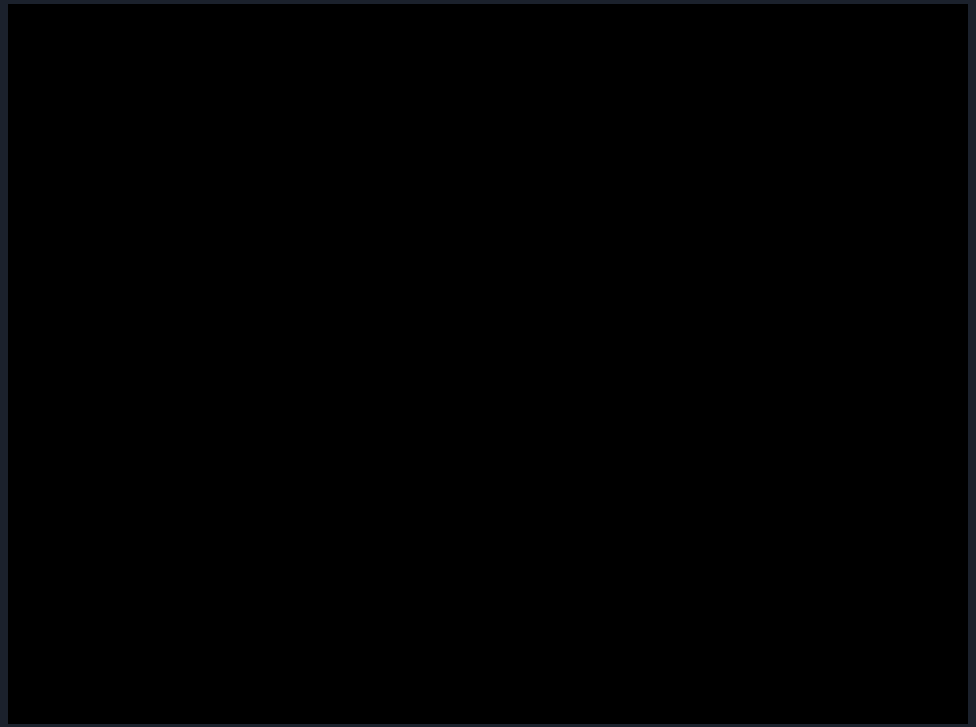
# Text-Transcriber

Cheerful Cheetahs

# Video



# Purpose

The model's purpose is to take a line of **handwritten text** as an **input** and **output** the text in **typed and spoken form**

This is intended for use by various audiences such as the **visually impaired**, **teachers**, and anyone who wants to **eliminate the time** it takes to transcribe text.

It can be a monotonous task to hand-transcribe sentences, and even impossible to many.

**Text-Transcriber saves time and provides a way for the visually impaired to hear text they may not be able to read**

# Process

**Back-end:**

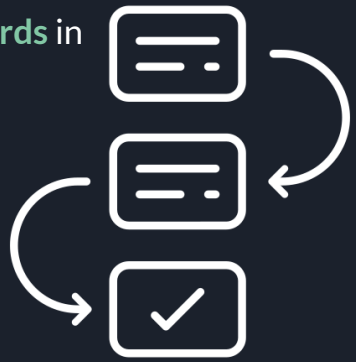
First, we trained a model taken off of github in order to **recognize handwritten words** and **translate them to text** (Text-Transcriber)

Next, we implemented a “word-detector” that will find and **separate the different words** in a sentence and save them as separate images

Using these images, we put them through our first model, the Text Transcriber, and **translated the whole sentence in parts.**

We used Google's Text-To-Speech to **read the sentence we formed out loud**.

Lastly, we used **Flask** to develop the website to function and **ran it locally**



**Front-end:**

We **modified a template** using HTML/CSS and enhanced it with Javascript

# Difficulties

Setting up the project and installing all the dependencies.

The model took a very long time to train to the point where we had to let it run overnight.

Streamlining all the processes to get everything to work together.

Getting the audio to work properly and keeping it up to date

Getting the text and audio outputs to show up on the website

# Future Improvements

Training the model on a **broader dataset** to improve accuracy

Adding a **translation feature** after the text is transcribed.

Overall our model is functional but has opportunities to improve

# Resources

- To be able to create an audio playback after transcribing the image uploaded we used **Google's TTS API**

- We utilized some **githubs** to get resources we needed

  - https://github.com/githubharald/SimpleHTR - (Base for training our model)

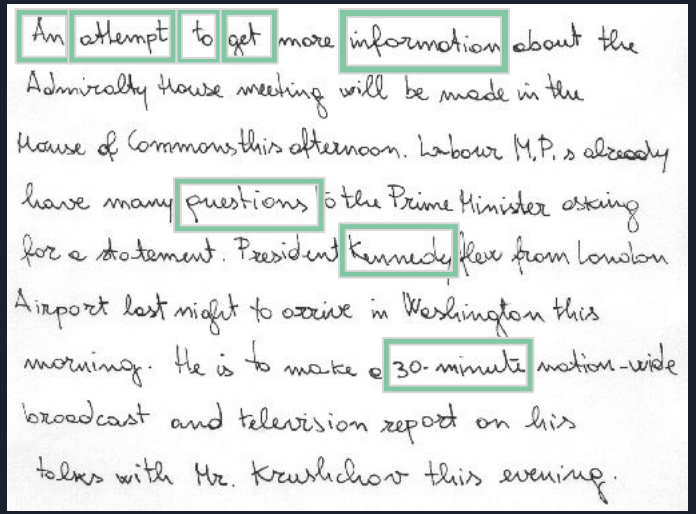
  - https://github.com/filyp/autocorrect - (To autocorrect when the probability is >0.75

  - https://github.com/githubharald/WordDetector - (For figuring out when a word ends)

- We used the **Scrolling Nav** template from start **bootstrap** to get started on our website

# Performance & Accuracy

- Best character error rate: ~10.861%
- Best  word accuracy: ~73.569%
- Run times (cpu only):
  - ~5 seconds for page
  - ~2.5 seconds for line
  - ~2 seconds for single word
- Word detector accuracy ~99% (for IAM dataset)



# Demo of website