Jake Liu, Bogdan Sotnikov, Joyce Lin, Naomi Kurian
Lost and Found
P00: Half-Quick
10/22/25
TARGET SHIP DATE: 2025-11-14

QCC: Should the user base and article base be consistent across multiple devices?

Program Components
Landing page - search bar, login button,
Navbar - help user navigate article
Search bar - search for articles
Login page - log in to access editing and profiles
User base - store users
User - store our user's information (username, password, articles authored/edited) or basically each row in the user base
Article base - store articles
Article page - displays article information from specified article
Profile pages - display user information
Logout page - log out of session
Edit Page - allows user to edit article contents

Component Relationships
Landing page
- Contains *search bar*
    - Utilizes *Article base* (through a case-insensitive search) to pull up an article
        - Potential feature: matching articles pop up following each letter
- Contains a random list of suggested articles (changes on refresh) that pulls from the *article base*
    - Displays nothing if empty
- *Navbar* will contain an 'publish new article' button (leads to *author page*) if user authenticated
- Checks if user is authenticated —> check if session is empty
    - If the user is, then display: a go to profile button (leading to the *profile page*), a logout button (leading to logout page) and maybe a 'hi, [username]!' —> in *navbar*
        - Uses existing username in session
    - If the user is not, then display a login button (leading to *login page*) —> in *navbar*
Login Page
- Asks for username/password and matches it to what is contained in the *user base*

- ○ Takes the username and compares it to existing *users*
    - If username does not exist, output 'username does not exist'
    - If username exists but the password does not match the one attached to the corresponding *User*, output 'password incorrect'
    - If username exists AND password matches, sends the user back to the *landing page* and adds the username to the session
- Contains a button to create an account (either another page or a checkbox)
    - ○ Once account is created, adds the new *user* to the *user base* and signs them in (sending them to *landing page* and adding the username to the session)
- Potential feature: instead of sending back to landing page, use session to hold a "last location" and send them there instead

Profile Page
- When go to profile button is clicked, match session username with username in *user base* and sends them to the path stored in the *user* (profile page)
- Displays pfp, 'hi, [username]!', associated *articles* (two separate lists for authored articles and edited articles), a back button (leads to *landing page*), and a logout button (leads to *logout page*) —> both buttons located in a *navbar*
    - ○ *Articles* will be pulled from the *article base* by checking if *user* is present as an editor or author for any of the articles, and the link will be included (location.href or something similar)
- Should include a button or something that allows you to choose from a curated list of profile pictures (while the current one is randomly selected when the account is first created)

Article Page
- Displays the title of the article at the very top, headers and content below, and then the editors and authors at the very bottom (structure may change based on whatever's aesthetically pleasing)
- *Navbar* contains a *search bar*, a logout button (leading to *logout page*), a back button (leading to *landing page*) and an edit button (goes to *edit page*) if the user is authenticated.
    - ○ If the user is NOT authenticated, navbar will only contain a *search bar*, a back button, and a login button (leads to *login page*)

Edit Page
- Through the usage of session (previous location, last accessed site, whatever), identify the *article* that is being edited
- Have the edit page be displayed in the same format as associated *article page* but each area that contains content will be replaced with a form response box
    - ○ Form response box will be prefilled with original content
- Page will contain a submit button which will prompt a comparison of original content and submitted content

- ○ Whichever section that has changed will be updated in the corresponding variable stored in the *article*
- ○ Will send user to corresponding article with changes published
  - ■ Once you go back, the *user* currently in session will be saved to the editor list of the corresponding *article*
- In the *navbar*, only a back button will exist that'll send you back to the article you were originally on (changed are unsaved)
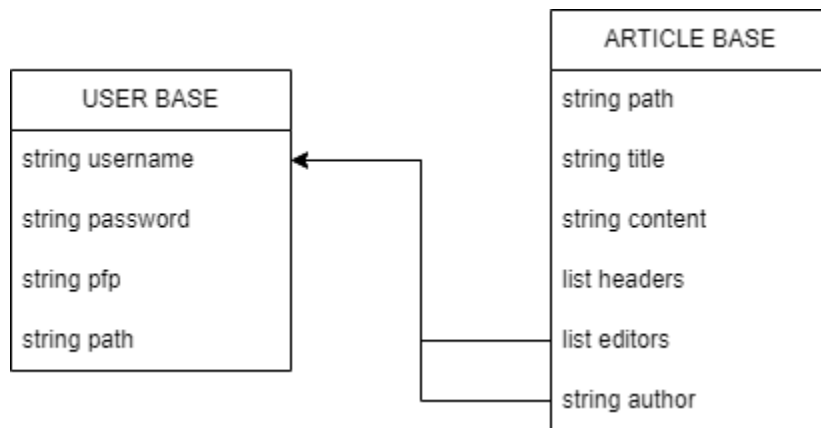
Author Page
- Works similarly to edit page except all form response boxes are blank
  - ○ All response boxes correspond to a variable and will be stored as entered
- Submission (using the submit button) will send you to the newly created article
- In the *navbar*, only a back button will exist that'll send you back to the article you were originally on (changed are unsaved)
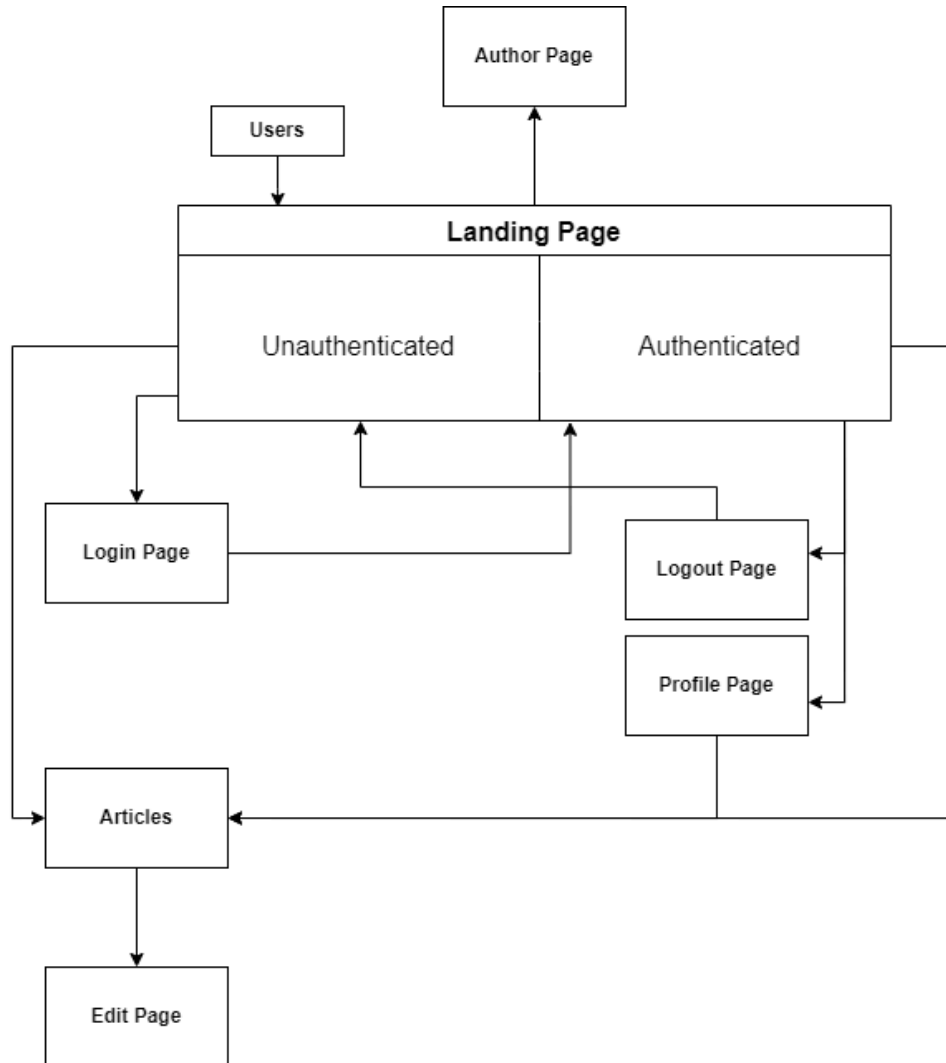
Logout Page
- Removes username from session and contains a button that sends you to the *landing page*
  - ○ Include a goodbye message

Database Organization
sqlite tables will contain the following information:

ARTICLE BASE
- string path
- string title
- string content
- list headers
- list editors
- string author

USER BASE
- string username
- string password
- string pfp
- string path

Site Map

Task Breakdown

Team member	Tasks
Jake Liu	Login page, Edit Page
Bogdan Sotnikov	Landing Page, Author Page
Joyce Lin	Profile Page, Logout Page