CIPHER by BAD BUNNIES

Bad Bunnies: Ethan Sie, Wen Zhang, Stanley Hoo, Raymond Lin
Softdev
P05: Le Fin
Time Spent: 6 hours
SHIP DATE: 2025-06-06

Project Description:
Users will have two main functions on our website: converting text to emojis and converting handwritten samples into digital text. Both are fed into models using AI, vectorization, and techniques such as KNN. After users have either uploaded an image or entered some amount of text, they will see the final results on a display screen.

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. **/emoji | accessible without logging in**
        i. Users upload samples of digital text
        ii. Will apply a model to understand semantic meaning and display a string of emojis that represent a users' desires.
        iii. This data will be stored if a user is logged in.
    f. **/handwriting | accessible without logging in**
        i. Users upload samples of handwritten text.
        ii. Will apply a model to identify words and correct them to represent logical phrases.
        iii. This data will be stored if a user is logged in.
    g. **/history | must login**
        i. History of past user requests
    h. **/transcriptions | accessible without logging in**

i. Public display of all user requests

i. **/admin** **| must be a meta user**

i. Moderates the images/text being displayed on /transcriptions
ii. Only accessible by set admin users

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

handwriting: will store the paths to the sampling images and the resulting text

count	user_id	image_path	output	approved
INTEGER AUTOINCREMENT	INTEGER NOT NULL	TEXT NOT NULL	TEXT NOT NULL	BOOLEAN

emoji: will store the input text and output from our model

count	user_id	input	output
INTEGER AUTOINCREMENT	INTEGER NOT NULL	TEXT NOT NULL	TEXT NOT NULL

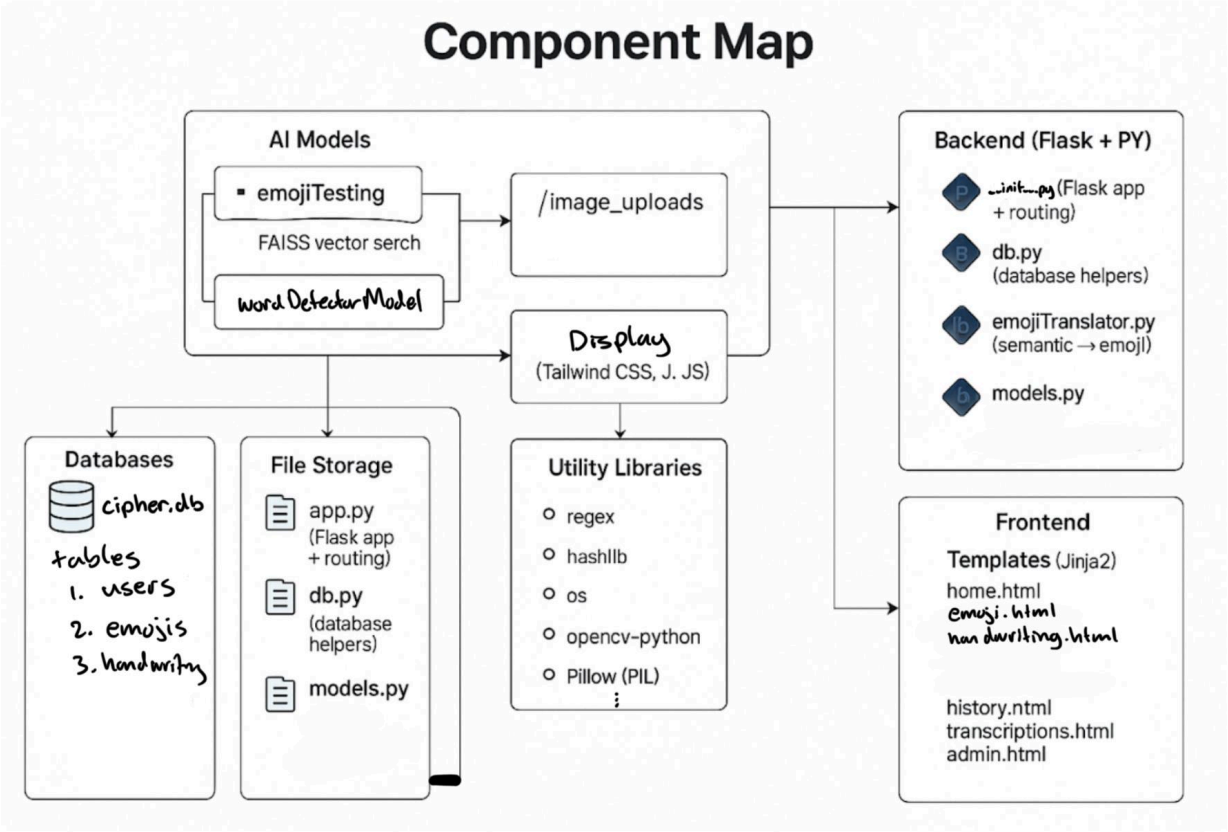
3. **Tailwind CSS - Frontend Framework**
   a. Tailwind CSS allows the writer to make use of existing utility classes as a shorthand when directly styling elements in HTML.
   b. Tailwind has built in support for a responsive design, making it easier to create aesthetic buttons and sliders for this project.

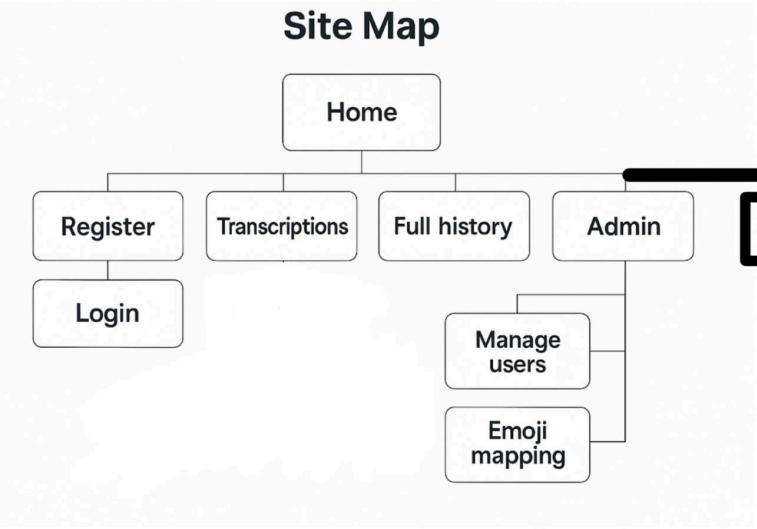
4. **File/AI Model Storage** - stores the user uploaded images and the AI model
   a. Images will be stored in the /image_uploads folder and will be accessed via the image path that is stored in the database.
   b. AI models will be stored in the /WordDetectorModel and /emojiTesting folder
      i. Main model stored here for handwritten transcription
      ii. Sentence_transformers, FAISS model for emoji conversion will be stored here.

5. **APIs to use:** Gemini API

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

Package	Purpose
`Werkzeug`	Used for security, in this case password hashing
`transformers`	Hugging Face library that provides pretrained models like TrOCR for handwriting OCR and image-to-text conversion.
`FAISS`	Facebook AI Similarity Search allows for easy access to AI for nearest neighbor searching and embedding semantic meanings.
`regex`	Regex allows for easy manipulation of messy data to extract the information that you want.
`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.


Component Map
AI Models
emojiTesting
FAISS vector serch
word DetectorModel
/image_uploads
Display
(Tailwind CSS, J. JS)
Backend (Flask + PY)
__init__.py (Flask app + routing)
db.py
(database helpers)
emojiTranslator.py
(semantic → emoji)
models.py
Databases
cipher.db
tables
1. users
2. emojis
3. handwrity
File Storage
app.py
(Flask app + routing)
db.py
(database helpers)
models.py
Utility Libraries
regex
hashlib
os
opencv-python
Pillow (PIL)
Frontend
Templates (Jinja2)
home.html
emoji.html
handwriting.html
history.nhtml
transcriptions.html
admin.html


Site Map
Home
Register
Transcriptions
Full history
Admin
Emoji
Handwriting
Login
Manage users
Emoji mapping

Tasks:

Ethan Sie (Project Manager):

- Emoji Model
- Setting up basic flask routing

Wen Zhang:

- DB Lead
- Handling public page for transcriptions and approval
- Creating file upload system

Stanley Hoo:

- Handwriting transcription 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.