Team Zeroes -- *XINHUI XU, NICOLE CHENG, SHAMAUL DILMOHAMED*

tldr;

This project will contain a homepage with links to the blog(s), an about page that is formatted with information given by the client as well as a link back to the homepage, and a database used to contain username and password combinations, as well as storing user blog posts. The database management system used will be SQLite.

PROGRAM COMPONENTS LIST:

REGISTER: [register.py]
- [db] Add information to db
  - user/pass, bio, pfp, urls, etc.
- Redirect to home / login [login.py]

LOGIN: [login.py]
- [db] Database processing for emails / passwords
- [Flask] Session (encrypted)
- Redirected to either their home blog or to their feed

PROFILE:
- Done upon registration [registration.py]
- [db] Create a "profile" for each user
  - blog url
  - Subscribers / stats
  - Bio / profile picture
- [db] Editing profile
  - Edit db fields
    - Can probably use an altered function from [register.py]

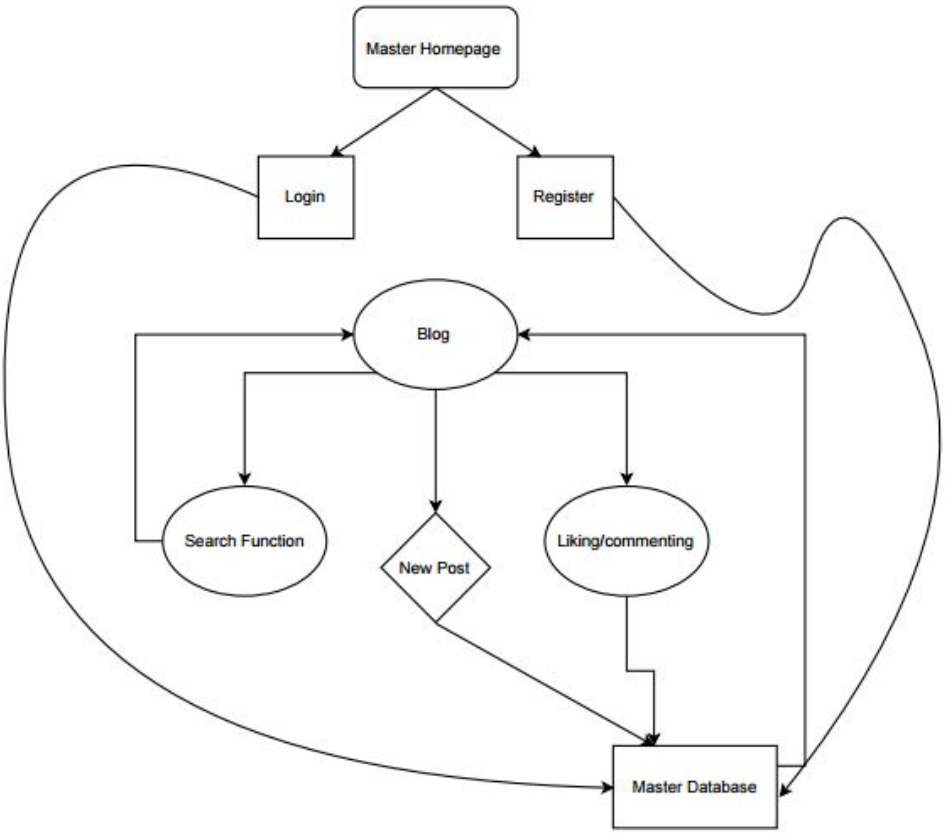
UPLOAD:
- [db] Blog post (can contain media) [upload.py]
  - Store blog posts (Table)
  - Comments on blog posts (Table)
  - Basics: post title, timestamp, author, etc.

LOGOUT:
- [Flask] End session [logout.py]
  - [db] Maybe: keep timestamps of logging in / out
  - If logged out, redirect to home / login page
  - Perhaps end session upon closing of browser

EXTRA:
- [db] User direct messaging
  - Create a MESSAGE table containing timestamp, sender, recipient, content
  - Push notifications for unread messages
- [Flask] List of bloggers "online"

- [db] Friending / favoriting blog(ger)s
- Blog theme
- [db] Tags
    - Browse blog[ger]s by tag and/or category
- Blogs can be toggled to be public or private
- [Flask][db] Timestamps

COMPONENT MAP:

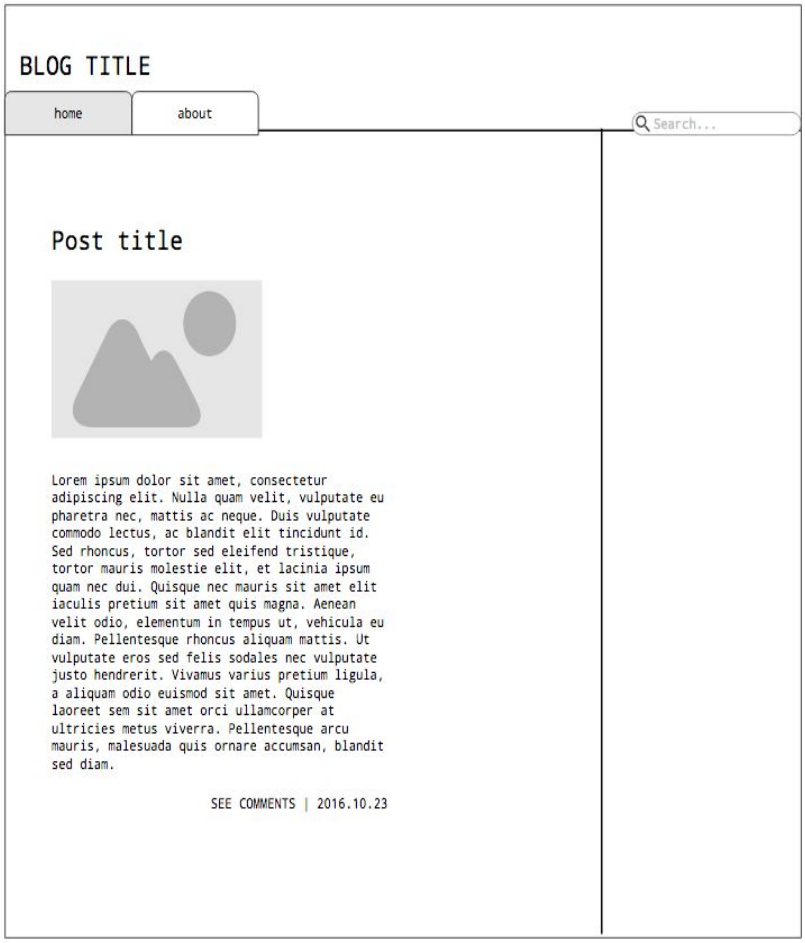


DATABASE SCHEMA:
(we will be implementing relational databases -- linking tables)
TABLE MAP DRAFT: (*<TABLE>: field1,)*
    USERS: email, username, hashed pw
    BLOG: email, blog url, profile url
    HOME: post id (newest first)
        POST: title, content, post id, # of like/share
        COMMENTS: post id, comment content, commenter url
    PROFILE: profile picture, bio, <any other personal info field>

How they all tie in: USERS has all of the basic info. BLOG contains the urls to the blogs, linking each one to the user. HOME has the actual blog posts and comments, with POST containing each blog's posts, id, and stats; there will be a link to COMMENTS on each blog post page. COMMENTS will contain the post id to link the comment to the correct post, as well as the actual content of the commenter.

SITE MAP:

(login page)



(blog page)

TASKS ASSIGNMENT:

SHAMAUL:

- Create a database and format it to take in all forms of data we need to store
    - Media
        - Audio as blog background music
        - Images / gifs accompanying blogs and profiles
- Deal with changes to the database (i.e. blog edits made after uploading)
- Keep all tables impeccably linked

NICOLE:

- Create a flask app to connect with the homepage and the database in order to handle logins and uploading files
    - Hashing passwords
    - Forms and forms and forms
        - Take information and jam them into the database(s) that Shamaul made
    - Correctly storing uploaded files in archive folders
        - Record their [relative] locations in the database
        - Reject oversized files

XINHUI:

- Organize uploads and rendering them -- overall visuals and front-end
    - Html / css templates
    - Desktop to mobile config
    - Resizable components to match all screen sizes
    - Ensure a user-friendly interface
        - Maybe: Forgot your password? Password recovery service