P00: ~~Half-Quick~~ Move Slowly and Fix Things
Design Doc by DECK, pd. 5
↳ *Roster: Cindy Liu, David Lee, Ethan Cheung, Kalimul Kaif*

PROJECT NAME: “BlogDECK”
A Flask application utilizing SQLite databases to store and display blogs. Readers will create or log into accounts to read others’ blogs or create and edit their own.

TARGET SHIP DATE: 2025-11-10

Program Components:

- ☐ sqlite3 (backend data storage system)
  - ☐ users (stores relevant information for each user)
  - ☐ blogs (stores relevant information for each blog)
  - ☐ entries (stores entries of all blogs)
- ☐ Python (application layer)
  - ☐ __init__.py (runs Python)
- ☐ Flask (web server/delivery framework)
- ☐ html (frontend display)
  - ☐ login.html (checks entered login info against database)
  - ☐ register.html (adds new login info to database)
  - ☐ homepage.html (displays user’s blogs and other blogs)
  - ☐ profile_page.html (displays user profile, bio, creation date, list of blogs)
  - ☐ edit_profile.html (allows users to edit their profile bio)
  - ☐ create_page.html (enables blog creation with an initial entry)
  - ☐ edit_page.html (displays list of user’s blogs for editing)
  - ☐ view_blog.html (displays blog and all its entries with timestamps)
  - ☐ edit_blog.html (enables user “blog” edits)

Our MVP:

A blogging platform with user login, viewable profiles, and a system for tracking contributions to blogs.

- SQLite (store user and blog data)
- Python (run scripts to serve HTML web pages)
- Flask (serve dynamic HTML pages)
- CSS (style the served HTML pages)

Data Organization

- **users**
  - a master list of all users and related information (username, password, profile biographies, date of creation)
- **blogs**
  - a master list of all the blogs with related information, expanded upon in entries (name, creator, creation date)
- **entries**
  - a master list of all entries in all blogs (title of entry, content, name of blog, blog creator, timestamp)

users is used by Flask in a Python script for:

- ***login.html*** (retrieve and compare user input credentials to database)
- ***register.html*** (add new user information to database, check for duplicate usernames)
- ***profile_page.html*** (display username, biography, and creation date for current user or other users)
- ***edit_profile.html*** (update user biography)

blogs is used for:

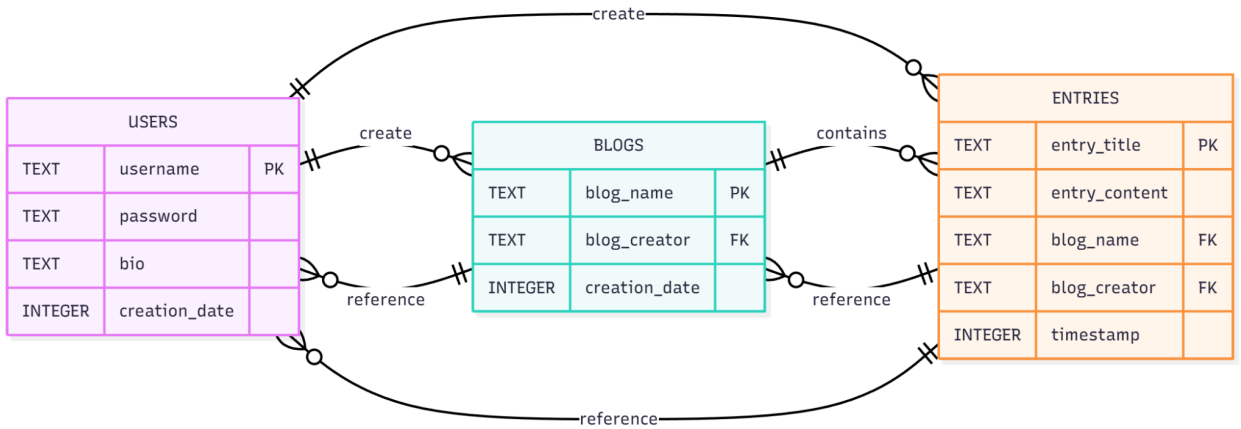
- ***homepage.html*** (display personal blogs and all other blogs available for viewing, reverse chronological order)
- ***create_page.html*** (add new blog to database)
- ***edit_page.html*** (display user's own blogs for management)
- ***profile_page.html*** (display the user's bio, creation date, list of blogs created by user)
- ***view_blog.html*** (retrieve blog information)
- ***edit_blog.html*** (given blog is owned by user, allow edits)

entries is used for:

- ***create_page.html*** (append first entry after form is submitted)
- ***view_blog.html*** (retrieve entry information)
- ***edit_blog.html*** (retrieve and change entry information)
- ***profile_page.html*** (displays most recent edit timestamp of all entries in a blog)

All pages are dynamic in nature, because they each have user-specific outcomes (changes made to user state) and/or interact with at least one database for the addition/removal of displayed contents on each page.

Component Map:



Database Organization:

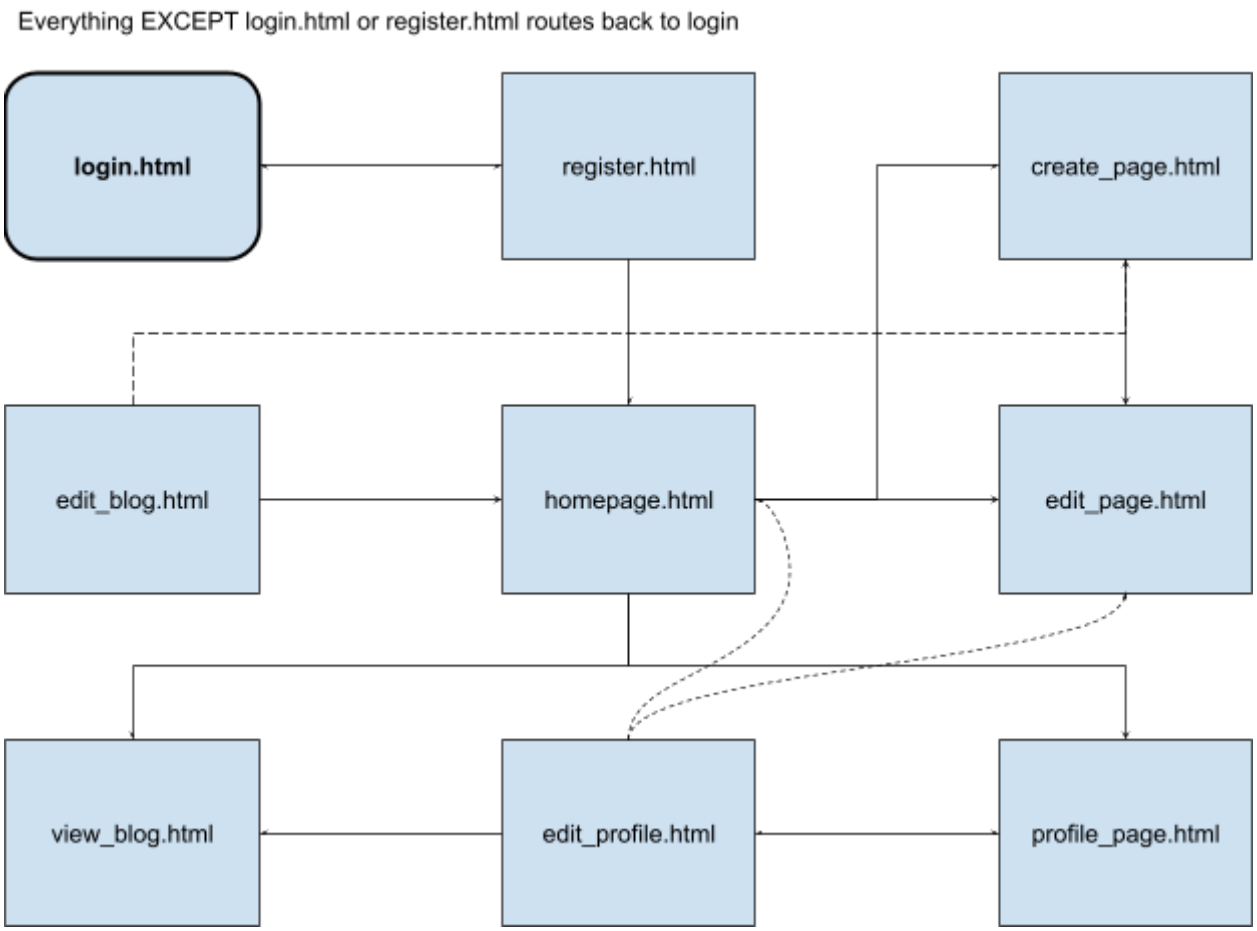
USERS			
TEXT	username	PK	unique identifier of blog user
TEXT	password		
TEXT	bio		user customization of profile
INTEGER	creation_date		only populated once during acc. creation

BLOGS			
TEXT	blog_name	PK	unique blog names
TEXT	blog_creator	FK	references users in user_master_list
INTEGER	creation_date		Python datetime import to EST convert

Database Organization (cont.):

ENTRIES			
TEXT	entry_title	PK	unique entry name
TEXT	entry_content		
TEXT	blog_name	FK	reference blog names in blog_master_list
TEXT	blog_creator	FK	references users in user_master_list
INTEGER	timestamp		

Site Map:



Breakdown of Tasks:

- ☐ install guides & launch codes (cindy, ethan)
- ☐ requirements.txt
- ☐ setting up sqlite3 databases in app.py (david)

creating templates + corresponding functions in app.py:

- ☐ login.html (david)
- ☐ register.html (david)
- ☐ homepage.html (cindy)
- ☐ profile_page.html (kalimul)
- ☐ edit_profile.html (kalimul)
- ☐ create_page.html (david, ethan)
- ☐ edit_page.html (david, ethan)
- ☐ view_blog.html (david)
- ☐ edit_blog.html (ethan, david)

css:

- ☐ navbar (kalimul)
- ☐ misc. (all at will)