CIPHER

Bad Bunnies: Ethan Sie, Wen Zhang, Stanley Hoo, Raymond Lin
Softdev
P05: Le Fin
Time Spent: 4 hours
TARGET SHIP DATE: 2025-06-02

Project Description:
Users can train our AI model by uploading samples of their writing and our model will learn handwriting patterns unique to each user. Then users can upload a document of their handwriting that they want to convert to text, and the AI will use what it learned to convert messy handwriting to legible, digital text.

Project Extension (if time/feasibility allows):
Users can train an AI model to create a custom font style based on their handwriting. Users will be able to upload documents of their handwriting similarly as before, but instead will create a font that is representative of their writing.

Program Components:

1. **Flask Routes (with Jinja templating) - Updated as new data is requested by python**
    a. **/ | semi-accessible without logging in**
        i. Allows users to register, create an account, or login
        ii. If logged in, will display instructions
    b. **/register | accessible without logging in**
        i. Allows user to register with password confirmation
        ii. Must have a unique username
        iii. Redirects to / -> renders home.html
    c. **/login | accessible without logging in**
        i. Cannot log in if the account is already in an active session
        ii. Redirects to / -> renders home.html
    d. **/logout | must login**
        i. Redirects to / -> renders home.html
    e. **/train (to train) | accessible without logging in**
        i. Users upload samples (at least 2) of their writing along with a translation of it to train the AI model (uploads must all be of type png, jpg, jpeg, heic, heif, tif, tiff, or pdf)
        ii. Will update the user how accurate a transcription may be based on the amount of input (handwriting samples)
            1. Users will be given text to copy and upload that will be used to train the model. Users will have an option to upload additional samples to improve the accuracy of the model.

2. Split all isolated characters from input data randomly into 80-20. 80% to train the model and 20% to test the model.
3. We will also require the user to train all of the letters of the alphabet, all numbers, all punctuation, and common special characters (@#$%^&*<>{}[]()=+~). This could be helpful for us to identify which characters are not being trained well and can be shown as a statistic of each letter's accurateness

iii. Model will continue to train with each additional user upload, and will use the output when analyzing to retrain the model. This can be stored safely in a pickle file and can also be saved locally.

f. /upload (to transcribe to text) | accessible without logging in

i. Stores all user uploaded images, filenames will be hashed with original filename, user id, and current datetime for uniqueness
ii. Here, users can upload files of their handwriting that they want to convert to digital text

g. /results | accessible without logging in

i. Result of what the AI model converts the handwritten text to, can also be downloaded as a PDF

h. /history | must login

i. History of past user models analytics: validation accuracy
ii. History of past applied models (through recognition/handwriting table in SQL); Will display the images

2. SQLite3 Databases - Stores information

users: will store user identification, password

user_id	username	password_hash
INTEGER AUTOINCREMENT	TEXT NOT NULL UNIQUE	TEXT NOT NULL

uploads: will store the paths to the sampling images (stored in a local folder) that the user has inputted

image_id	user_id	training	image_path	datetime
INTEGER AUTOINCREMENT	INTEGER NOT NULL	BOOLEAN	TEXT NOT NULL	TEXT NOT NULL

3. Tailwind CSS - Frontend Framework

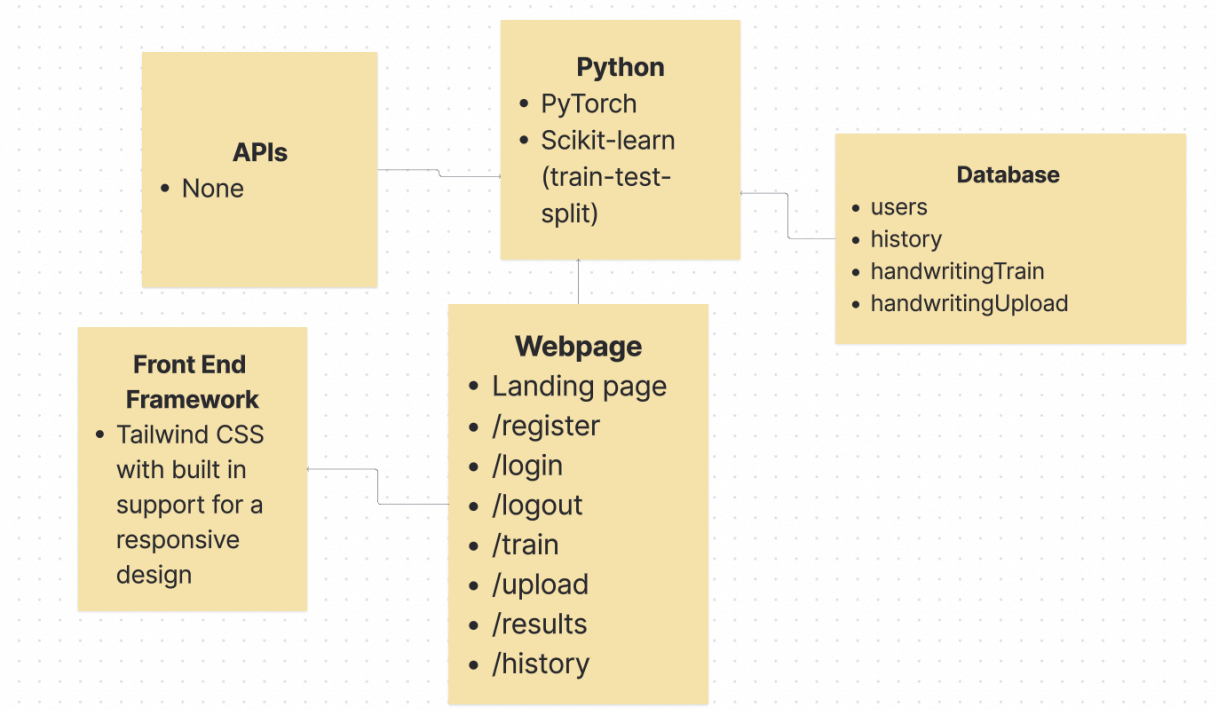
a. Tailwind CSS allows the writer to make use of existing utility classes as a shorthand when directly styling elements in HTML.
b. Tailwind has built in support for a responsive design, making it easier to create aesthetic buttons and sliders for this project.

4. **File/AI Model Storage** - stores the user uploaded images and the AI model
    a. Images will be stored in the /image_uploads folder, each filename will be a hashed string encrypted using SHA-256 using the original filename, user id, and datetime for uniqueness
    b. AI models will be stored in the /models folder
        i. Main model stored here
        ii. LoRA models (models fine tuned to each user) stored in /user_adapters folder
5. **APIs to use:** None

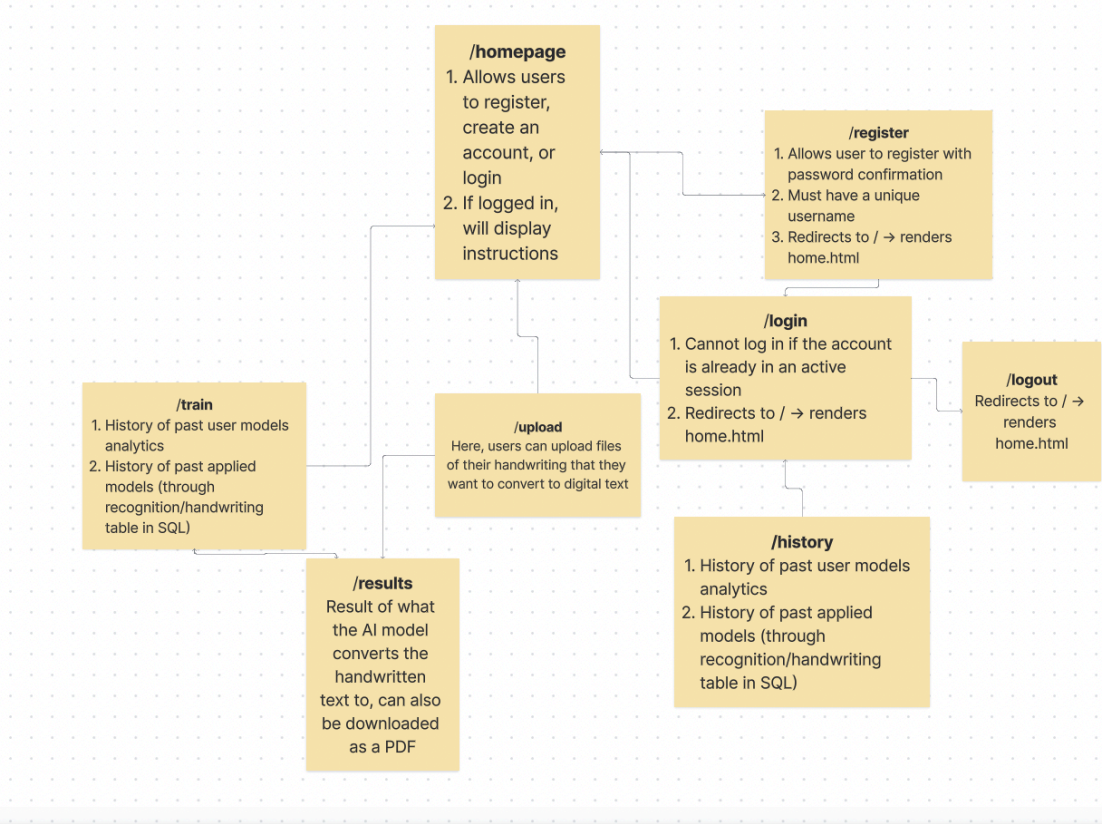
6. **Libraries/Modules/Packages:**

Package	Purpose
Werkzeug	Used for security, in this case password hashing
`torch` **(PyTorch)**	Core deep learning framework for building and training neural networks (e.g., CNNs, RNNs). Handles layers, losses, optimizers, and backpropagation.
`torchvision`	Adds vision-specific utilities to PyTorch, including image transformations, data loaders, and pretrained models.
`scikit-learn`	Used for tasks like `train_test_split` to split datasets into training and test sets (e.g., 80/20 split).
`transformers`	Hugging Face library that provides pretrained models like TrOCR for handwriting OCR and image-to-text conversion.
`peft`	Adds **LoRA (Low-Rank Adaptation)** modules to Hugging Face models, enabling lightweight, per-user fine-tuning without retraining the whole model.
`Pillow` **(PIL)**	Handles image loading, conversion (e.g., to grayscale), resizing, cropping, and saving. Fast and lightweight for basic image manipulation.
`opencv-python`	Used for more advanced preprocessing like denoising, skew correction, thresholding, and contour detection. Good for cleaning handwritten input.
`pdf2image`	Converts `.pdf` files to images (e.g., PNG), allowing users to upload scanned documents for handwriting recognition.
`os`	Native Python module to work with directories, paths, file operations, and permissions.
`hashlib`	Hashes filenames (e.g., using SHA-256) to avoid collisions and ensure each file is uniquely identifiable.
`datetime`	Generates timestamps to help version files or append unique datetime strings to filenames or database records.

Component Map:



Site Map:



Tasks:

Ethan Sie (Project Manager):

- Database Lead
- LoRA Model
- Setting up basic flask routing

Wen Zhang:

- JS Lead
- Image processing: Isolating characters in images
- Creating file upload system

Stanley Hoo:

- Torch General Model
- LoRA Model
- DB help

Raymond Lin:

- Create designs for frontend of the website
    - Tailwind styling
- Creation of handwriting samples for training
- Backend help

Everyone will be helping out wherever/whenever is required.