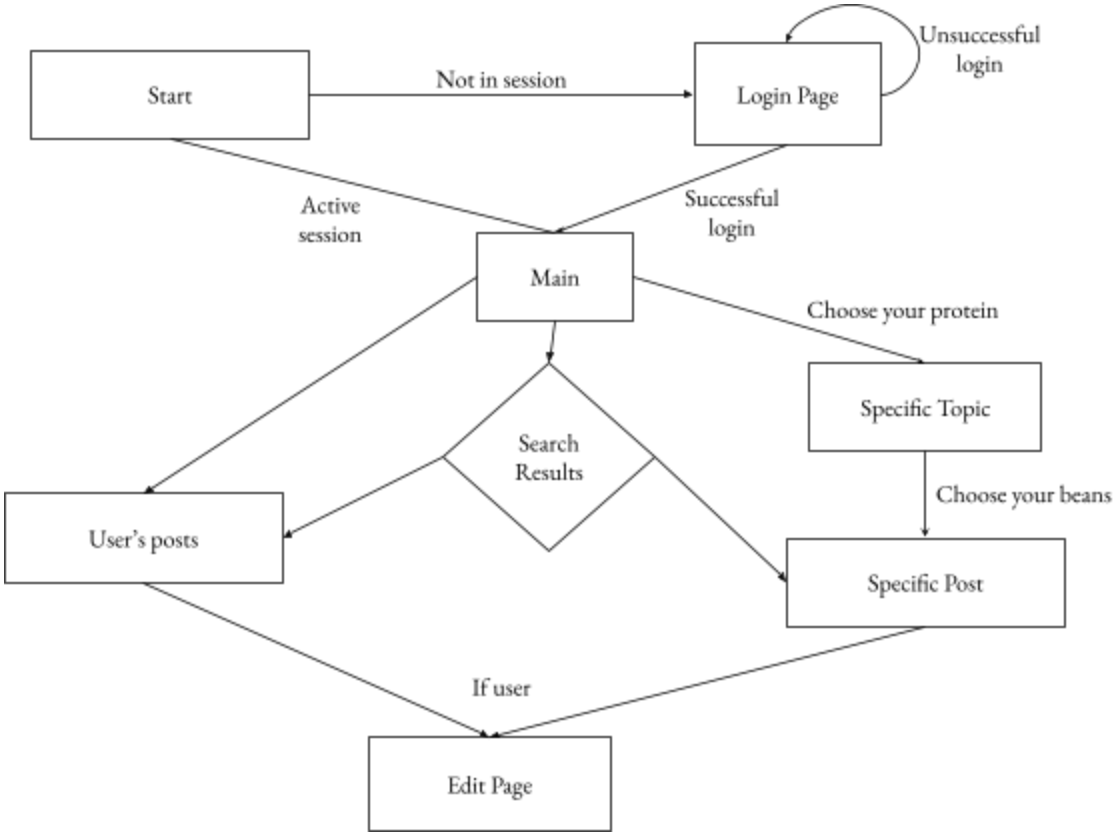
ETSSettera - Ella Krechmer (PM), Tina Nguyen, Sean+Patrick Ging, Shyne Choi
SoftDev
P00 - Move Slowly and Fix Things - Project Design Document
2021-10-27
Time Spent: 75 min

*Scenario Two: Your team has been contracted to create a we**b log** hosting site.*

Program Components

- Landing page is the login page
- Main page that shows different topics and a site introduction
- Heading with different topics (relating to Stuy culture)
    - Posts will be classified under topic(s)
- Adding/editing/viewing entries
    - Can only edit your own entries
    - All would be separate pages
- User info
    - Sidebar with username and current posts
    - Users can press the title of one of your current posts to enter it or just look at all their posts at once (e.g. we would have something like “view all”); when you get to your post there will be an option to edit it
    - There will also be a place to add a new post (in both main page w sidebar and all posts page)
- Search bar
    - For username
    - For topic
    - For entry title
- Checks if session is active
    - If somebody tries to look at a specific post and just goes to that site right away, they’ll still be prompted to log in unless they are already logged in
    - Can logout if “logged in”: pops session
- Stores the posts that the user has already made
    - The user will be able to go to their posts from their sections and view or edit them
    - If no posts are started, begins with a blank editable post
    - Can create new

Relationship Between Components




We will be accessing the database when logging in and when looking at posts, which will allow us to access all the info but also to check whether or not a user can access the specific page.

Database Organization

Tables for:

- ~~Username : blog posts~~
- ~~Username: password~~
- Username : password : [blog title: blog post]

Site Map

- "/" → landing (login) page
- "/main" → once logged in, the user will see a page with an introduction, header with links to different topics, and a way to look at their account and view/edit old posts or add new ones
- "/<topic>" → main page will link to a topic page, which will show all the posts that are under that topic
    - "/<topic>/<post>" → this will show a specific post on its own, <post> will be name/number of the post

        - If the specific post is your own, you will be able to edit it
- "/myposts" → this will show the user all their posts
- "/edit<post>" → this will allow the user to edit a post; we can also use this when they add a post, except the post they're editing will just start out blank

<u>Tasks</u>

- Database
    - Add info into username, password database
        - Check for existing info
        - Add new info (from sign up) (no repeat usernames allowed)
        - Use for log in, check if pair works
            - Return wrong pass
            - Enter username
            - Enter pass, enter matching username and password
    - Database for text for blog
        - Update - check if there is a saved one
        - Add new with [username, password], text
- HTML
    - Show text for blog
    - {{username}}, {{text}}, {{header}}
- FLASK
    - Add to database, rid of it, check info in database, use info from database for HTML (to display the blog)

*Sean's Tasks*

- Database stuff, tie it into login
- Create, add blog file into database, use data in HTML to show blog (frontend)
    - Flask, Sqlite (backend)

*Ella's Tasks*

- Html and formatting
- Moving between pages
    - e.g. after you login it takes you to the main page, and so on
- PM duties

*Shyne's Tasks*

- Username/Password logins
    - Making sure passwords match the username
    - Creating a new user/pass login (no repeated usernames)

- Accessing posts under different topics

*Tina's Tasks*

- Add search bar
- Make posts editable for the author and uneditable for another user
- Save the posts under the user
- User session technicalities (popping sessions, prompting logins, etc.)