Foo Bar:: Ameer Alnasser (PM), Wan Ying Li, Kevin Wang
SoftDev
P00 -- wiki design Revised
2022-11-1
time spent: 1 hrs

Target ship date: 2022-11-22

DB organization:

- A table to store data for blogs (handles both current blog and past blogs)
    - Username
    - Title
    - Blog content
    - Timestamp
- A table to store usernames and passwords
    - Username
    - password

Program components:

Database (Kevin)

- See DB organization

Homepage (Kevin)

Helps a user navigate through the possible pages

- Directs a user through all the intermediate pages
- When authenticated:
    - User can navigate to the create blog page
    - Otherwise: don’t show the navigation to the create blog page

Blog (Kevin)

A beautiful, magnificent, sexy page for viewing your favorite blogs

- Includes:
    - Author (their username)
    - Time (last edited)
    - Title
    - Blog content

- Reads the table BLOGS for the requested username’s title’s blog, but the most recent timestamp for it

Blog History (Kevin)

View the past history of the blog at {username}/{title}

1. Query the database table BLOGS for all blogs with {username}/{title} except the most recent
2. Sort them by timestamp (obtained from python), recent earlier
3. Display on the website

Profile (Ameer)

See all the blogs a certain user has published, sorted by title

- Direct user to a blog when they clicked the hyperlinked title of that blog

1. Query all of the blogs that belong to the specific user
2. Sort previous query by timestamp
3. Filter down by unique title (means it’s a unique blog - because there can be many old, edited versions of the same blog)

Sign up (Ameer)

Gives fields for a user to create a new account with username and password.

1. Make sure that there is no user already logged in
2. Searches database table AUTHENTICATION for existing user
    a. If it exists: send an error message back “user already exists”

3. Create a new entry in the table with the username and password provided

4. Do some flask user voodoo for persisting the account for the session

Log in (Ameer)

Gives fields for a user to login to their existing account with username and password.

1. Make sure that there is no user already logged in
2. Searches database table AUTHENTICATION for matching username and password

   a. If username exists but password is wrong, send back an error message for “incorrect password”
   b. If username doesn’t exist, send back an error message for “user not found”
3. Do some flask user voodoo for persisting the account for the session

Create (Wanying)

Gives fields (title, content) for a user to create a new blog.

1. Check if the user is logged in
   a. Yes, process to step 2
   b. No, direct user to log-in page
2. Create a new entry in the BLOG table with title and content provided by the user, username and timestamp from __init__.py

Edit (Wanying)

Users can change the content of their blog.

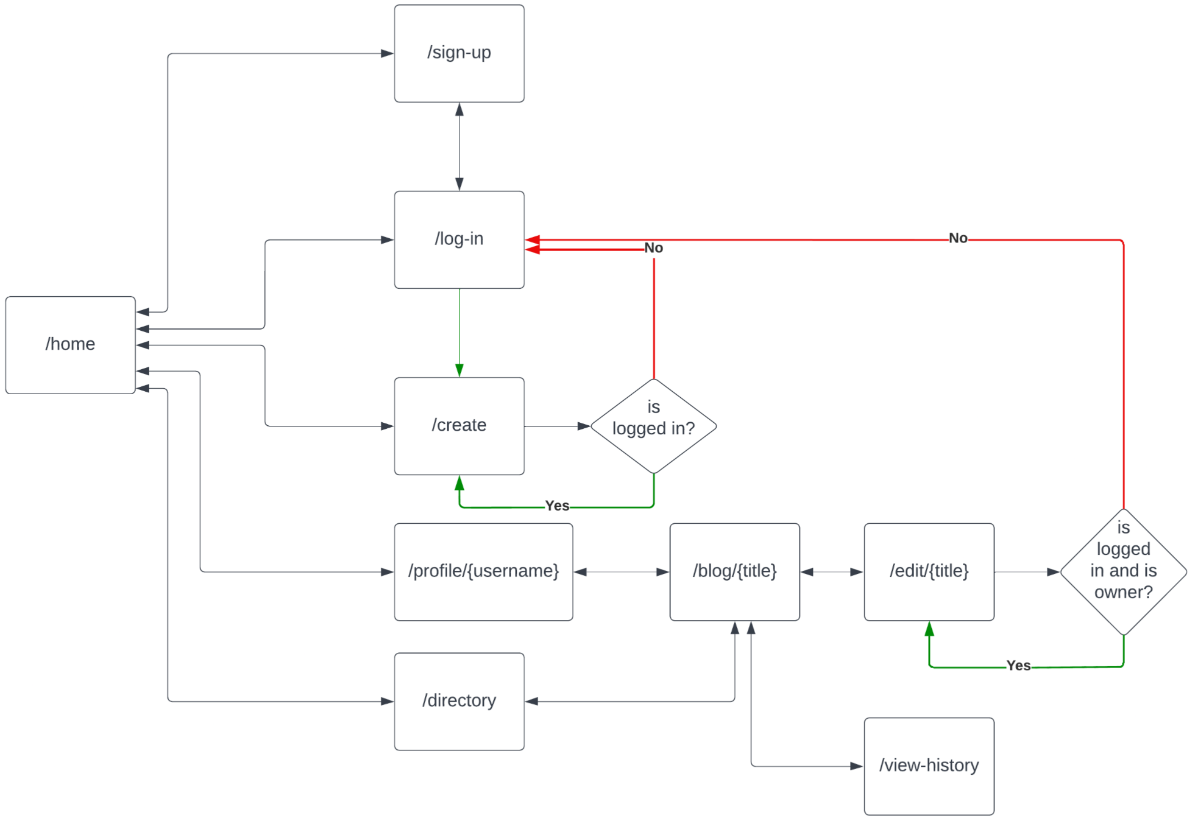
1. Check if the user is logged in and is the owner of this blog
   a. Yes, process to step 2
   b. No, direct user to log-in page
2. When the user submits, makes a new entry into BLOGS with the same username and title, but different content and timestamp

Directory (Wanying)

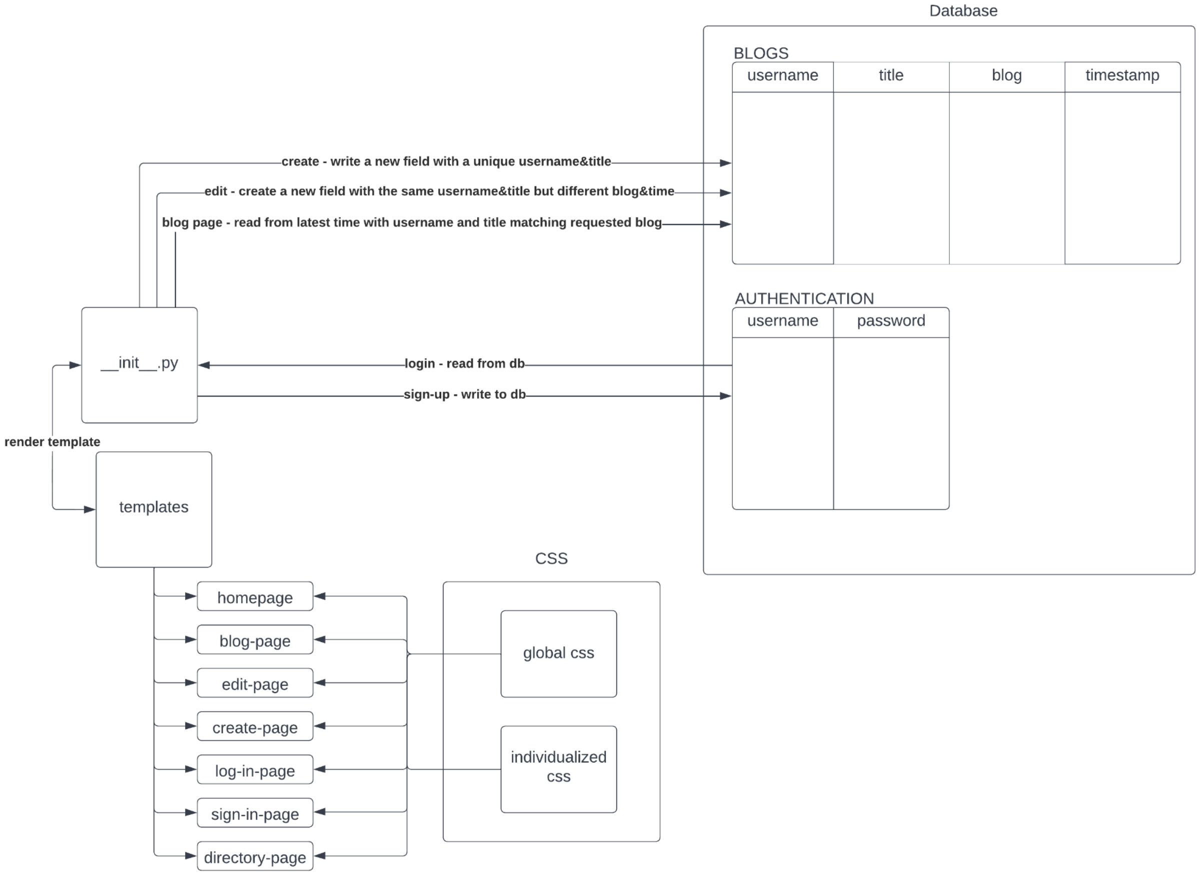
Anyone can access/view blogs through this page/route

- Includes
  - an alphabetically sorted list of blog titles that will link username to the blog corresponding to the title

Site Map


/sign-up
/log-in
/home
/create
is logged in?
No
No
Yes
/profile/{username}
/blog/{title}
/edit/{title}
is logged in and is owner?
Yes
/directory
/view-history

Component Map



Database

BLOGS

username	title	blog	timestamp

AUTHENTICATION

username	password

create - write a new field with a unique username&title
edit - create a new field with the same username&title but different blog&time
blog page - read from latest time with username and title matching requested blog
__init__.py
login - read from db
sign-up - write to db
render template
templates
homepage
blog-page
edit-page
create-page
log-in-page
sign-in-page
directory-page
CSS
global css
individualized css