Devo Council : Alex Luo, Tawab Berri, Nia Lam, Ivan Gontchar
P00: Scenario Two
TARGET SHIP DATE: 2024-11-08
Devo council, unite!

Scenario Two: Your team has been contracted to create a we**b log** hosting site, with the following features

- Users will have to register to use the site.
- A logged-in user will be able to
    - Create a new blog
    - Update their blog by adding a new entry
    - View and edit their own past entries
    - View the blogs of other users

Design Document Specifications:

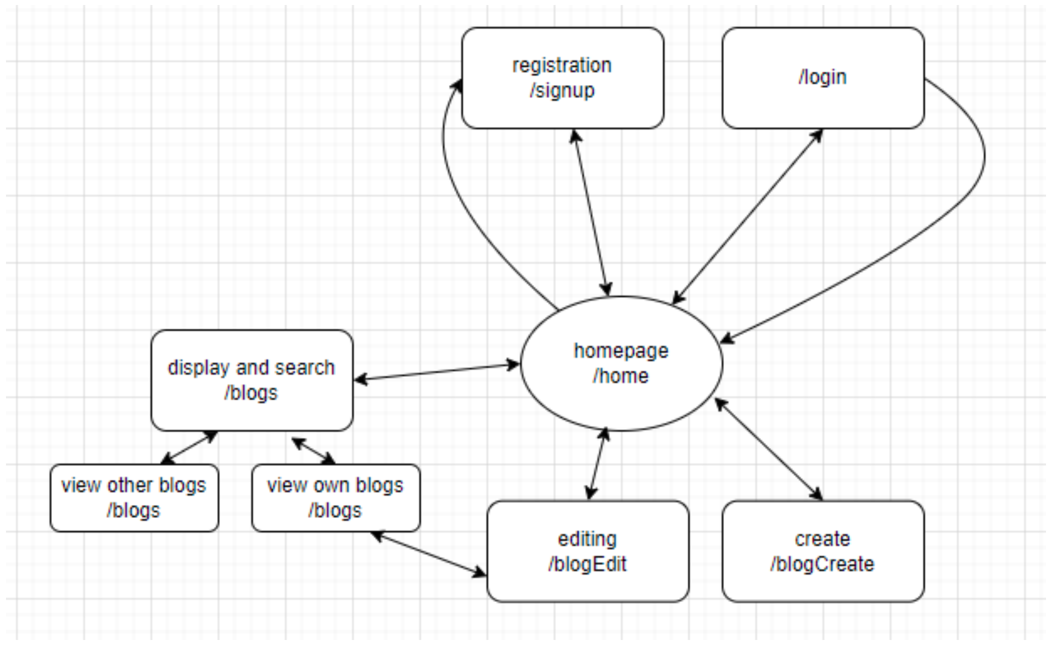
Site Map/Site Preview:

Home Page
- Access control: the home page is only accessible to logged in users. Non-logged-in users will be automatically redirected to the login/registration page if they try to access it.
- Welcome banner: a "Welcome, [username]!" banner for logged in users, to indicate that they are currently logged in.
- Navigation links: link to the user's **own blog** page, where they can view and edit their past entries, a link to **other blogs**, a page displaying blogs from other users, and a **logout** button.
- Search bar: positioned at the top, the search bar will allow users to search for other users' blogs, their own blogs, or specific blog entries. The search opens a new page that displays the results based on the search query.
- Search results: results page displays matches based on keywords in titles, content, or author names.

Registration/Login Page
- Form fields: fields for entering username and password
- Redirection: after successful login, user will be redirected to the home page
- Error handling for incorrect login attempts

Personal Blog Page
- Create new blog button to start a new post
- Display drafts and published posts separately, tags indicate status
- Past entries : each entry is listed with options to view, edit, or delete.

Other Blog Pages
- grid/list view of blogs from other users with usernames for differentiation



Database Organization:

- Login data storage will be similar to in-class session activity
- If the user is the author of a blog, they will have access to add a new entry onto said blog, as well as view past entries.
- This means that blog will have to be defined as completely random keys (os.urandom(32)) and a user will have possession of the keys of the blogs that they are allowed to edit (they are the author of these blogs)
- If the user does not possess the private key of a blog they are viewing, they do not have edit permissions on said blog nor do they have permission to view past entries of this blog.
- Creation of a blog will consist of creation of a new private key that refers to said blog as well as linkage of said private key to the creator of the blog (currently logged-in user)
- Viewing of past entries will require storage of said entries, so maybe we could store entries in a list (connected to the secure private key given to the user), and append new entries to the list as the user creates them, as well as access older list items (entries) upon the user's request

Program Components:

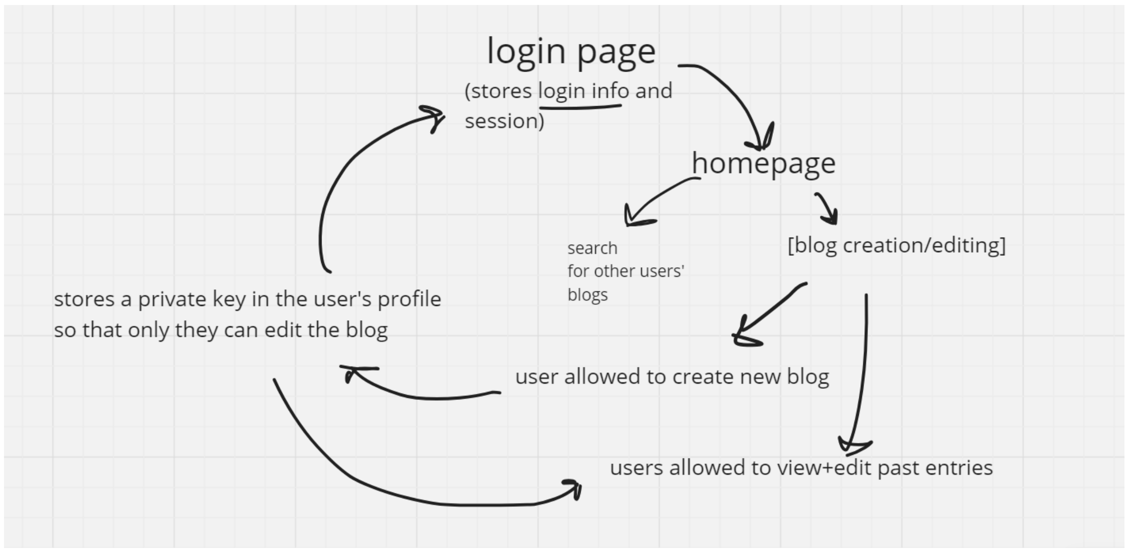
1. blog.py: this will be the main python file that we will run to have it load all the other components of our project. It will be responsible for handling user input, storing users, and redirecting and rendering pages
    a. To handle permission levels, we will have one default account with username "admin" and a predetermined password that the highest admin would have. From

that account, they can add or remove other mods. Mods can remove posts, and they must submit a brief reason why.

   b. Removed posts are not permanently deleted, just hidden and will not be displayed in the list of all users' posts, but will show up in the user's personal page with some sort of differentiation (in a different color, under a different section, etc). A user can appeal a blog removal to the highest level admin, with a short reasoning. Admin will have removed blogs at the top of the blog list, and they will have the option to add it back or keep it removed

2. home.html: this will be our template to display the landing page, where the user can navigate the website, as described in site map, as well as logout/view account
   a. Underneath the welcome message, there will be a search bar. Whatever the results of the user's search are, upon clicking enter, will be redirected and shown in blogSearch.html
3. login.html: the template that is generated when a user is attempting to log in or register an account.
4. logout.html: the template that is generated when a user is logging out of their account, to ask them as a last step if they are sure.
5. blogs.html: this template will load all the user's blogs from the databases and will be searchable for specific ones.
6. editing.html: template for being the user interface where one can choose if they want to edit one of their past blogs or make a new one
7. blogView.html: template for the page that will display a certain blog that a user has clicked on to view, whether that is from the total list, or from their personal page
8. blogSearch.html: the template that is generated when a user wants to look at all blogs on the site or search for a specific blog
   a. It will have a search bar at the top, with all available blogs underneath. As the user types in the search bar, it will eliminate the shown blogs below (only by titles that match user input) until it stops matching or user stops typing
9. blogCreate.html: template for the page where a user could be able to create a new blog entry with prompts for things like title, short summary, actual writing for the blog, etc.
10. blogEdit.html: template for the page where a user can edit their blogs. We will also use this same framework to load the info of a past blog they may have chosen to edit. The user can then submit their entry or changes, which will be updated in the database.
11. userData.db: database where we will have all our information stored on users: their usernames, passwords, blog entries, permission levels, and more
12. table users: table where we will store all our information on users with their usernames, passwords, permission level, and private keys (as mentioned above)
13. table blogs: table where we will store all our blogs with information about their general necessities, such as title, summary, content, if their hidden, etc, and also their private key, so we can see which user may, or may not access them.
14. style.css: makin' everything pretty

Linkage diagram of program components:


login page
(stores login info and session)
homepage
search for other users' blogs
[blog creation/editing]
stores a private key in the user's profile so that only they can edit the blog
user allowed to create new blog
users allowed to view+edit past entries

Assignments of each group member:

Tawab (PM): Back End(SQLITE/databases)
- Facilitate storage of blog private keys in user’s profile (to allow edit access and viewing of past entries)
- Facilitate storage of blog entries as list items connected to the keys in the user's profile
-

Ivan: Back End(SQLITE/databases)
- Facilitate storage of all blog entries in the table, with their unique attributes that will correlate them to given users, and store their information
-

Alex: Front End (HTML+CSS)
- Implement HTML structure and CSS styling for the home page
- Style welcome banner, navigation links, search bar, and indicator of login status
- Create layout for registration/login and personal blogs/other blogs
- Style search results page

Nia: Middleware (Flask, Python) + HTML
- Facilitate user sessions/accounts and permissions (admin or user, usernames and passwords, login/logout and signup)