Tiffany Yang, Claire Song, Jessica Yu, Jackie Zeng
Team Anything
SoftDev
P00
2024-10-23
Time spent:
Target Ship Date: 2024-10- 24

Scenario Two: *Your team has been contracted to create a web 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*

Front end:

- ☐ HTML templates and CSS (if time): create layout of web pages; display tables and information from databases

Back end:

- ☐ Flask app: host web pages
- ☐ Database handling: create and link databases; sort data

Components:

- User accounts
    - User sessions
    - User login/logout system
- Routes
    - Route to submit new blogs
    - Route to edit past blogs
    - Route to view past blogs
- SQLite db for storing blog info
    - Page_id
    - Usernames/logins
    - Password hash

Routes:

Home (/):
    List of blogs
Login (/login):
    Login with sessions and keys/cookies
Viewing blog page (/{username}/{page_id}):
    Displays a single blog page
Create Page (/create):
    Goes to a form to create a new blog page
Signing up: (/signup):
    Allows you to make an account and saves username & password information
Edit: (/{page_id}/edit):
    Allows you to edit an already created blog
Viewing past entries: (/viewpast):
    Shows list of past entries stored in database
Blog: (/{username}):
    Shows list of blogs made by {username}

Front end: Claire and Tiffany	Backend: Jessica and Jackie
☐ newPage.html - *{page with forms and runs forms.py upon submission; form answers = blog.db record and fields }* ☐ editPage.html ☐ signUp.html ☐ login.html ☐ home.html ☐ blog.html (list of past entries) ☐ userBlogs.html (view the blogs of other users)	☐ blog.db (stores page data) ☐ app.py (main script) ☐ forms.py (generates the new pages) ☐ logins.db (stores login data) ☐ edits.db (stores past edits)

app.py:

(‘/’)

render home.html

('/signup)

If [logged in]

Return <logged in message>

If [logged out]

Return signup.html

('/login')

If [logged in]

Return <logged in message>

If [logged out]

Return login.html

('/{username}/{page_id}')

render page_id.html

('/create)

Render newPage.html

(‘/editpage’)

Render editPage.html

(‘/blog’)

Render blog.html

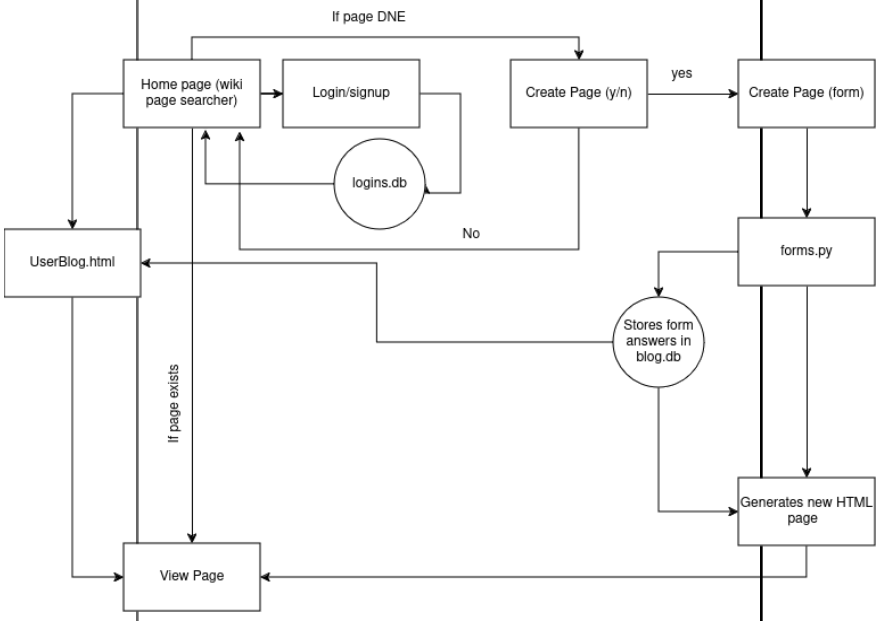
(‘/{username}’)

Render userblog.html

forms.py:

Stores form information into the database

Takes the information from the database to fill in the rendered page



Database Organization:

Blog.db

Username or User ID	Page ID	Page Title	Page details (multiple copies of this field if needed)
Stores the username or ID of the user currently logged in who made the blog post	Stores a generated ID of the page for easier navigation	Rendered title or page details	Rendered title or page details

Logins.db

Username	Password	User ID
Stores the username of the user	Stores the password for the user	Stores the generated ID (or possibly unique user key)

Edits.db

Page ID	Page Title	Past Page Details	Current Page details
Stores a generated ID of the page for easier navigation	Rendered title or page details	Prior versions	Current version

Page Draft

<table>
<tr>
<td>

Home (Page name)

Home > **Users (dropdown of letter ranges)** **My Pages**

Login	**Sign-up**

Possible search bar?

</td>
<td>

Users

Home > **Users (dropdown of letter ranges)** **My Pages**

A - L

User1
User2
User3

</td>
</tr>
<tr>
<td>

My Pages/User Pages

Home > **Users (dropdown of letter ranges)** **My Pages**

Page1
Page2
Page3

</td>
<td>

Page1 [Title]

Home > **Users (dropdown of letter ranges)** **My Pages**

Page Details here [if own page, edit button here]

[if editing, pre-filled textbox with previously saved details here]

</td>
</tr>
</table>

Site Map:

Home (/)

- Outgoing routes:
    - Login (/login)
    - Sign-Up (/signup)
    - View Blog (/username/{page_id}) for viewing a specific blog

Login (/login)

- Outgoing routes:
    - Home (/) if successful login
    - Sign-Up (/signup) if the user doesn't have an account

Sign-Up (/signup)

- Outgoing routes:
    - Login (/login)

View Blog (/username/{page_id})

- Outgoing routes:
    - Home (/) to return to the main blog list
    - Edit Page (/{page_id}/edit) if the user is logged in and wants to edit their own post

Create Page (/create)

- Outgoing routes:
    - Home (/)
    - User Blogs (/username) to view all their created blogs

Edit Page (/{page_id}/edit)

- Outgoing routes:
    - View Blog (/username/{page_id}) after editing is done
    - Home (/) after editing is done

View Past Entries (/viewpast)

- Outgoing routes:
    - View Blog (/username/{page_id}) to view specific entries
    - Edit Page (/{page_id}/edit) if editing past entries

User Blogs (/username)

- Outgoing routes:
    - Create Page (/create)
    - View Blog (/username/{page_id})

Task Assignment:

- Backend: Jessica and Jackie
    - Responsibilities:
        - blog.db (storing page data)
        - app.py (main script)
        - forms.py (generating new pages)
        - logins.db (storing login data)
        - edits.db (storing past edits)
- Frontend: Claire and Tiffany

- Responsibilities:
  - newPage.html (page with forms, runs forms.py on submission)
  - editPage.html (for editing pages)
  - signUp.html, login.html (user sign-up and login pages)
  - home.html (home page)
  - blog.html (list of past entries)
  - userBlogs.html (viewing blogs of other users)