RohanSpace by HeistLEET

TARGET SHIP DATE: 2025-11-10

# Scenario Δ

entries

| | | |
|---|---|---|
| int | user_id | |
| string | title | |
| string | post | |
| date | timestamp | |
| date | last_edit | |

account

| | | |
|---|---|---|
| string | username | PK |
| string | email | |
| string | password | |
| string | first_name | |
| string | last_name | |
| string | img_path | |

Components:

- SQLite file containing the table **accounts** to store users of the wiki and their appropriate data, the table **entries** to contain entry data, and the **history** table to contain the records of who has changed each wiki page and when.
- Routes:
    - /sign-in, verifies values against **accounts** table, redirects to / with cookies when done, uses **fetch_creds()**
    - /register, uses **fetch_creds**() to verify account does not exist, adds values to **accounts** with **set_creds**(), redirects to sign-in
    - /, Jinja2 search bar that uses **fetch_page()** to search for users and redirect to their pages
    - /account, displays information about a user, uses **fetch_creds**() as well as **fetch_posts**().
    - /account/edit **post()** is used to upload a new post, and **update_post**() is used to update past posts.

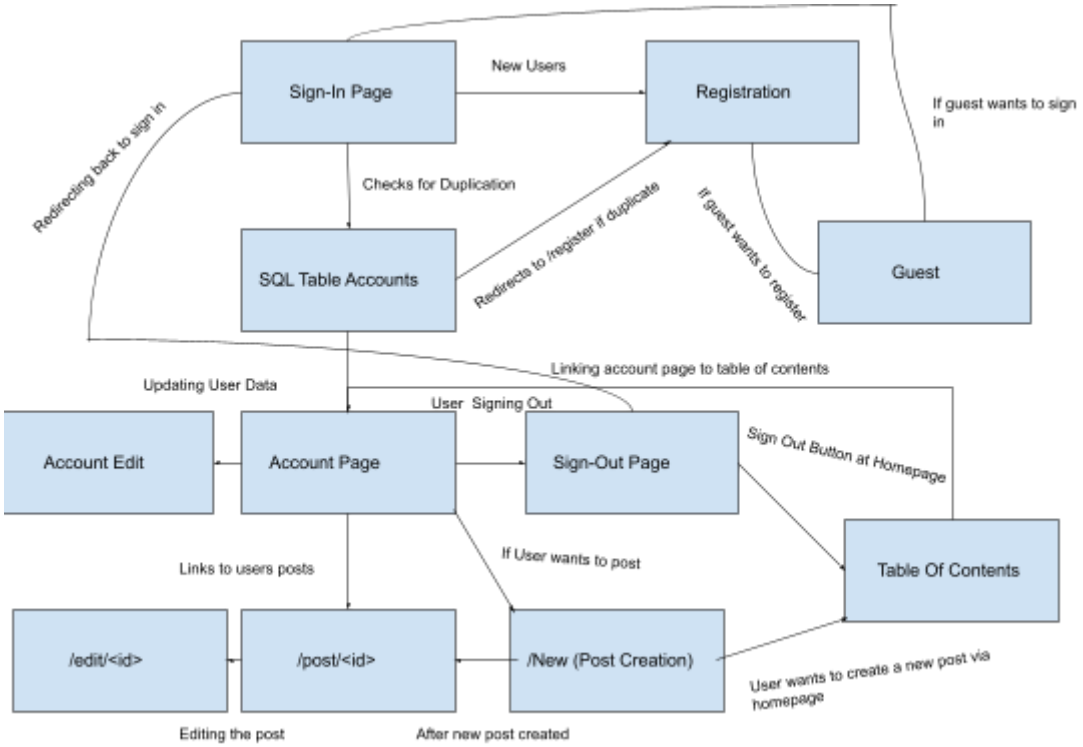
- /<page>, uses **update_creds**(), allows user to change their account details
- /new
    - Form for logged-in users to create a new blog post
    - Checks via **fetch_page()** if page already exists
    - If not: saves via **post()** into **entries** table
    - Redirects to **/page/<id>** after submission
- /post/<id>
    - Displays full blog post (title, author, date, and content) from **entries** table
    - If the user is logged in and the page is theirs they are shown an “Edit Post” button that redirects them to **/edit/<id>**
    - Guests can only view the page
    - If page doesn’t exist then redirect to **/**
- /edit/<id>
    - Displays a full blog post (title, author, date, and content) from entries table.
    - Allows logged in users to edit and submit updates
    - Updates the **entries** table upon submission via **update_post()**
    - Redirects to **/page/<title>** after a successful update

- Helper Functions:
    - **fetch_page**(page), returns a dictionary containing the values needed to populate a page from the sql, returns false if the page does not exist
    - **post**(...), enters a new record into the **entries** table
    - **update_post**(...), updates a record from the **entries** table
    - **fetch_creds**(username), returns dictionary of user information, if the user isn’t signed in then this is bypassed
    - **set_creds**(...), sets values to **accounts** table
    - **update_creds**(...), updates values from **account** table

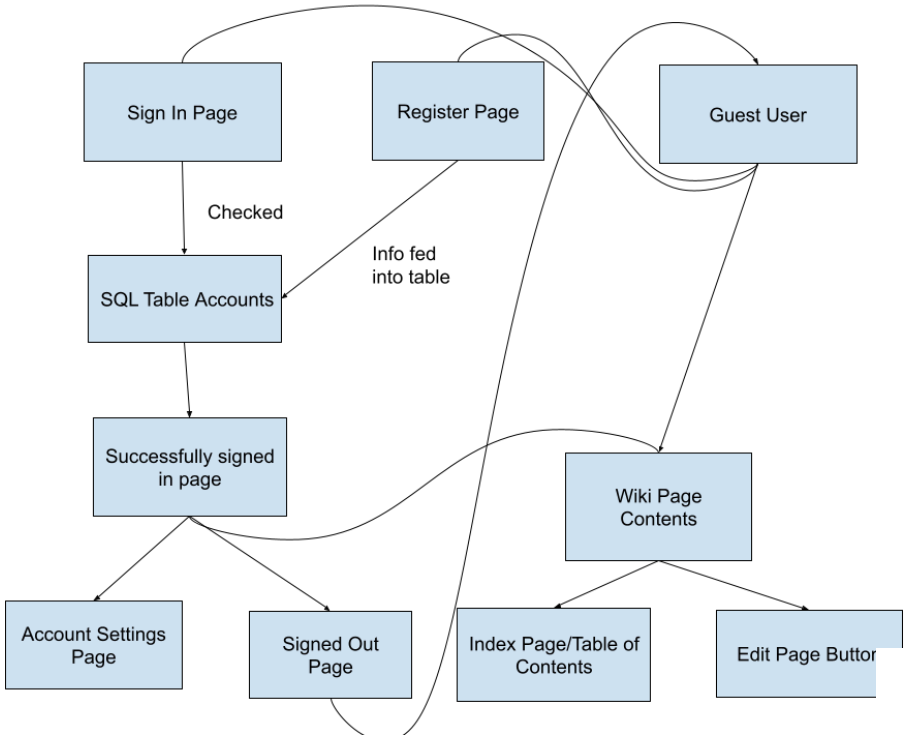
Site Map:

- /sign-in
    - Cookie functionality as we created in class for assignment #16
    - Account credentials should be checked against the SQL table **accounts** from this route
    - Link to /register for new users
    - If account credentials are verified, we redirect to / with cookies set
    - If account credentials are invalid, we reload with an error message
- /register
    - Again, functionality as created in prior assignment
    - Credentials fed into the **accounts** table, registering their account details
    - Redirects to /sign-in  upon successful registration
    - Validation in the accounts table (preventing duplicate accounts)
- /
    - Displays all existing blogs (title + author)
    - Greets user ("Welcome, NAME") and provides buttons for:
        - "Create New Blog": Links to /new
        - "View My Posts":  Links to /account
        - "Sign Out": Links to /sign-out
- /account
    - Displays profile information (Username and Gmail)
    - Lists all pages this user has created or edited
    - Each listed page title links to **/page/<id>**
    - Buttons for **/sign-out, /new, /account/edit**
- /account/edit
    - Allows users to update username, password, or email
    - Redirects to **/account** after save
- /new
    - Form for logged-in users to create a new blog post
    - Redirects to **/page/<id>** after submission
- /post/<id>
    - Displays full blog post (title, author, date, and content)
    - If the page is theirs they are shown an "Edit Post" button that redirects them to **/edit/<id>**
    - If page doesn't exist then redirect to **/**
- /edit/<id>
    - Displays a full blog post (title, author, date, and content) from entries table.
    - Allows users to edit and submit updates
    - Redirects to **/page/<title>** after a successful update
- /sign-out
    - Clears user cookies/session
    - Redirects to **/sign-in**
    - Displays "You have successfully signed out."

User Flow Map:




Program Components:



## Task tracker

| Assignee | Title | Date | Status |
|---|---|---|---|
| Rohan Sen | "Install Guide" + "Launch Codes" section of readme | Oct 28, 2025 | Not started |
| rohans33@nycs… | A *list* of program components with role of each specified. (e.g., a car engine is comprised of various components: carburetor, alternator, radiator, spark plugs, etc. Each must perform its role for the engine to do its overall job.) | Oct 27, 2025 | Completed |
| Michelle Chen | Explanation of how each component relates to the others.<br>• Component map visualizing relationships between components. | Oct 27, 2025 | Completed |
| Aoanul Hoque | Database Organization (tables? Relationships b/t tables? etc.) | Oct 27, 2025 | Completed |
| Haowen Xiao | Site map for front end<br>• Represent each page you envision for your | Oct 27, 2025 | Completed |

## Task tracker

| Assignee | Title | Date | Status |
|---|---|---|---|
| | site.<br>• Show linkages conveying all possible pathways for a user traversing site. | | |
| michellec397@… | A breakdown of the different tasks required to complete this project<br>• Include assignments of each task to each group member | Oct 27, 2025 | Completed |
| Rohan Sen | Append this line to your heading: **TARGET SHIP DATE: {yyyy-mm-dd}** | Oct 23, 2025 | Completed |