do the tasks such as uploading images that are hand-drawn and object detection and cropping and he would be able to generate the HTML code.

Component diagram

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities.

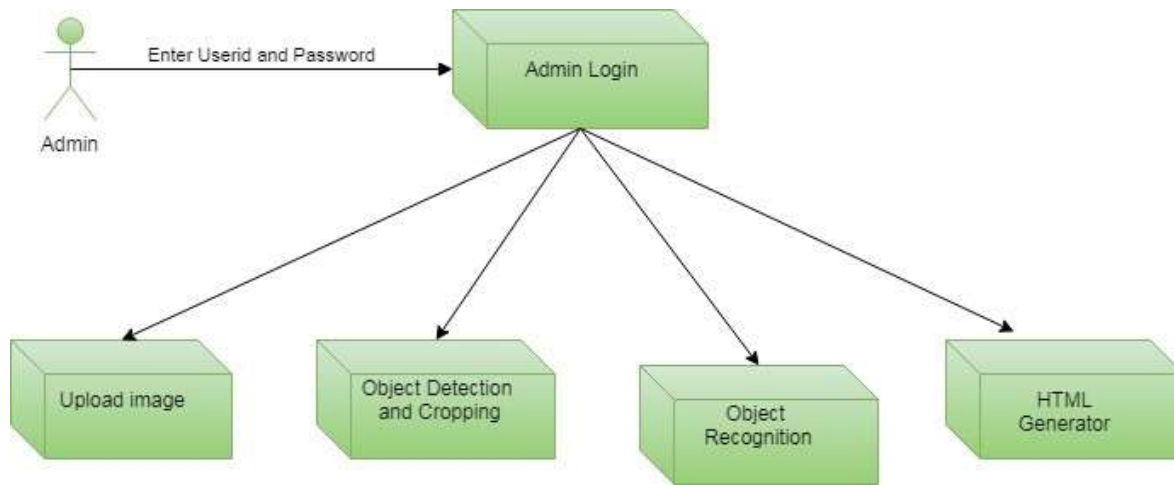


Fig 3.9: Component diagram

Fig 3.9 shows the nodes corresponding the actions of the user. He would be able to upload the mock-up image and then using the inbuilt mechanisms he would be able to create the HTML code that automatically generates a webpage.

DATA DICTIONARY

A data dictionary contains metadata i.e., data about the database. The data dictionary contains the information such as what is in the database, who is allowed to access it, where is the database physically stored, etc. The users of the database normally don't interact with the data dictionary, it is only handled by the database administrators.

Data Dictionary consists of the following information

Name of the tables in the database.

Constraints of a table i.e., keys, relationships, etc.

Columns of the tables that related to each other.

Owner of the table.

Last accessed information of the object.

Last updated information of the object.

Table: 3.1 Login authentication

Field name	Data type	Field length	Constraints	Extra
user id	varchar	200	Primary key	autoincrement
first name	varchar	200	Not null	-
last name	varchar	200	Not null	-
email	varchar	200	Not null	-
password	varchar	200	Not null	-
password confirmation	varchar	200	Not null	-

Table: 3.2 Upload image

Field name	Data type	Field length	Constraints	Extra
check_imgid	varchar	200	foreign key	-
check_tile	varchar	200	-	-
check_userid	varchar	200	foreign key	-
check_des	varchar	500	-	-
txt_filed	varchar	1000	-	-
cate_list	varchar	200	-	-

CHAPTER-4 METHODOLOGY

SYSTEM REQUIREMENTS

A System Requirements Specification is a structured collection of information that embodies the requirements of a system. All computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

Functional requirements

A functional requirement defines a function of a system or its component, where a function is described as a specification of behavior between outputs and inputs. Functional requirements may involve calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is supposed to accomplish. Functional requirements are supported by non-functional requirements. These also known as "quality requirements", which impose constraints on the design or implementation. Functional requirements specify particular results of a system.

Functional requirements are features that allow the system to function as it was intended. Put another way, if the functional requirements are not met, the system will not work. Functional requirements are product features and focus on user requirements. It defines the functionality of a system or one of its subsystems. It also depends upon the type of software, expected users and the type of system where the software is used.

Graphical User Interface: GUI stands for Graphical User Interface. It refers to an interface that allows one to interact with electronic devices like computers and tablets through graphic elements. It uses icons, menus and other graphical representations to display information, as opposed to text-based commands. The graphic elements enable users to give commands to the computer and select functions by using mouse or other input devices.

Non-Functional requirements

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behavior. In systems engineering and requirements engineering, a non-functional requirement (NFR) is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behavior. They are contrasted with functional requirements that define specific behavior or functions. The plan for implementing functional requirements is detailed in the system design. The plan for implementing non-functional requirements is detailed in the system architecture, because they are usually architecturally significant requirements.

Non-functional requirements are often called "quality attributes" of a system however there is a distinction between the two. Non-functional requirements are the criteria for evaluating how a software system should perform and a software system must have certain quality attributes in order to meet non-functional requirements. So, when we say a system should be "secure", "highly-available", "portable", "and scalable" and so on, we are talking about its quality attributes. Other terms for non-functional requirements are "qualities", "quality goals", "quality of service requirements", "constraints", "non-behavioral requirements", or "technical requirements". Informally these are sometimes called the "ilities", from attributes like stability and portability. Qualities that is non-functional requirements can be divided into two main categories:

Execution qualities, such as safety, security and usability, which are observable during operation (at run time).

Evolution qualities, such as testability, maintainability, extensibility and scalability, which are embodied in the static structure of the system.

Scalability:

Scalability requirements describe how the system must grow without negative influence on its performance. This means serving more users, processing more data, and doing more transactions. Scalability has both hardware and software implications. For instance, you can increase scalability by adding memory, servers, or disk space. On the other hand, you can compress data, use optimizing algorithms, etc.

Performance:

System performance is the most important quality in non-functional requirements and affects almost all the other preceding ones. System performance defines how fast a system can respond to a particular user's action under a certain workload.

Reliability:

Reliability defines how likely it is for the software to work without failure for a given period of time. Reliability decreases because of bugs in the code, hardware failures, or problems with other system components. To measure software reliability, you can count the percentage of operations that are completed correctly or track the average period of time the system runs before failing.

Usability:

Usability defines how difficult it will be for a user to learn and operate the system. Usability can be accessed from different points of view: Efficiency of use: the average time it takes to accomplish a user's goals, how many tasks a user can complete without any help, the number of transactions completed without errors, etc.

Hardware Requirements

Processor: Pentium IV or higher

RAM: 1 GB

Space on Hard Disk: Minimum 1GB

Software Requirements

Python IDLE (3.7 version)

Django (3.1.5 version)

MySQL (1.3.12 version)

Operating system supported: Windows 7 and higher versions

Technologies and languages used: Python (Version-3.9.1)

Debugger and emulator - Any browser (particularly chrome)

Python: Python is an interpreted, high-level and general-purpose programming language. Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are optional. It has fewer syntactic exceptions and special cases than C or Pascal.

Django: Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and "plug ability" of components, less code, low coupling, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

MySQL: A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

Operating Systems: An Operating System (OS) is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers. An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Windows: Windows is an operating system designed by Microsoft. The operating system is what allows you to use a computer. Windows makes it possible to complete all types of everyday tasks on your computer.

IMPLEMENTATION

Implementation is carrying out, execution, or practice of a plan, a method, or any design, idea, model, specification, standard or policy for doing something. Implementation is the action that must follow any preliminary thinking in order for something actually to happen.

Convolutional Neural Networks

In deep learning, a convolutional neural network is a class of deep neural network, most commonly applied to analyze visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation equivariant responses known as feature maps. Counter-intuitively, most convolutional neural networks are only equivariant, as opposed to invariant, to translation. They have applications in image and video recognition, recommender systems, image classification, image segmentation.

In CNN, each input image will pass through a sequence of convolution layers along with pooling, fully connected layers, filters (Also known as kernels). After that, we will apply the Soft-max function to classify an object with probabilistic values 0 and 1. These machines learn to process image? It uses Convolution Neural Network for doing this task effectively.

Generally, a Convolutional Neural Network has three layers, which are as follows:

Input: If the image consists of 32 widths, 32 height encompassing three R, G, B channels, then it will hold the raw pixel ($[32 \times 32 \times 3]$) values of an image.

Convolution: It computes the output of those neurons, which are associated with input's local regions, such that each neuron will calculate a dot product in between weights and a small region to which they are actually linked to in the input volume. For example, if we choose to incorporate 12 filters, then it will result in a volume of $[32 \times 32 \times 12]$.

ReLU Layer: It is specially used to apply an activation function elementwise, like as max(0, x) thresholding at zero. It results in ($[32 \times 32 \times 12]$), which relates to an unchanged size of the volume.

Pooling: This layer is used to perform a down sampling operation along the spatial dimensions (width, height) that results in $[16 \times 16 \times 12]$ volume.

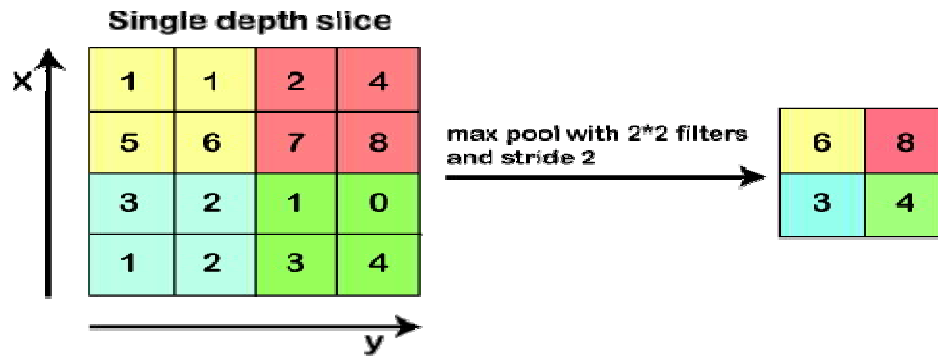


Fig 4.1: Convolution layer

Locally Connected: It can be defined as a regular neural network layer that receives an input from the preceding layer followed by computing the class scores and results in a 1-Dimensional array that has the equal size to that of the number of classes. We will start with an input image to which we will be applying multiple feature detectors, which are also called as filters to create the feature maps that comprises of a Convolution layer. Then on the top of that layer, we will be applying the ReLU or Rectified Linear Unit to remove any linearity or increase non-linearity in our images.

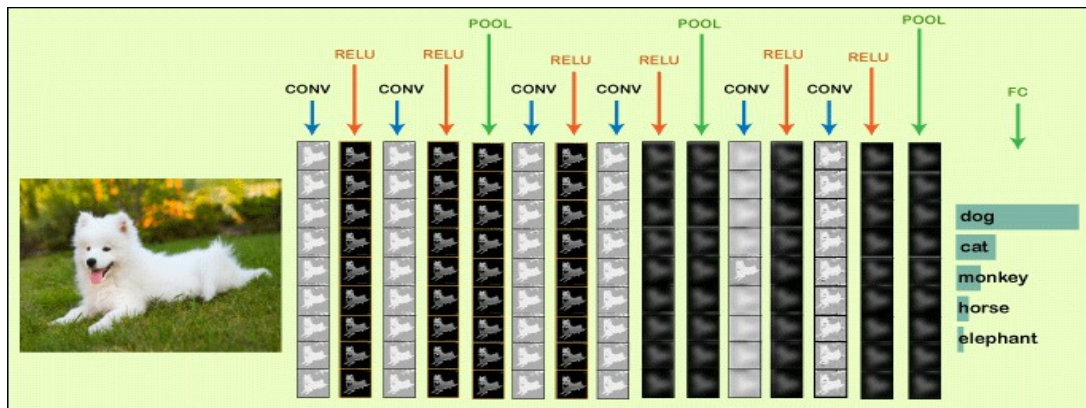


Fig 4.2: Applying the ReLU or Rectified Linear Unit

Next, we will apply a Pooling layer to our Convolutional layer, so that from every feature map we create a Pooled feature map as the main purpose of the pooling layer is to make sure that we have spatial invariance in our images. It also helps to reduce the size of our images as well as avoid any kind of overfitting of our data. After that, we will flatten all of our pooled images into one long vector or column of all of these values, followed by inputting these values into our artificial neural network. Lastly, we will feed it into the locally connected layer to achieve output.

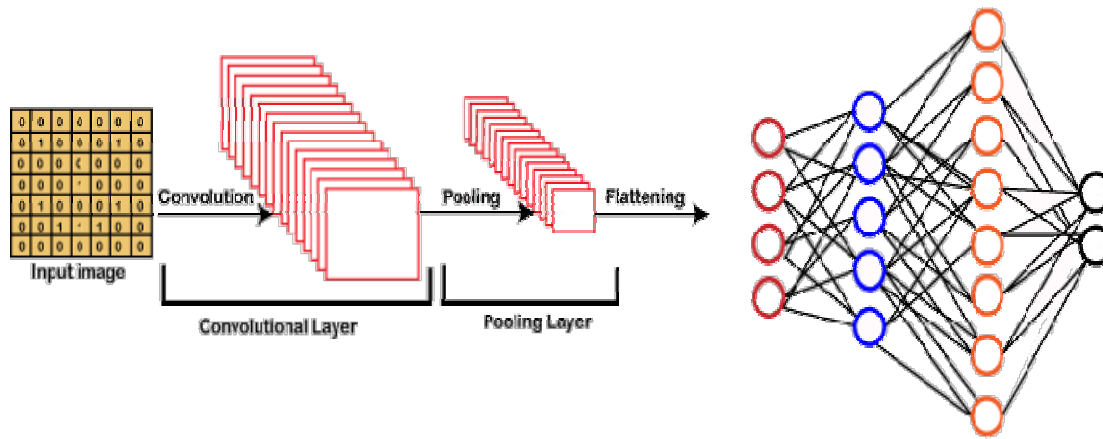


Fig 4.3: Convolutional neural network

SAMPLE CODE

#Registration code

```
from django import forms
```

```
from compiler.models import RegisterModel
```

```
class Registerform(forms.ModelForm):
```

```
    class Meta:
```

```
        model=RegisterModel
```

```
        fields = ("first_name", "last_name", "email", "password", "password_confirmation")
```

#Creating models code

```
from django.db import models
```

```
from tkinter import CASCADE
```

```
# Create your models here.
```

```
class RegisterModel(models.Model):
```

```

first_name=models.CharField(max_length=300)

last_name=models.CharField(max_length=200)

userid = models.CharField(max_length=200)

email=models.CharField(max_length=200)

password=models.CharField(max_length=200)

password_confirmation=models.CharField(max_length=200)

class UploadModel(models.Model):

    image=models.FileField()

class CheckModel(models.Model):

    check_tile=models.CharField(max_length=200)

    check_des=models.CharField(max_length=500)

    check_imgid=models.ForeignKey(UploadModel)

    img_path=models.CharField(max_length=200)

    check_userid=models.ForeignKey(RegisterModel)

    txt_filed=models.CharField(max_length=1000)

    cate_list=models.CharField(max_length=200)

class Html_Page(models.Model):

    htmlfile= models.FileField()

```

#Creating views code

```

from __future__ import print_function

import os

from os.path import basename

```

```
from compiler.classes.Compiler import Compiler

from compiler.classes.Utills import Utills

import cv2

import sys

from django.core.files.storage import FileSystemStorage

from django.shortcuts import render, redirect

from django.views.decorators.csrf import csrf_exempt

from django.http import JsonResponse

import numpy as np

import urllib

import json

import os

import requests

from tkinter import *

from PIL import Image

from PIL import ImageTk

from tkinter import filedialog

import tkinter.filedialog

import cv2

from skimage import exposure

import pickle

import sys
```

```

import imutils

from matplotlib import pyplot as plt

from skimage.measure import compare_ssim

from scipy import linalg

import numpy

import argparse

# Create your views here.

from code_generation.settings import BASE_DIR

from compiler.models import RegisterModel, UploadModel, CheckModel

def login(request):

    if request.method == "POST":

        email = request.POST.get('email')

        password = request.POST.get('password')

        try:

            check = RegisterModel.objects.get(email=email, password=password)

            request.session['userid'] = check.id

            return redirect('upload_page')

        except:

            pass

    return render(request, 'compiler/login.html')

def register(request):

    if request.method == "POST":

```

```

first_name = request.POST.get('first_name')

last_name = request.POST.get('last_name')

userid = request.POST.get('userid')

email = request.POST.get('email')

password = request.POST.get('password')

password_confirmation = request.POST.get('password_confirmation')

RegisterModel.objects.create(first_name=first_name,
last_name=last_name,userid=userid,email= email,
password=password,password_confirmation=password_confirmation )

return render(request,'compiler/register.html')

def upload_page(request):

    myfile = "

    if request.method == "POST" and request.FILES['myfile']:

        myfile = request.FILES['myfile']

        UploadModel.objects.create(image=myfile)

    return render(request,'compiler/upload_page.html')

def viewresult(request):

    def select_image1():

        # grab a reference to the image panels

        global panelA, panelB

        # open a file chooser dialog and allow the user to select an input

        # image

        path = filedialog.askopenfilename()

```



```

input_file = "

input_file = path

FILL_WITH_RANDOM_TEXT = True

TEXT_PLACEHOLDER = "[]"

dsl_path = "compiler/assets/web-dsl-mapping.json"

compiler = Compiler(dsl_path)

def render_content_with_text(key, value):

    if FILL_WITH_RANDOM_TEXT:

        if key.find("btn") != -1:

            value = value.replace(TEXT_PLACEHOLDER, Utils.get_random_text())

        elif key.find("title") != -1:

            value = value.replace(TEXT_PLACEHOLDER, Utils.get_random_text(length_text=5,
space_number=0))

        elif key.find("text") != -1:

            value = value.replace(TEXT_PLACEHOLDER,

                                Utils.get_random_text(length_text=56, space_number=7,
with_upper_case=False))

    return value

file_uid = basename(input_file)[:basename(input_file).find(".")]

print(file_uid)

path = input_file[:input_file.find(file_uid)]

print(path)

input_file_path = "{}{}.gui".format(path, file_uid)

```

```

print(input_file_path)

output_file_path = "{}{}.html".format(path, file_uid)

ted = output_file_path

print(ted)

compiler.compile(input_file_path, output_file_path,
rendering_function=render_content_with_text)

# initialize the window toolkit along with the two image panels

root = Tk()

panelA = None

panelB = None

# create a button, then when pressed, will trigger a file chooser

# dialog and allow the user to select an input image; then add the

# button the GUI

btn = Button(root, text="Select an Brain MRI image", command=select_image1)

btn.pack(side="bottom", fill="both", expand="yes", padx="10", pady="10")

root.mainloop()

return render(request, 'compiler/viewresult.html')

```

CHAPTER-5 RESULTS

SPECIFICATIONS:

1. Processor: Pentium IV or higher
2. RAM: 1 GB
3. Space on Hard Disk: Minimum 1GB
4. Python IDLE (3.7 version)
5. Django (3.1.5 version)
6. MySQL (1.3.12 version)
7. Operating system supported: Windows 7 and higher versions
8. Technologies and languages used: Python (Version-3.9.1)

LOGIN PAGE OF THE WEBSITE

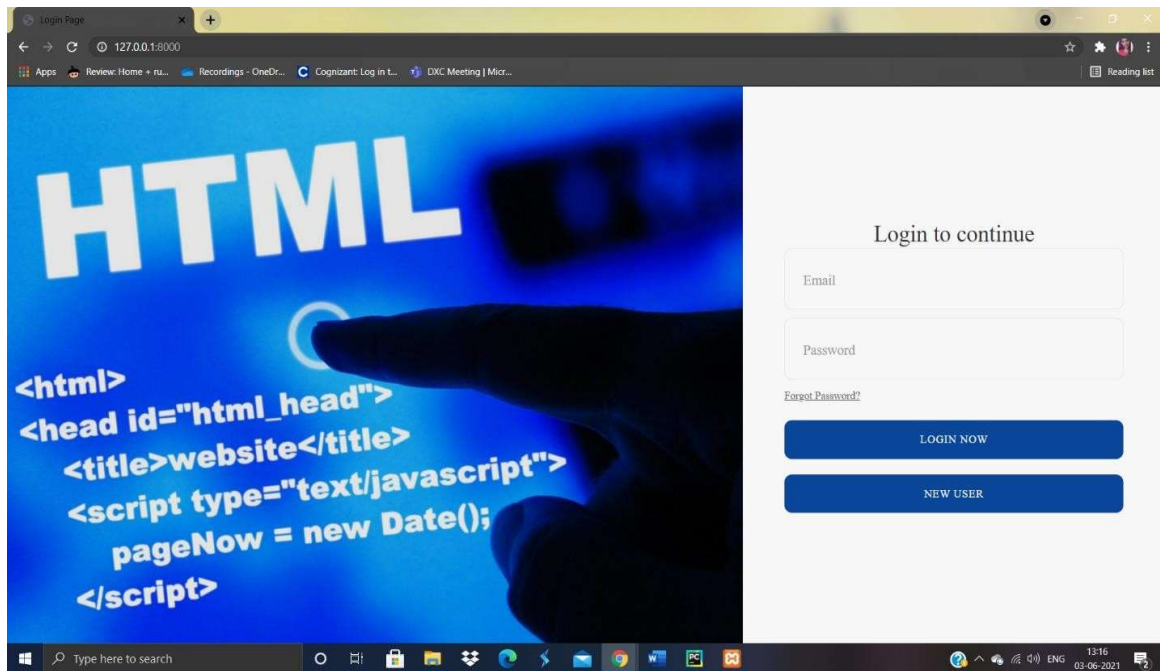
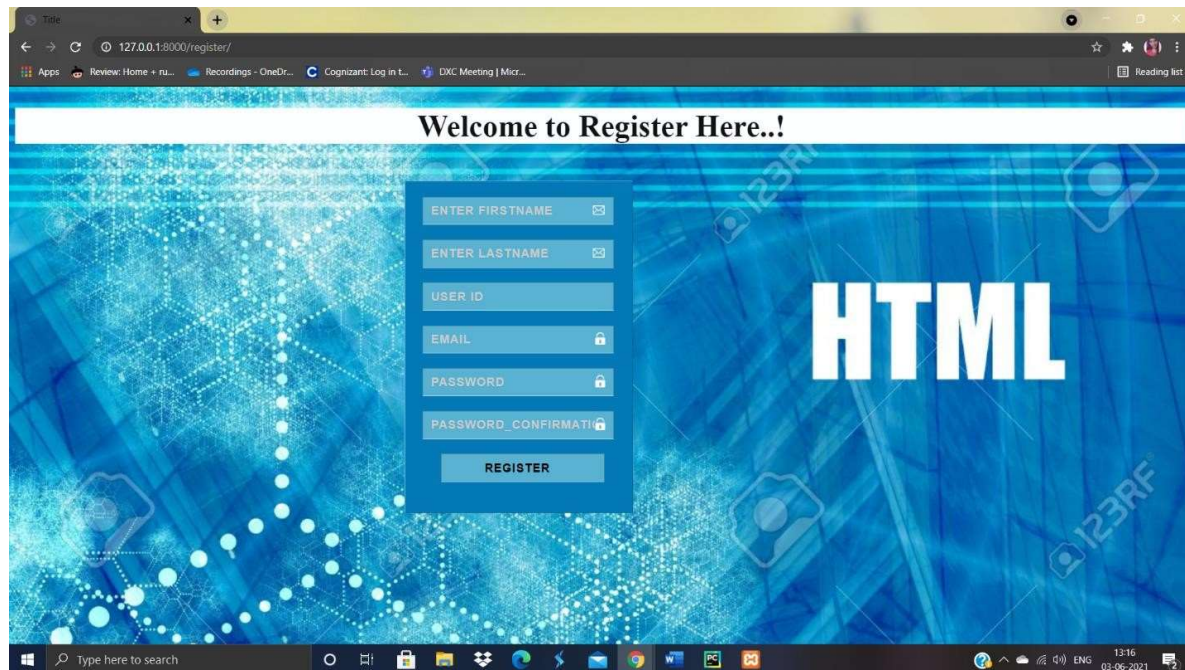


Fig 5.1: Login page

Figure 5.1 appears as the front page of the website. This enables a user to login to the system and perform his desired tasks. If a user is a new user, then he should click on the new user option then he can click on the login now option for login.

REGISTRATION PAGE

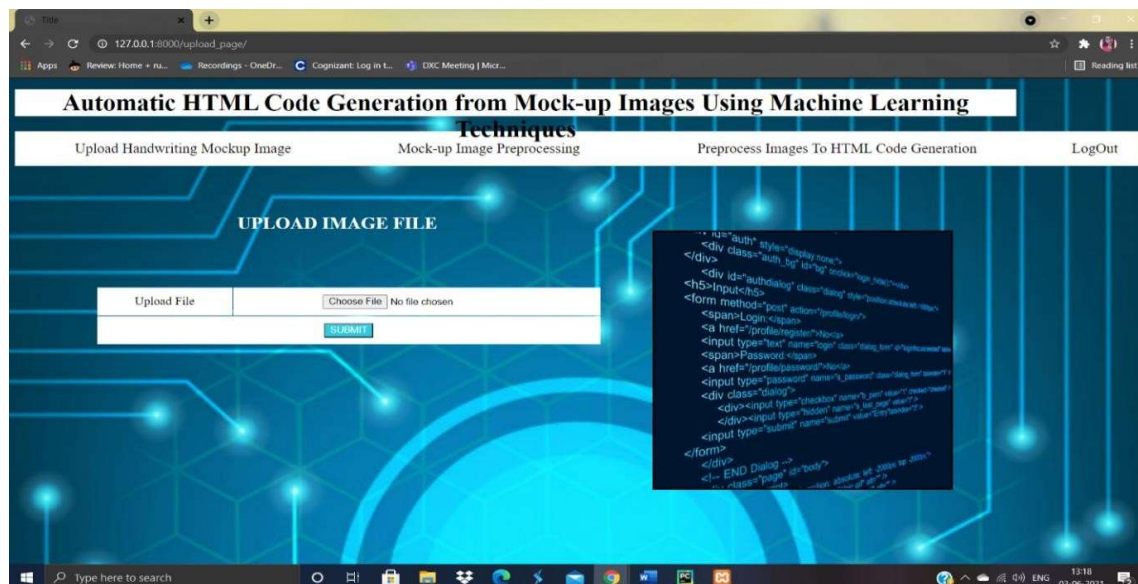


The screenshot shows a web browser window with the URL `127.0.0.1:8000/register/`. The page has a blue background with a network-like pattern. At the top, a white banner reads "Welcome to Register Here..!". Below this, a registration form is centered. The form contains the following fields: "ENTER FIRSTNAME", "ENTER LASTNAME", "USER ID", "EMAIL", "PASSWORD", and "PASSWORD_CONFIRMATION". Each field has a corresponding icon (e.g., a person for first/last name, a key for password). A "REGISTER" button is at the bottom of the form. To the right of the form, the word "HTML" is displayed in large, white, bold letters. The browser's taskbar at the bottom shows various application icons and the system clock indicating 13:16 on 03-06-2021.

Fig 5.2: Registration page

If the user clicks on “Register now”, then the above page (fig 7.2) appears. This enables the user to register. Once the user completes the registration, he can now login to the system.

UPLOAD PAGE



The screenshot shows a web browser window with the URL `127.0.0.1:8000/upload_page/`. The page has a blue background with a network-like pattern. At the top, a white banner reads "Automatic HTML Code Generation from Mock-up Images Using Machine Learning". Below this, a navigation bar contains the following links: "Upload Handwriting Mockup Image", "Mock-up Image Preprocessing", "Preprocess Images To HTML Code Generation", and "LogOut". The main content area is titled "UPLOAD IMAGE FILE". It features a file upload interface with a "Choose File" button and a "SUMMIT" button. To the right of the upload area, a code editor displays HTML code for a login form, including fields for "username" and "password", and a "login" button. The browser's taskbar at the bottom shows various application icons and the system clock indicating 13:18 on 03-06-2021.

Fig 5.3: Upload page

Figure 5.3 appears once the user login is authenticated. This page enables the user to upload handwritten mock up images, image preprocessing and code generation. The user can logout of the system once he completes execution.

UPLOAD IMAGE

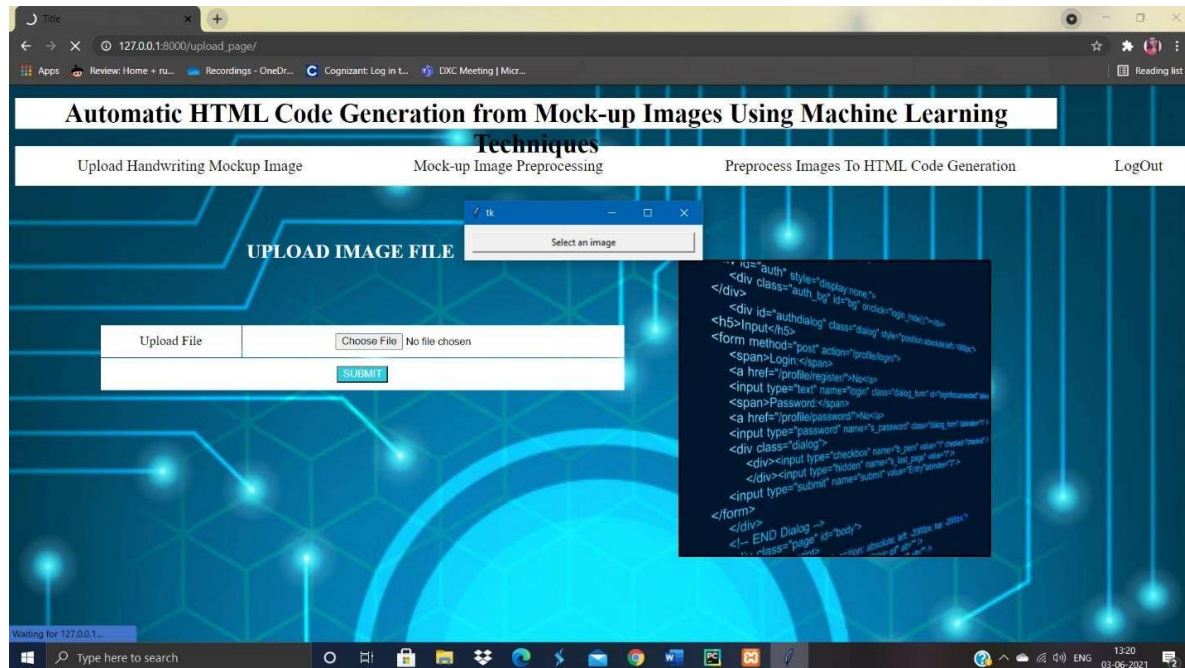


Fig 5.4: Upload image

Figure 5.4 appears when the user clicks on upload handwritten images. The user needs to upload a file that should be converted to web page.

IMAGE PROCESSING PAGE

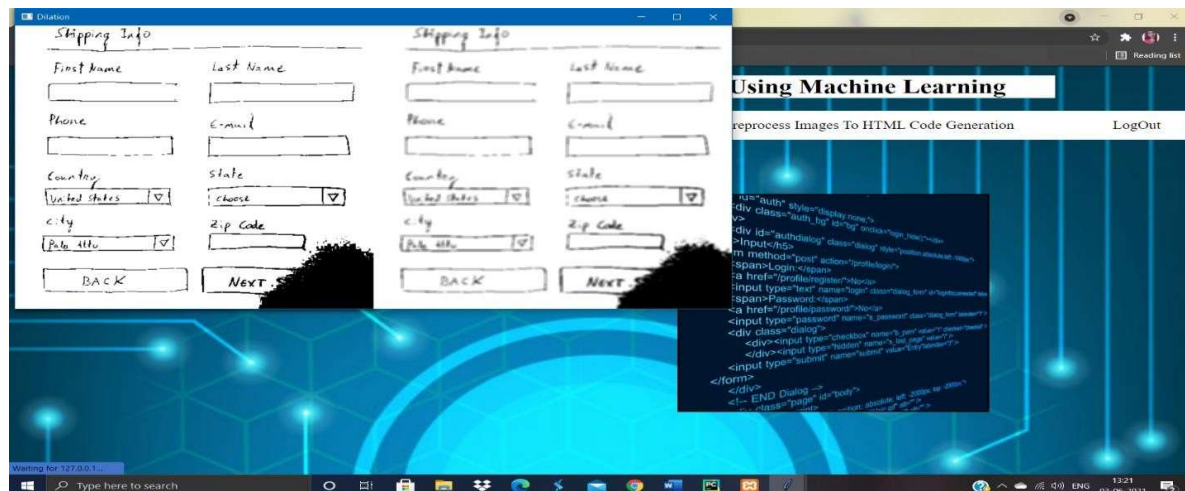


Fig 5.5: Image processing page

WEB PAGE CREATION FROM MOCKUP IMAGE

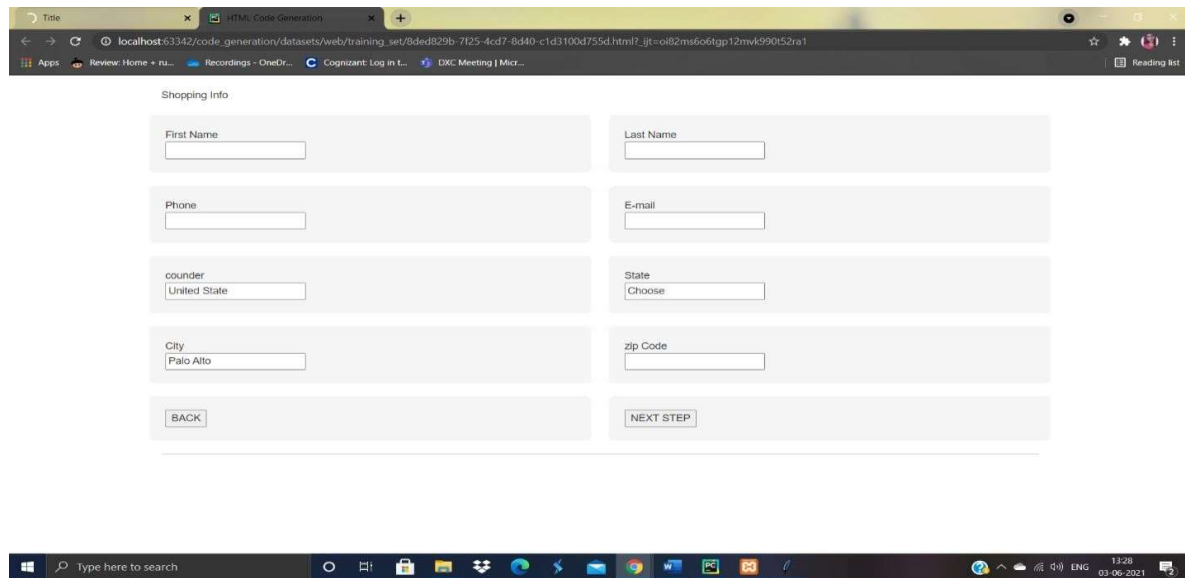


Fig 5.8: Web page created from mockup image

Figure 5.8 is the output of the HTML code that is generated using hand-drawn mockup images.

CHAPTER-6 ADVANTAGES AND DISADVANTAGES

ADVANTAGES

Permits small scale organizations to concentrate more on components, esteem and less on making an interpretation of structures into useful application code.

Automatic creation of web pages decreases programming process, cost and resource utilization

Reduce the time taken and wasted in rewriting the code for components with similar functions and page structures that change over time.

DISADVANTAGES

When the user wants to add an element to his page that is not a part of the training set then, that element may not be detected by the system.

This could be overcome by training the system with more possible components.

APPLICATIONS

It will be easier for the programmers and designers to work smartly by utilizing the automatic system for generating webpage from images which doesn't need any rework.

Any non-programmer will also be able to generate his own desired webpage according to his requirements.

Useful for front-end engineers and originators face with building precise GUIs, this exposes that the need for more effective resolution in web page designing.

CHAPTER-7 CONCLUSIONS

PRESENT WORK

Converting web page mock-ups to their mark-up code with minimum time and labor cost has become a significant topic in recent years when the artificial intelligence has been rapidly revolutionizing the industry by entering almost every field. Hence this system that takes hand-drawn web page mock-ups and gives a structured HTML code will reduce the cost, effort and time required for creating webpages. To that end, a dataset consisting of images containing various hand-drawn sketches of web page designs was used. This dataset, which consists of 186 samples in total, has also been utilized in the creation of a corresponding dataset that contains the components contained in each image. Thus, the dataset, which was created by grouping all the components in 4 different classes, was used as training data for the CNN model to perform the process of object recognition. In this study, the components in the picture were cropped by performing object detection with image processing techniques. It was determined which components were obtained by our trained CNN model. Finally, the purpose of generating HTML code was achieved using our HTML builder script with the help of the coordinates came from the algorithms of contour finding.

FUTURE SCOPE

There are quite few things that can be added in the future work. Though we were able to identify, detect and classify some of the components such as checkboxes, radio buttons, etc. but there are many other components on which the system is not trained. The system has been trained on only few components which are most common. When the user wants to add an element to his page that is not a part of the training set then, that element may not be detected by the system. This could be overcome by training the system with more possible components. So that the user can use his desirable components. Also in this system, the user has to upload a file of the image that has been drawn. In future, this could be enhanced in such a way that the user can directly draw on the screen and the associated webpage would be created automatically.

REFERENCES

- [1] Sketch2code. Microsoft AI Labs. [Online]. Available: <https://github.com/Microsoft/ailab/tree/master/Sketch2Code/model/images>
- [2] T. A. Nguyen and C. Csallner, "Reverse Engineering Mobile Application User Interfaces with REMAUI (T)," in 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, nov 2015, pp.248–259.[Online].Available: <http://ieeexplore.ieee.org/document/7372013/>
- [3] S.Natarajan and C. Csallner,"P2A:AToolfor Converting Pixels to Animated Mobile Application User Interfaces," Proceedings ofthe5th International Conference on Mobile Software Engineering and Systems -MOBILESoft '18, pp. 224–235, 2018. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=3197231.3197249>
- [4] T. Beltramelli, "pix2code: Generating code from a graphical user interface screenshot," CoRR, vol. abs/1705.07962,2017. [Online]. Available: <http://arxiv.org/abs/1705.07962>
- [5] K. P. Moran, C. Bernal-Cardenas, M. Curcio, R. Bonett, and D. Poshyvanyk,"Machine learning-based prototyping of graphical user interfaces for mobile apps," IEEE Transactions on Software Engineering, pp. 1–1, 2018.
- [6][Online]. Available: <https://github.com/>
- [7] S. P. Reiss, Y. Miao, and Q. Xin, "Seeking the user interface," Automated Software Engineering, vol. 25, no. 1, pp. 157–193, mar 2018. [Online].
- [7] S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural computation, 9(8):1735– 1780, 1997.
- [8] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3128–3137, 2015.