User Interface Code Generation from Hand-drawn Sketch

Team Members

Manoj Paudel	(THA077BCT025)
Prince Poudel	(THA077BCT036)
Ronish Shrestha	(THA077BCT040)
Sonish Poudel	(THA077BCT042)

Supervised By:

Er. Devendra Kathayat

Department of Electronics and Computer Engineering
Institute of Engineering, Thapathali Campus

June 21, 2024

Presentation Outline

- Motivation
- Problem Statement and Objectives of Project
- Scope of Project
- Proposed Methodology
- Expected Results
- Project Applications
- Tentative Timeline
- References

Motivation

- Gain expertise in various programming languages and frameworks used for GUI development(eg:HTML/CSS, Javascript,React,Flutter).
- Creation of design tool that can can rapidly prototype ideas by converting hand-drawn sketches into functional interfaces.
- Learning about machine learning models,particularly those related to image recognition and processing.

Problem Statement

- Conversion of sketches into function GUI code is time-consuming, error-prone process which requires significant technical expertise.

Objectives of Project

- To construct a model able to generate quick GUI prototype from sketch into HTML code.
- To create interactive user interface.

Scope of Project

- Collaboration: Facilitating better communication between designers and developers through accurate, automated code generation.
- Educational Use: Providing a learning tool for design students to understand the link between sketching and coding.
- Accuracy: Improving the precision of UI implementation by minimizing manual translation errors.
- Rapid Prototyping: Speeding up the creation of interactive prototypes from initial design sketches.
- Accessibility: Enabling designers without coding skills to generate functional UI code.

Proposed Methodology - [1] (Datasets)

- Dataset is a wireframe sketch and associated DSL code.
- We were not aware of any dataset which contained wireframes sketches and DSL code
- We will create our own dataset.

Proposed Methodology - [2] (DSL)

- Specialized language designed to address specific aspect or needs of a particular language.
- DSLs are optimized for a particular set of tasks within a specific domain.
- Design the simple lightweight DSL to describe the GUI.
- Elements in DSL will be categorized into different hierarchical structures.

Proposed Methodology - [3] (DSL)

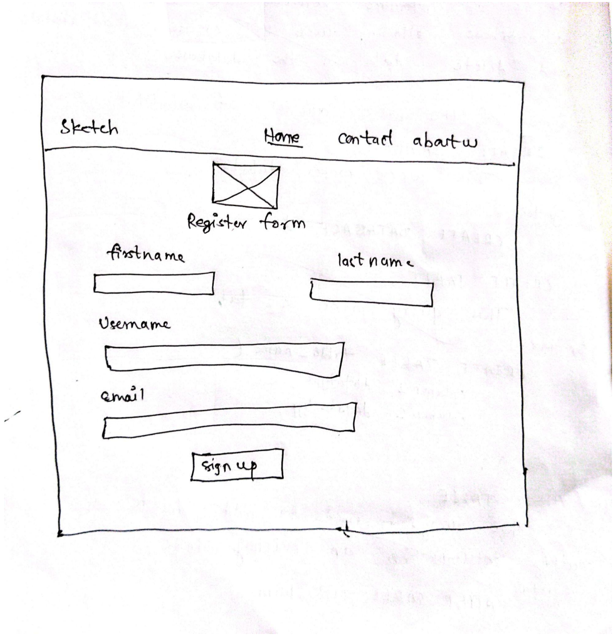
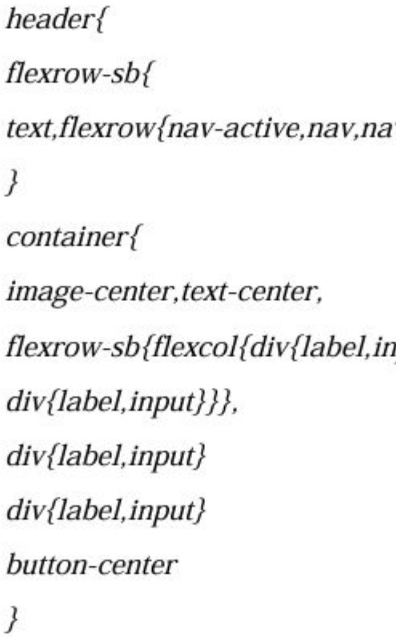


Figure: Input image



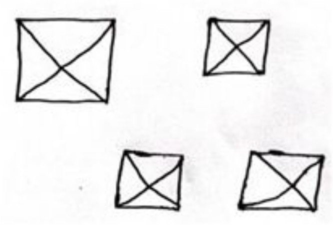
```
header{
flexrow-sb{
text,flexrow{nav-active,nav,na
}
container{
image-center,text-center,
flexrow-sb{flexcol{div{label,in
div{label,input}}},
div{label,input}
div{label,input}
button-center
}
```

Figure: DSL code

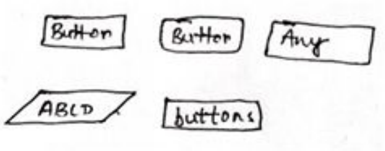
Proposed Methodology - [4] (Dataset synthesis)

- Dataset requires the hand-drawn sketch of the elements.
- Images, buttons, div, headers etc are the elements which will later be combined for UI.
- Code in DSL language is first generated.
- Different images of sketches containing the elements are synthesized.

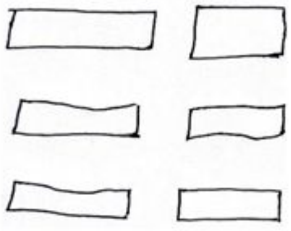
Proposed Methodology - [5] (Sample elements)



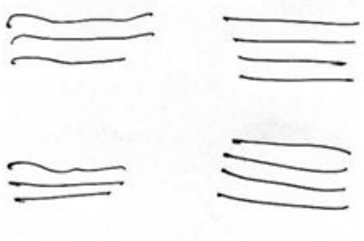
Image



Button

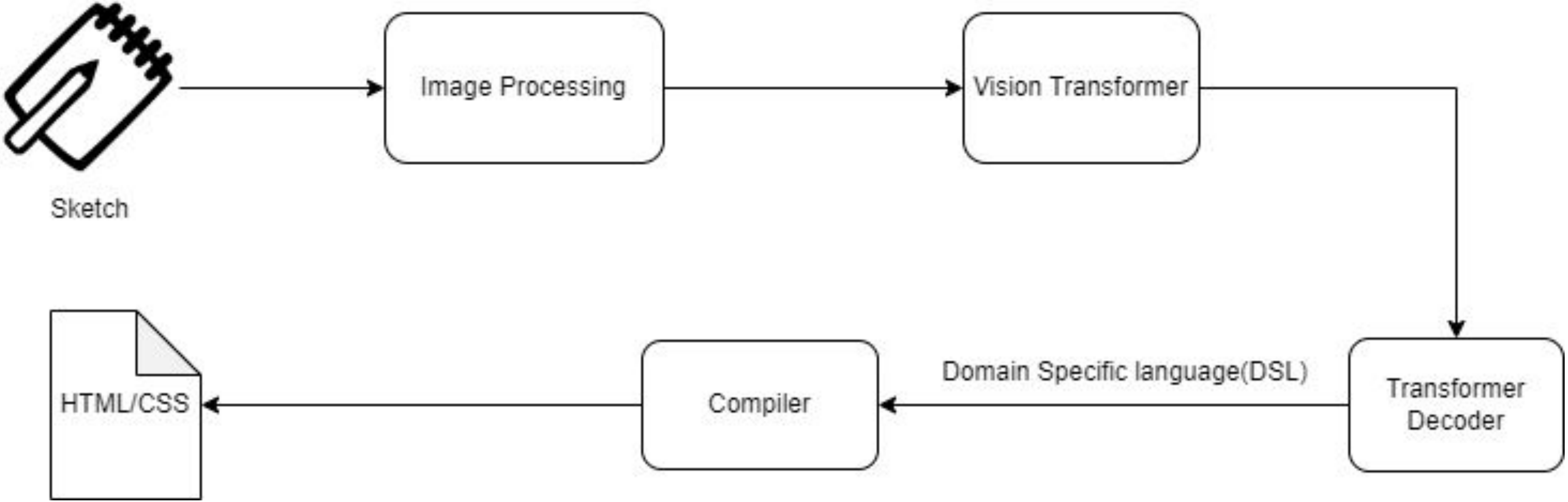


Input



Paragraph

Proposed Methodology - [6] (System Block Diagram)



Proposed Methodology - [7] (Image Processing)

Main Challenges:

- The image may not fill the entire frame, as such the background must be removed.
- The paper may be skewed or rotated.
- The image may contain noise or alterations due to lighting.

Proposed Methodology - [8] (Image Processing)

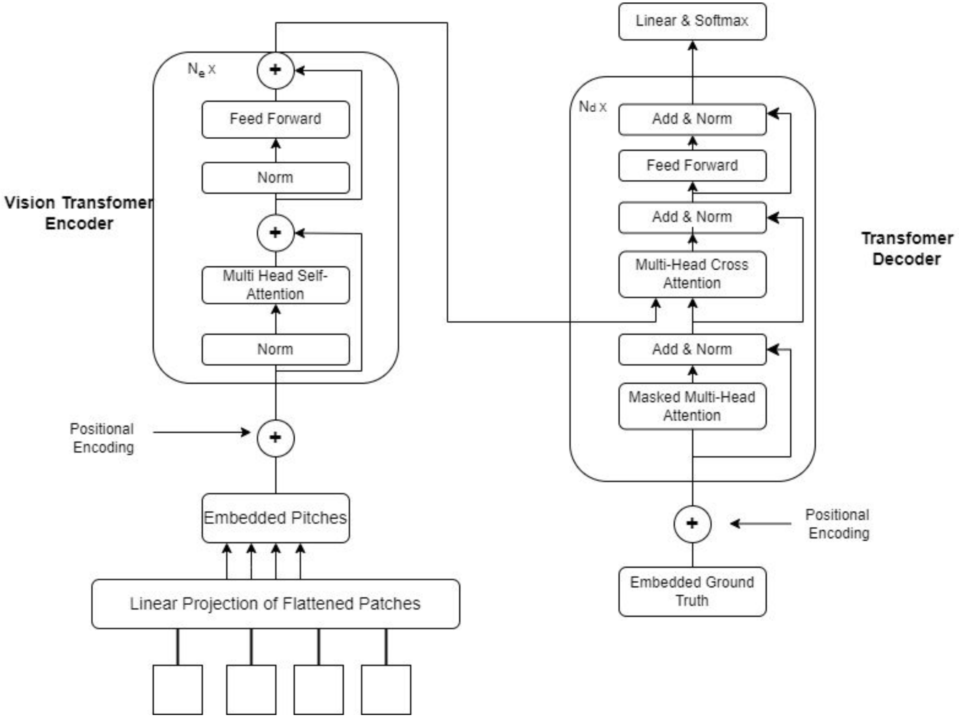
Solution:

- Using threshold filtering, canny edge detection and contour detection to find four sides of paper.
- Perspective warping to unwrap the four corners for correcting the orientation of the page.
- Canny edge detection and dilated the edge map to close small gaps between lines..

Proposed Methodology - [9] (Transformer Model)

- A deep learning architecture that consists of encoder-decoder structure.
- Consists of stacked self-attention and fully connected layer for both encoder and decoder.
- Will be used to generate structural description of the image in the form of DSL code.

Proposed Methodology - [10] (Transformer Model)



Proposed Methodology - [11] (Vision Transformer Encoder)

- Transformer designed for computer vision.
- Breaks down an image into series of patches and serializes each patch into a vector
- Vector embeddings processed by transformer encoder as token embeddings.

Proposed Methodology - [12] (Vision Transformer Encoder)

- Consist of N_e identical encoder blocks.
- Consists of two sub-layers: Multi-Headed Self-Attention (MHSA) and a Multi-layer Perceptron (MLP) head.
- Sub-layer led by a layer of normalization (LN), followed by a residual connection to the next sub-layer.

Proposed Methodology - [13] (Transformer Decoder)

- Consists of N_d stacked identical transformer block similar to the encoder.
- Composed of a masked multi-head self-attention sublayer followed by a multi-head cross attention sublayer and a positional feed-forward sublayer

Proposed Methodology - [14] (Transformer Decoder)

- Takes in encoded image embeddings and embedded ground truth sequences.
- Positional embedding is added to ground truth sequences.
- Utilizes the last decoder block's output feature to predict the next word via a linear layer whose output dimension equals the vocabulary size.

Proposed Methodology - [15] (Positional Embeddings)

- The model is actually uninformed of the token's spatial relationship.
- Used to add spatial information to the image data.
- Usually, involves assigning tokens weights derived from two high-frequency sine waves

Proposed Methodology - [16] (Multi-Head Attention)

- Contains multiple attention head with different learned linear projections..
- Contains three vectors queries, keys and values derived from input embedding through linear transformation.
- The attention is given by:

$$Attention(Q,K,V) = softmax\left(\frac{Q.K^T}{\sqrt{d_k}}\right).V$$

Where d_k is the dimension of keys and queries

Proposed Methodology - [17] (Customization)

- Transforming the sketch to HTML code is not enough.
- Component must be styled accordingly for good look.
- Some customizing by theme selection, color selection, font-selection, style selection etc.
- Contains randomly generated text can be edited
- Can add image from link or file

Proposed Methodology - [18] (Compiler)

- A computer program that translates computer code written in one programming language into another language.
- Translate the Domain Specific language into a HTML/CSS code.

Proposed Methodology - [19] (Software Requirements)

- Python Programming Language
- OpenCV
- Keras
- Tensorflow
- Numpy
- Cloud Computing resources for training: Google Colab and Kaggle

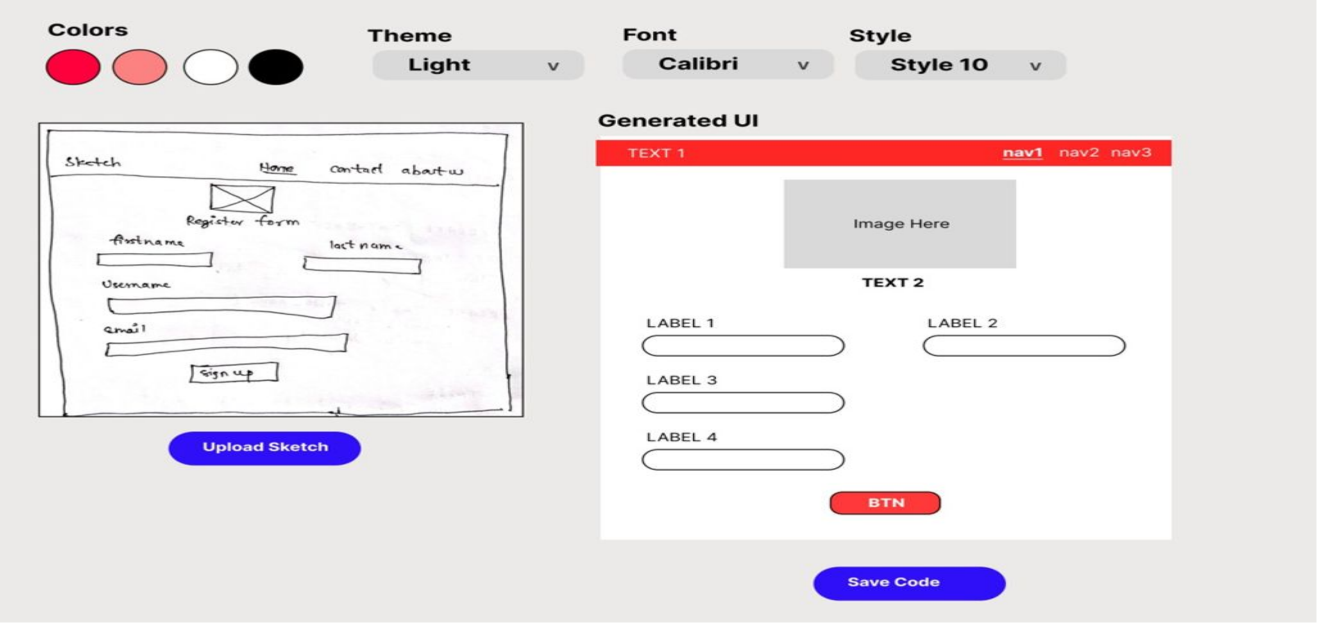
Proposed Methodology - [20] (Hardware Requirements)

- Camera to capture image
- Computer system with at least 8 GB RAM
- Dedicated GPU for model development

Expected Results-[1]

- User provide a hand-drawn sketch as a input.
- The model generates the UI where user are able to choose the colors and styles to obtain required HTML/CSS code.

Expected Results-[2]



Theme: Light

Font: Calibri

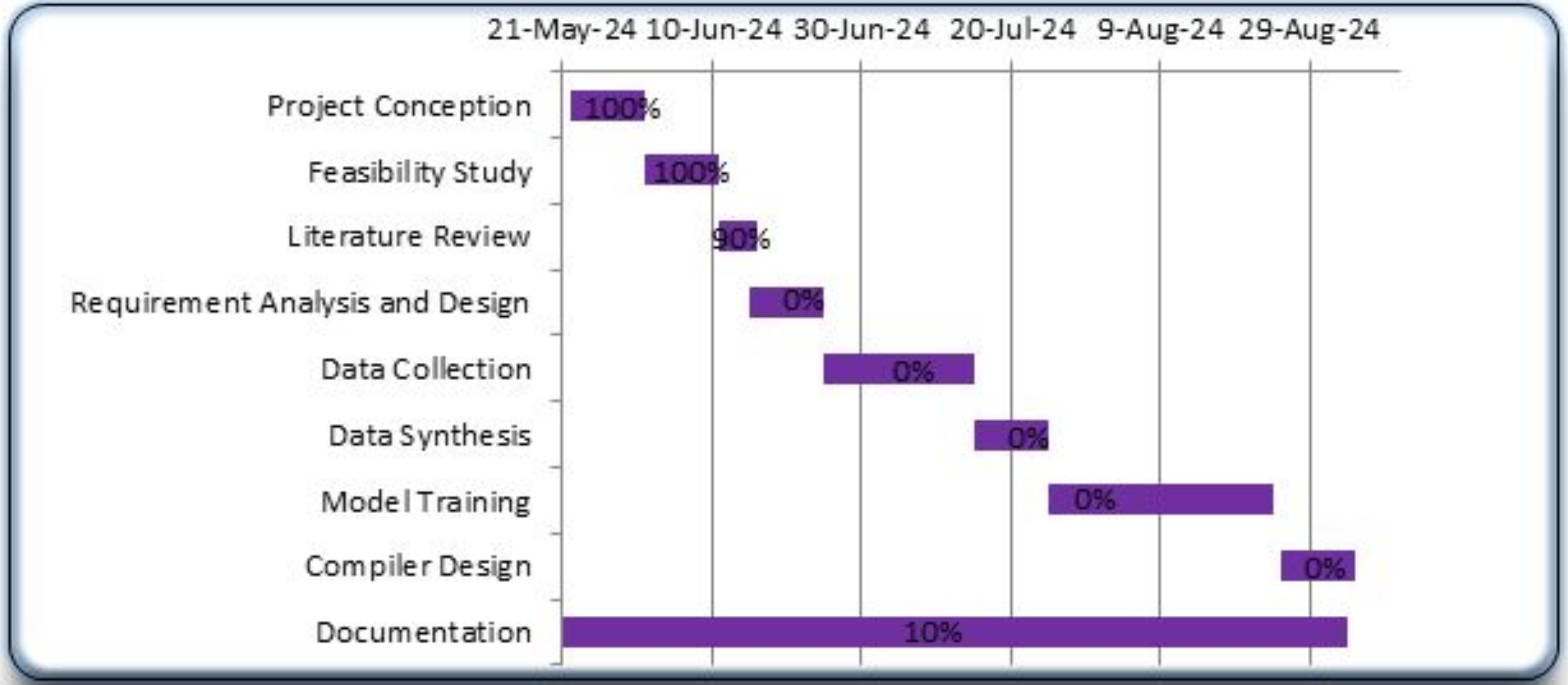
Style: Style 10

Figure: Expected User Interface

Project Applications

- It can be used in rapid prototyping for various industries like healthcare, finance and retails etc.
- It can be used in hackathons to create functional prototypes quickly.
- It can be used in startups to create functional prototypes and MVPs quickly.

Tentative Timeline



References-[1]

1. Tony Beltramelli,"Pix2code: Generating Code from a Graphical User Interface Screenshot," 2017. [Online]. Available: https://www.researchgate.net/publication/325920827_pix2code_Generating_Code_from_a_Graphical_User_Interface_Screenshot. [Accessed: June 2024].
2. Sarah Suleri, Vinoth Pandian,Svetlana Shiskovets,Matthias Jarke,"Eve: A sketch-based Software Prototyping Workbench," 2019. Available: https://www.researchgate.net/publication/332777261_Eve_A_Sketch-based_Software_Prototyping_Workbench. [Accessed: June 2024].
3. Biniam Behailu Adefris, Ayalew Belay Habtie, Yesuneh Getachew Taye, "Automatic Code Generation from Low Fidelity Graphical User Interface Sketches Using Deep Learning," 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9971204. [Accessed: June 2024].
4. Daniel Baulé, Christiane Gresse von Wangenheim, Aldo von Wangenheim, Jean C. R. Hauck and Edson C. Vargas Júnior, "Automatic Code Generation from Sketches of Mobile Applications in End-User Development Using Deep Learning," [Online]. Available: https://www.researchgate.net/publication/349963791_Automatic_code_generation_from_sketches_of_mobile_applications_in_enduser_development_using_Deep_Learning . [Accessed: June 2024].

References-[2]

5. Jia Li, Yongmin Li, Ge Li, Zhi Jin, Yiyang Hao, and Xing Hu, "STC (Sketch To Code) - An Enhanced HTML & CSS Autocode Generator from Handwritten Text and Image Using Deep Learning," IEEE, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10537336. [Accessed: June 2024].
6. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," [Online]. Available: https://arxiv.org/abs/2010.11929. [Accessed: June 2024].
7. "Canny edge detector," Wikipedia, The Free Encyclopedia. [Online]. Available: https://en.wikipedia.org/wiki/Canny_edge_detector. [Accessed: 17-Jun-2024].
8. Q. Xin, Y. Zhang and B. Tan, "Image Captioning with Vision/Text Transformers," [Online]. Available: https://qi-xin.github.io/image%20caption%20generation.pdf. [Accessed: June 2024].