QAFFLE

by JoséMad

0. Roster

Michael “Bob” Zhang (PM)
Michael “William” Lin
Pratham Rawat
Jesse “McCree” Chen

1. Project Description

The QAFFLE is the next generation solution to the old and bland QAF. Similar in premise, the QAFFLE allows users to create posts and comment at its most basic level. The frontend will be sleek and sexy. The backend will be snappy and robust. We will be implementing features not offered in the original QAF to enhance the user experience.

We have elected to use *Foundation* as our Front End framework, due to its wider range of features and easier integration with more complex thinking such as

Core Features of Minimum Viable Product

- Create an account and sign in/log out
- Brew (make), tag, edit, and delete posts
- Comment on a post
- Upvote/downvote posts and comments

Additional Features for Complete Final Product

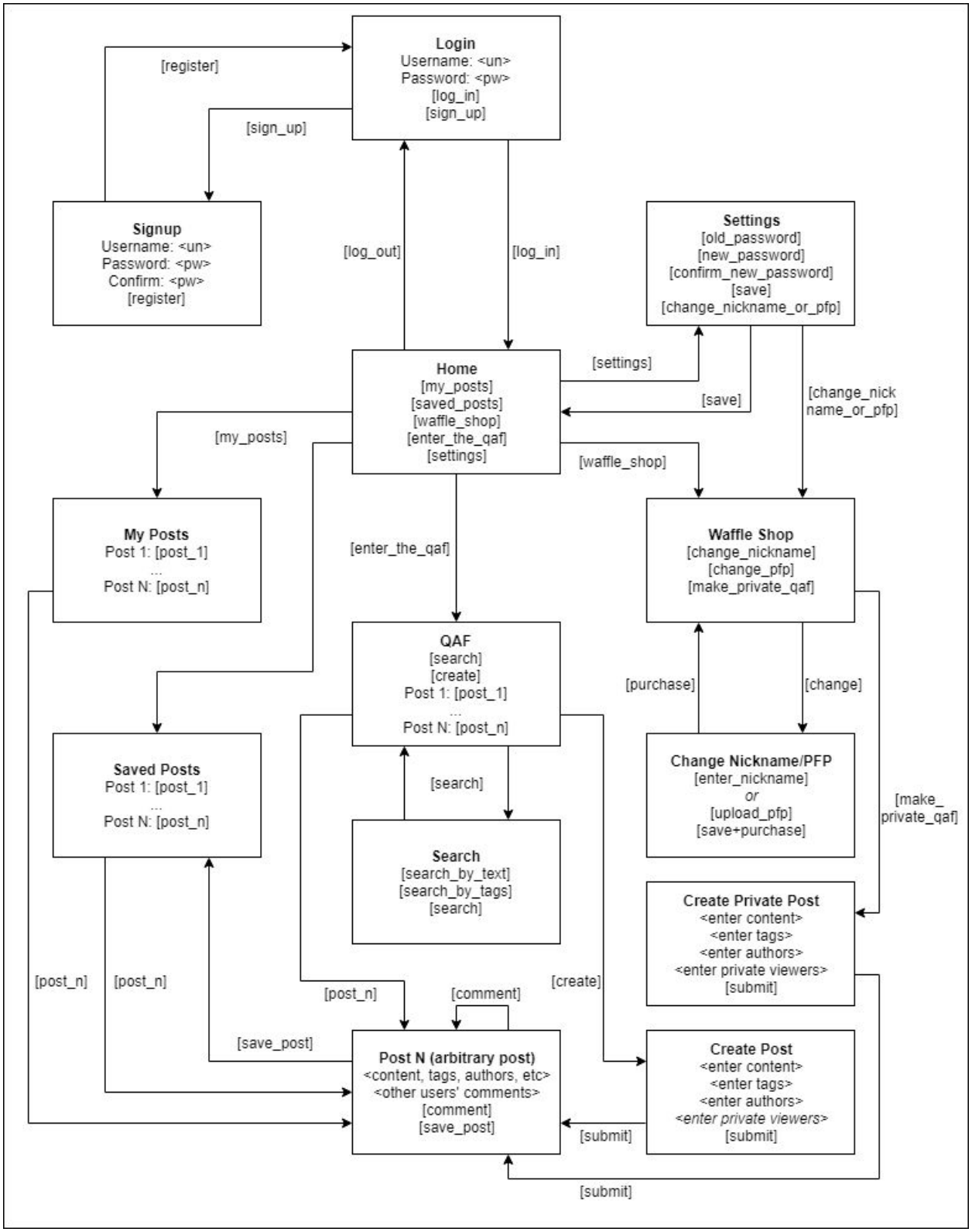
- Search for post by title, content, or tags
- Waffle point system:
    - Every like gets the user one waffle, a form of currency
    - Waffles may be used to redeem rewards and features
    - Waffles can also be used to flex, and will be displayed when clicking on someone’s profile
- Redeemable features (one time use):
    - 10 waffles:
        - Private QAF (post one QAF where you choose who can see it)
            - The purpose of this would be for use cases like our own class QAF, where we would not want random people on the internet to interrupt our discussions.
    - 20 waffles:
        - Change displayed name
        - Change profile picture
    - 99999999 waffles:
        - AirPods next to your profile picture

- Save posts and view saved posts in homepage
- Share author permissions
    - When creating a post, you can denote who else are authors, so they can edit the post as well.

Miscellaneous Features Under Consideration

- Email notifications for traffic (can be turned on or off per post)
- Hyping posts with redeemable “syrups” (maple, chocolate, caramel, etc)
    - Posts with most syrups show up on top
- Sexy JS animations for real time user interactions

2. Site Map


Login
Username: <un>
Password: <pw>
[log_in]
[sign_up]
[register]
[sign_up]
Signup
Username: <un>
Password: <pw>
Confirm: <pw>
[register]
[log_out]
[log_in]
Settings
[old_password]
[new_password]
[confirm_new_password]
[save]
[change_nickname_or_pfp]
Home
[my_posts]
[saved_posts]
[waffle_shop]
[enter_the_qaf]
[settings]
[settings]
[save]
[change_nick
name_or_pfp]
[my_posts]
[waffle_shop]
My Posts
Post 1: [post_1]
...
Post N: [post_n]
[enter_the_qaf]
Waffle Shop
[change_nickname]
[change_pfp]
[make_private_qaf]
QAF
[search]
[create]
Post 1: [post_1]
...
Post N: [post_n]
[purchase]
[change]
Saved Posts
Post 1: [post_1]
...
Post N: [post_n]
[search]
Change Nickname/PFP
[enter_nickname]
or
[upload_pfp]
[save+purchase]
[make_
private_qaf]
Search
[search_by_text]
[search_by_tags]
[search]
Create Private Post
<enter content>
<enter tags>
<enter authors>
<enter private viewers>
[submit]
[post_n]
[post_n]
[post_n]
[comment]
[create]
[save_post]
Post N (arbitrary post)
<content, tags, authors, etc>
<other users' comments>
[comment]
[save_post]
Create Post
<enter content>
<enter tags>
<enter authors>
<enter private viewers>
[submit]
[submit]
[submit]

3. Component Map

main.py

Contains all major routing between pages.
Handles integration between backend and frontend.

db_access.py

Contains methods for adding and updating tables entries, and tables themselves. This file will also have special use cases of each of those methods to create common constructs like tables for each QAF post, etc.

qaf_actions.py

For commonly repeated QAF actions, like posting, commenting, etc.

4. Database Design

Users table

id: int
username: string
password: string
waffles: int
qafs_joined: blob

Per QAF table of Posts: *(named after QAF ID)*

id: int
author_id: int
title: string
content: string
qaf_id: int
time_created: timestamp

Per 'QAF Post' table of Comments: *(named after Post ID)*

id: int
author_id: int
content: string
post_id: int
qaf_id: int
time_created: timestamp

QAF table: *(contains all QAFs)*

id: int
name: string
owner_id: int
is_private: string

5. Division of Labor

Member	Task
Michael “Bob” Zhang (PM)	PM duties: managing project calendar, updating design doc, and debugging
Michael “William” Lin	Routing, connecting backend to frontend.
Pratham Rawat	Sexy front-end and JS + helping with backend and routing as needed
Jesse “McCree” Chen	Back-end structure and maintenance *db.py* methods