JSESM

ISSN:2583-3294

https://doi.org/10.5281/zenodo.7892482

Journal of Scholastic Engineering Science and Management

May 2023, Volume 2, Issue 5, pp: 38-49

"Website Creator by Mock-Up Images Using Machine Learning"

[1]Mrs. Geetha M, [2]Mr. Jayanth R , [3]Mr. K R Tarun Kumar, [4]Mr. Chilamatturu Dileep

[1]Assistant Professor, [2,3,4]Student

[1,2,3,4]Department of Computer Science & Engineering, IMPACT College of Engineering & Applied Sciences, Bengaluru, India

Email: [1]geetham2306@gmail.com, [2]jayanthr26042000@gmail.com, [3]tarunkumarmani9060@gmail.com, [4]darlingdileep5556@gmail.com

ABSTRACT

The first step of designing a website is to build the mock-up images for the particular web pages by operating with the hands or using mock-up developer tools. It is efficiently used for the developer to transfer web pages mock-up to the coding. It's generating the proposed system to create the wireframe to the layout interfaces. There are two techniques mostly used: first is computer vision and second is deep systematic analysis. The automatic code generation is time reducing and cost effective. The design cycle for a website begins with the construction of individual web page mock-ups, which can be done by hand or with the help of graphic design and specialist mock-up production tools. Software programmers next turn the prototype into structured HTML or comparable markup code. This procedure is typically performed several times until the appropriate template is obtained. The goal of this research is to automate the process of creating code from hand-drawn mock-ups. Computer vision techniques are utilised to process hand-drawn mock-ups, and then deep learning approaches are employed to construct the suggested system.

Keywords: Website, Machine Learning, Mock-Up Images

1. INTRODUCTION

Nowadays, the internet is the most important in our day-to-day life. Websites are present in every field. The design cycle for a website opening due to creating mock-ups for separate webpage further away drawing by hand or by drawing in paint designs and intensive mock-up formation apparatus. The mock-up images were then transformed to HTML by a software engineer. This proceeding is recast extra instant as far as the wanted template is not got. Our main target is to equip html code out of hand drawn images. We use convolution neural network computer vision techniques and also deep learning was used for our proposed system. In today's world websites review the institution, hotels, business, people, etc. Websites are used in each and every factor. From education to knowledge, from training to social work. At the front end of every site that concerts with the user. It is actually relevant to give a surface a certain attraction in the user, it is very easy to use, and it has sufficiently advanced attributes. In other way, creating a webpage which gives active response expertly for this it required a very tiring pathway. In the development of webpage, many software engineering developers are working together to design the front view of the webpage. Software designer-built code to design the webpage basis on the draft. The resulting webpage can convert depending on feedback received by the as in user. For the elements built the code with the same feature with page format converting instant turn into the steps is difficult. This emerges from the need for expanding more improved features in a webpage format. The proposal of structuring the webpage by creating automatic code is very interesting as a research subject. Generation of automatic web pages minimises coding instant, step price as well as resource. So, by this way, thanks to the speedy pattern steps, the final website is created in a very less time period. In our survey, methods were used for automatically developing the hand drawn images by generating code for it. Its aim is to observe the factor that creates the hand drawing by encrypting the system in the way of the webpage format. The idea of designing a web page by generating automatic code is gaining importance as a research subject. Automatic production of web pages reduces programming time, process cost and resource consumption. Thanks to the faster progressive design stages, the final web-site is produced in a shorter time. In this study, an algorithm has been developed to automatically generate the HTML code for hand-drawn mock-ups of a web page. It is aimed to recognize the components created in the mock-up drawing and to encode them according to the web page hierarchy A public dataset of hand-drawn images of web sites obtained from Microsoft AI Labs' is used to train and verify the proposed scheme.

2. LITERATURE REVIEW

Automated code generation or automatic programming is a widely researched topic these days. The auto code generation fascinates the young researchers towards this field more than ever. Automatic code generation is the process to generate valid programming code in various languages and makes programming easy for the next generations.

Automatic coding is to build a machine which can write code. The code writing process involves several wearisome steps tearing down a process into small instructions, allotting specific memory locations or to specify the system interrupts. It can also be able to manage the input-output buffers and many other things. After observing these steps as mentioned above, we implement mathematical routines, a subroutine library, and sorting programs. In this paper, we look closely towards the factors involved in Automatic programming. We studied a more extensive programming and code writing process to understand how these factors affect the process and which can help anyone build a better machine for auto programming. As we programmed manually first, we observed the process and tried to invent ways to abstract these steps to combine them into a higher-level programming language. This includes the development of interpreters, assemblers, compilers, and finally, the automatic code generator programs designed to operate with the process as mentioned earlier, that is, automatic programming.

Automatic Code Generation for Language-Learning Applications by Gabriel Sebastián; Ricardo Tesoriero;Jose A. Gallud in IEEE Latin America Transactions (Volume: 18, Issue: 08,August 2020) Language-learning applications define exercises that are pedagogical tools to introduce new language concepts. The development of this type of applications is complex due to the diversity of language-learning methodologies, the variety of execution environments and the number of different technologies that can be used. This article proposes a complete Model-Driven Architecture (MDA) approach, from the definition of the Computational Independent Model (CIM layer) to the Implementation Specific Model (ISM layer), and the process of the necessary transformations for the automatic generation of the source code (in HTML and JavaScript) of language-learning applications.

HTML Code Generation from Website Images and Sketches using Deep Learning-Based Encoder-Decoder Model by D Yashaswini; Sneha; Nikhil Kumar in 2022 IEEE 4th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA) Making mock-ups of the website's numerous pages is the first step in website design. Mock-ups can be created manually, using graphic design software, or using specialist tools. Then, software engineers turn the mock-up into structured HTML code. It takes a lot of time and effort to build the required template by repeating this method several times. This work proposes two deep learning-based encoder-decoder models that automatically generate HTML (Hypertext Markup Language) code from screenshot images of web pages and hand-drawn sketches.

A Deep Learning Based Object Detection System for User Interface Code Generation Batuhan Aşiroğlu; Sibel Senan; Pelin Görgel; M. Erdem Isenkul; Tolga Ensari; Alper Sezen; Mustafa Dağt in 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA) In this study, a GUI code generating system for web sites is designed using the Deep Learning (DL) approach. The dataset including "coordinate, width,

JSESM

ISSN:2583-3294

https://doi.org/10.5281/zenodo.7892482

Journal of Scholastic Engineering Science and Management

May 2023, Volume 2, Issue 5, pp: 38-49

height and type" of GUI objects is created using 7500 webpages. The created dataset is applied to the proposed system in order to detect objects in the GUI image and generate DSL mark-up code. Experiments were carried out to analyse the effectiveness of the proposed system and the performance evaluations were made.

AI based Auto Web Page Generation from Hand-drawn Page Mock-up Sarah Sonje; Harsh Dave; Jaswantsingh Pardeshi; Sheetal Chaudhari in 2022 IEEE 7th International conference for Convergence in Technology (I2CT) It proposes a Document Object Model (DOM) approach which identifies the hierarchy of HTML elements in the sketches and generates its corresponding code. For this approach, we make use of the Object Detector and Long Short-Term Memory (LSTM) network. An implementation of the approach is also presented along with its results.

3. SYSTEM ARCHITECTURE

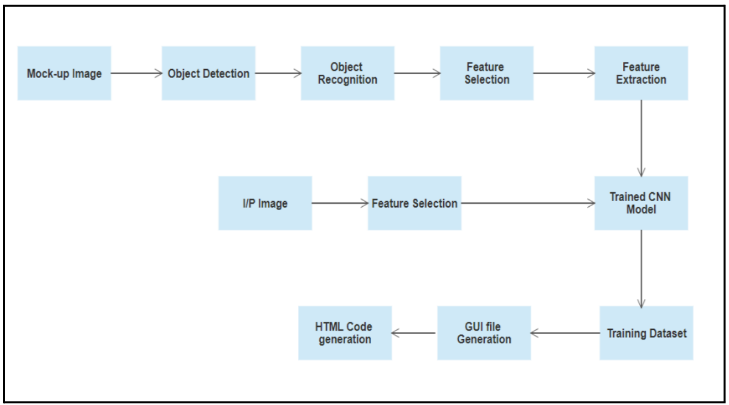


Fig.1.1: System Architecture

The first step while using the HTML builder algorithm is to execute the computer vision techniques which are used for detecting each single GUI component. The next step is to classify the detected component into the classes by the way of their work such as textbox, checkbox, toggle etc. For this step we used deep convolutional neural and at last the XML programming code is formed according to web programming structure. The author uses a program SUJSE called search programming by using this user rectify GI by simple hand-drawn & keyword. In this we use a dataset of Microsoft AI lab in a way to create our dataset.

- ❖ **Hardware Requirements**

Processor	:	I5 8th Generations
Hard Disk	:	500 GB
Monitor	:	15.6 inches
RAM	:	8 GB

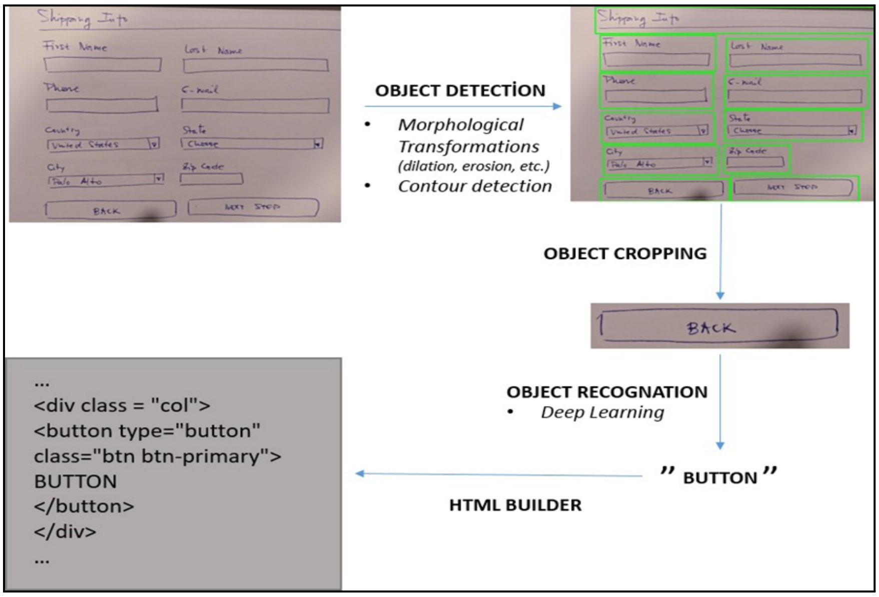
Mouse	:	Optical
Keyboard	:	Multimedia

❖ **Software Requirements**

Operating System	:	Windows 10/11
Coding language	:	Python
IDE	:	Python 3.7 IDLE

4. METHODOLOGY

This study was carried out in four basic steps. In the first step, object detection was applied on the input image with image processing techniques such as erosion, dilation, and contour detection. After this stage, the identified objects were cropped and then the components obtained were labelled with the trained CNN model. Finally, the output of this model has been converted to HTML code through the HTML Builder script.



```
...
<div class = "col">
<button type="button"
class="btn btn-primary">
BUTTON
</button>
</div>
...
```

Fig.1.2: Methodology Diagram

5. EXPERIMENTAL PROCEDURE

5.1. Object Recognition and Input/Output Design

The model was trained with the elements in our component dataset. As mentioned, 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 Cross entropy and RMS Prop algorithms by setting the batch size to 64. Afterwards, the 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 a 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. 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. The algorithm that developed for the purpose of performing HTML transformation has been demonstrated.

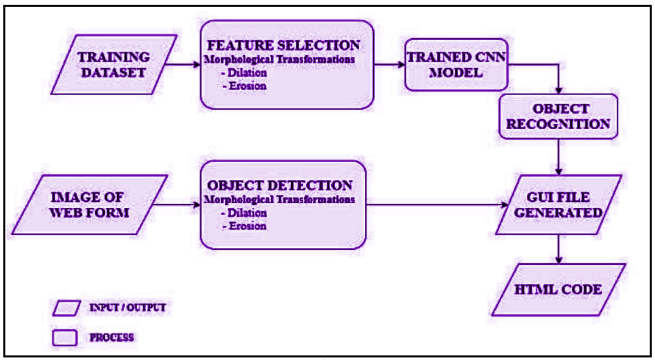


Fig.1.3: Input/Output Design

The above Figure 1.3 shows the input and output design of the Automatic code generation from mock-ups' main aim is to convert the hand-drawn mock-ups which contain text boxes, buttons, and pictures into the HTML code to make a website template or front end of our websites according to hand-drawn mock-up which is conceptual drawing. In order to convert this hand –drawn image into the HTML code to frontend template we used computer vision technique CNN model, object recognition, cropping etc. The author works on an object detection algorithm to detect the component from the hand-drawn image.

6. TESTING

Testing is the technique of inserting a system to the check in order to locate any gaps, faults, or missing necessities that are no longer met with the aid of the true needs. Principles of Testing Before attempting to shape huge take a look at cases, a software engineer must first be aware of the primary notion that underpins software program testing. All of the

JSESM

ISSN:2583-3294

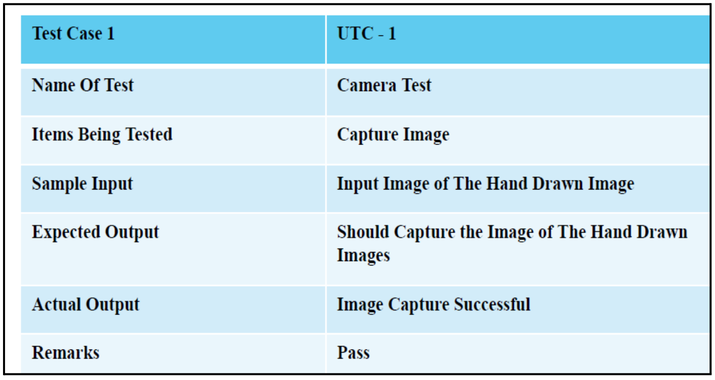
https://doi.org/10.5281/zenodo.7892482

Journal of Scholastic Engineering Science and Management

May 2023, Volume 2, Issue 5, pp: 38-49

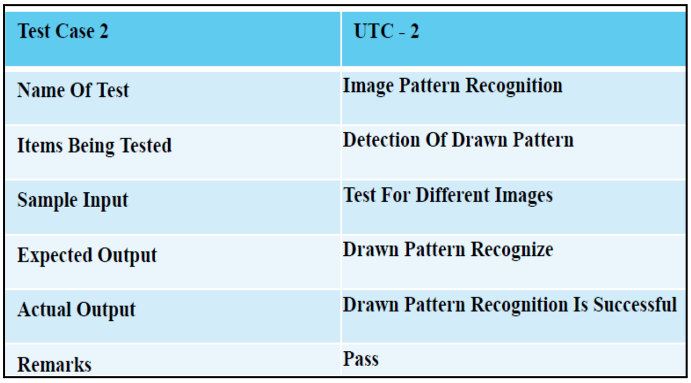
checks ought to be able to be traced back to the patron's criteria. Methods of Testing Software checking out can be completed in a number of extraordinary ways. They are, indeed, A test case is a single action, or occasionally a series of steps, used to verify that an application's behaviour/functionality and features are right. In most cases, a predicted result or outcome is offered.

Table 1.1: Input Image



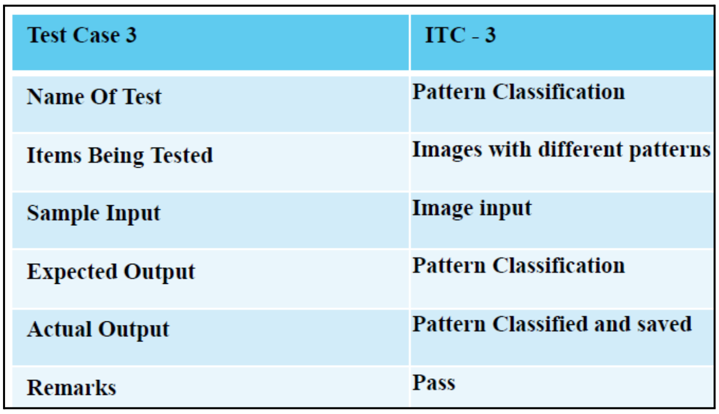
Test Case 1	UTC - 1
Name Of Test	Camera Test
Items Being Tested	Capture Image
Sample Input	Input Image of The Hand Drawn Image
Expected Output	Should Capture the Image of The Hand Drawn Images
Actual Output	Image Capture Successful
Remarks	Pass

Table 1.2: Pattern Recognition



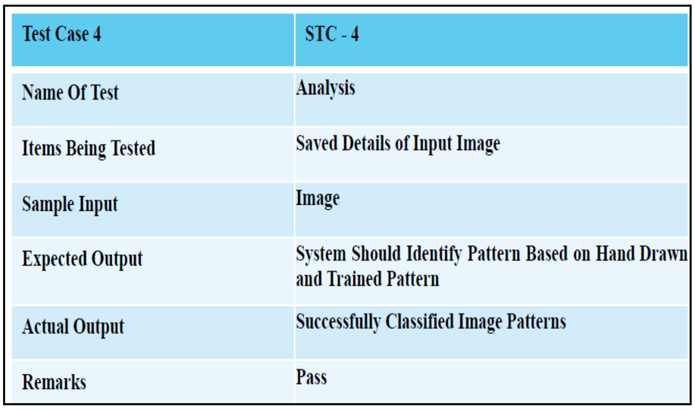
Test Case 2	UTC - 2
Name Of Test	Image Pattern Recognition
Items Being Tested	Detection Of Drawn Pattern
Sample Input	Test For Different Images
Expected Output	Drawn Pattern Recognize
Actual Output	Drawn Pattern Recognition Is Successful
Remarks	Pass

Table 1.3: Pattern Classification



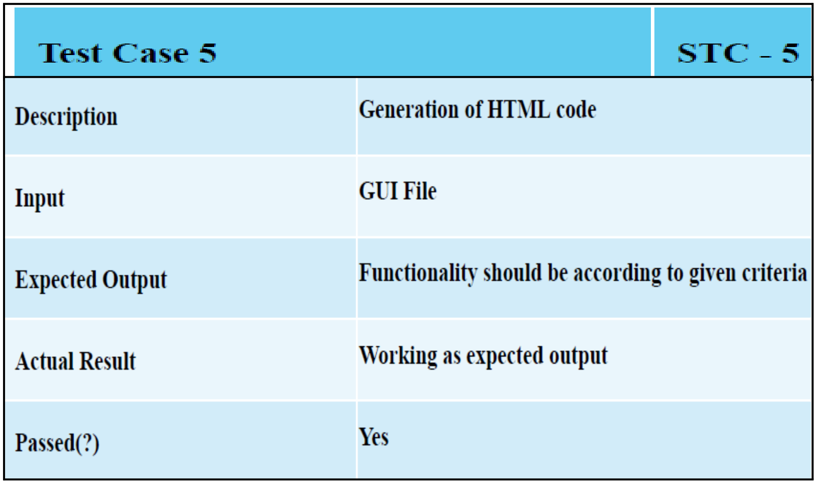
Test Case 3	ITC - 3
Name Of Test	Pattern Classification
Items Being Tested	Images with different patterns
Sample Input	Image input
Expected Output	Pattern Classification
Actual Output	Pattern Classified and saved
Remarks	Pass

Table 1.4: Analysis



Test Case 4	STC - 4
Name Of Test	Analysis
Items Being Tested	Saved Details of Input Image
Sample Input	Image
Expected Output	System Should Identify Pattern Based on Hand Drawn and Trained Pattern
Actual Output	Successfully Classified Image Patterns
Remarks	Pass

Table 1.5: Generation of HTML code



Test Case 5	STC - 5
Description	Generation of HTML code
Input	GUI File
Expected Output	Functionality should be according to given criteria
Actual Result	Working as expected output
Passed(?)	Yes

Additional data that could be included:

- Test Case ID - This parameter is used to identify a particular test case.
- Test case Description/Summary - This area explains the purpose of the test case.
- Test steps - The specific steps for conducting the test case are listed in this area.
- Pre-requisites - This field describes the criteria or steps that must be followed prior to the execution of the test steps.
- The category of the test
- The user
- Author- The Tester's name.
- Automated Ness - Whether or not this test case is automated.
- pass/fail
- Remarks

➔ **System Testing:** System checking out is a type of software program or hardware checking out that is carried out on a complete, built-in machine to verify the system's compliance with its particular requirements. System checking falls under the class of black-box testing, and as such, it must no longer necessitate knowledge of the code's inner workings or logic.

➔ **Acceptance Testing:** Acceptance testing is actually a series of particular tests whose primary goal is to thoroughly test the computer-based system. Include things like healing checks for crashes, safety checks for illegal users, and so on. Acceptance testing is done every now and then with the purchaser's actual records to ensure that the software programmer is running properly. This FDAC examination focuses on the system's external behaviour.

➔ **Validation Testing:**By generating html code after analysis and identifying the patterns from the input image against the trained patterns in the CNN model and generating html code for the GUI file using HTML builder script.Integration testing guarantees that software is constructed as a whole, that interface issues are identified and fixed, and that the last sequence of software program tests-validation attempts may begin. Validation can be defined in a variety of ways, but one simple definition is that validation is successful when software behaves predictably for users. The software programmer requirement specification, which is a document that describes all of the product's observable qualities to users, defines reasonable expectations. “Validation criteria" is a section of the specification. The validation check- out procedure is based on the data in that section.

7. RESULTS & DISCUSSIONS

7.1. Index Page: Index page where we can insert the image/hand drawn mock-up image.

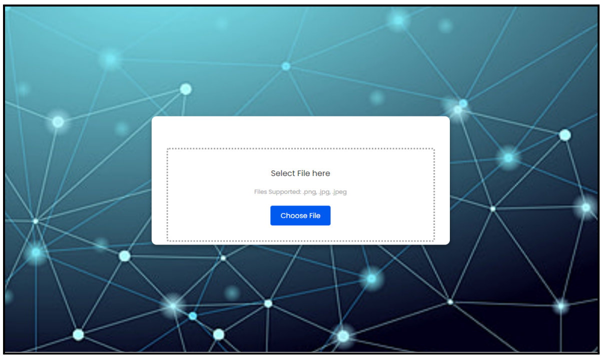


Fig.1.4: Index Page

7.2. Input Image: It is the hand drawn mock-up image which will be inserted into the system.

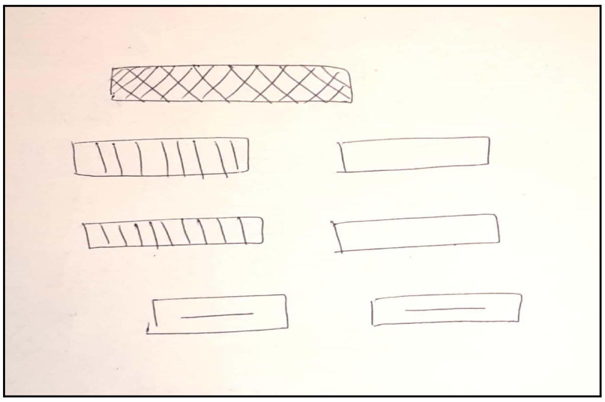


Fig.1.5: Input Image

7.3. Preprocessed Image: It is the preprocessed input image obtained after successfully performing all the image preprocessing techniques.

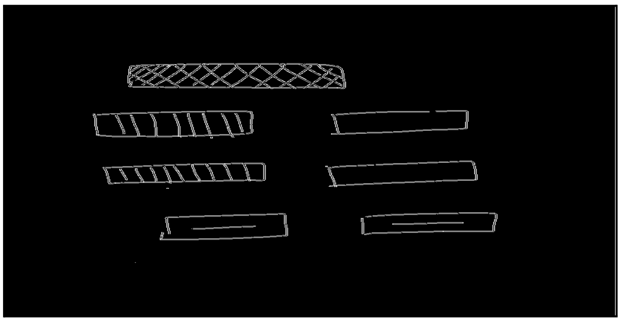


Fig.1.6: Preprocessed Image

JSESM

ISSN:2583-3294

https://doi.org/10.5281/zenodo.7892482

Journal of Scholastic Engineering Science and Management

May 2023, Volume 2, Issue 5, pp: 38-49

7.4. Upload Image: It is the page of uploading the input image to the system in order to perform all the operations to obtain the html code.

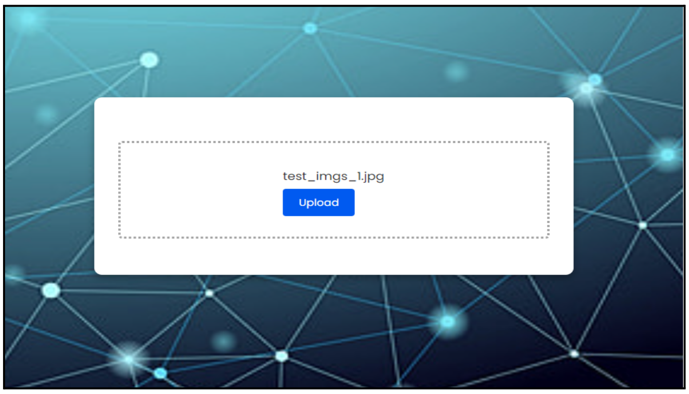


Fig.1.7: Upload Image

7.5. Input Image: It is the generated html page/GUI file along with the uploaded input image and preprocessed input image where the components in the image are identified and indicating the accuracy percentage.

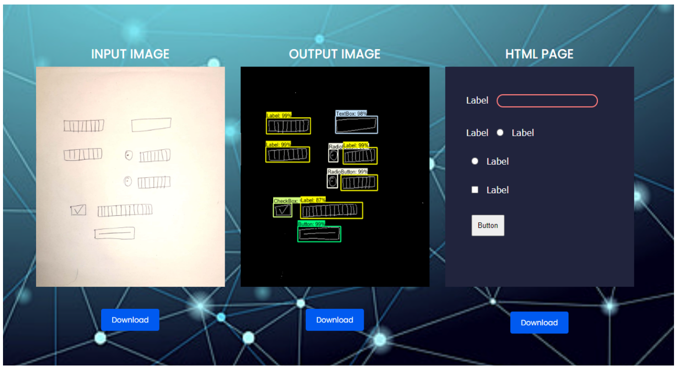
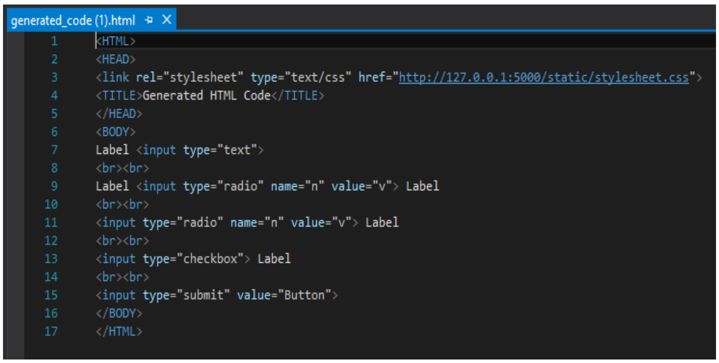


Fig.1.8: HTML Page

7.6. HTML Code: It is the generated html code for the hand drawn mock-up image given as input to the system.



generated_code (1).html

```html
<HTML>
<HEAD>
<link rel="stylesheet" type="text/css" href="http://127.0.0.1:5000/static/stylesheet.css">
<TITLE>Generated HTML Code</TITLE>
</HEAD>
<BODY>
Label <input type="text">
<br><br>
Label <input type="radio" name="n" value="v"> Label
<br><br>
<input type="radio" name="n" value="v"> Label
<br><br>
<input type="checkbox"> Label
<br><br>
<input type="submit" value="Button">
</BODY>
</HTML>
```

Fig.1.8: HTML Page

CONCLUSION

The development of an automatic HTML code generator using machine learning is a significant breakthrough in web development. This technology eliminates the need for manual coding and significantly reduces the time and effort required to create a website. The accuracy and efficiency of this approach are impressive, and it has the potential to revolutionise the web development industry. With the use of advanced machine learning algorithms and techniques, the generator can accurately generate HTML code from mock-up images, ensuring that the design and layout are retained. Additionally, the tool can be trained on a vast dataset to improve its performance, making it even more efficient in the future. Overall, the automatic HTML code generator is a game-changer, and we can expect to see more advancements in this area in the coming years. In creation of a website the cost labour and minimum time can be reduced by converting web page mock ups to their markup code. Also, by using this automatic HTML code generator it is easy to make modifications in web page creation with the least cost and minimum time. This can fasten the deployment process of the website. The purpose of this project is to create automatic HTML code from hand-drawn mock-ups. The components in the image were cropped in this study using object detection and image processing techniques. Our trained CNN model was used to determine which components were obtained. Finally, the goal of creating HTML code was accomplished utilising our HTML builder script.

REFERENCES:

1. S. Agrawal, S. Suryawanshi, V. Arsude, N. Maid and M. Kawarkhe, "Factors Involved in Artificial Intelligence-based Automated HTML Code Generation Tool," 2020 International Conference on Smart Innovations in Design, Environment, Management, Planning and Computing (ICSIDEMPC), 2020, pp. 238-241, doi: 10.1109/ICSIDEMPC49020.2020.9299609.

2. G. Sebastián, R. Tesoriero and J. A. Gallud, "Automatic Code Generation for Language-Learning Applications," in IEEE Latin America Transactions, vol. 18, no. 08, pp. 1433-1440, August 2020, doi: 10.1109/TLA.2020.9111679.

3. D. Yashaswini, Sneha and N. Kumar, "HTML Code Generation from Website Images and Sketches using Deep Learning-Based Encoder-Decoder Model," 2022 IEEE 4th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA), 2022, pp. 133-138, doi: 10.1109/ICCCMLA56841.2022.9989298.

4. B. Aşiroğlu et al., "A Deep Learning Based Object Detection System for User Interface Code Generation," 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), 2022, pp. 1-5, doi: 10.1109/HORA55278.2022.9799941.

5. S. Sonje, H. Dave, J. Pardeshi and S. Chaudhari, "draw2code: AI based Auto Web Page Generation from Hand-drawn Page Mock-up," 2022 IEEE 7th International conference for Convergence in Technology (I2CT), 2022, pp. 1-7, doi: 10.1109/I2CT54291.2022.9824521.

6. B. Aşıroğlu et al., "Automatic HTML Code Generation from Mock-Up Images Using Machine Learning Techniques," 2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT), 2019, pp. 1-4, doi: 10.1109/EBBT.2019.8741736.

7. S. Natarajan and C. Csallner, "P2A: A Tool for Converting Pixels to Animated Mobile Application User Interfaces," 2018 IEEE/ACM 5th International Conference on Mobile Software Engineering and Systems (MOBILESoft), 2018, pp. 224-235.

8. T. A. Nguyen and C. Csallner, "Reverse Engineering Mobile Application User Interfaces with REMAUI (T)," 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2015, pp. 248-259, doi: 10.1109/ASE.2015.32.

9. Moran, K. P., Bernal-Cardenas, C., Curcio, M., Bonett, R., & Poshyvanyk, D. (2018). Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps. IEEE Transactions on Software Engineering, 1–1. doi:10.1109/tse.2018.2844788

10. 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