lobo SoftDev Design Document

TARGET SHIP DATE: {2024-11-1}

Will Nzeuton, Daniel Park, Tim Ng, Yinwei Zhang

Blog - xbxlobog (“suh-buh(respirated/inhaled)-low-bog”)

Overview

We will be creating a blog platform where users can sign up, create, edit, and delete their blog posts. The site will feature categories for posts, comments, and the ability to view other users’ blogs. The focus of our blog will be food (recipes, information, etc.)

* “blog entry” is used interchangeably with “blog post”

Components

1. **User Authentication Module (UAM)**
   Role: Manages user sign-up, login, and logout functionality. To ensure that users would only be able to edit their own posts, users would have their own user ID. When we attempt to edit posts we would check if the current user ID matches the user ID stored with the post in the database and only allow editing if it matches.
2. **Post Management Module (PMM)**
   Role: Allows users to create new posts, edit existing posts, and delete their posts.
3. **Comment System Module (CSM)**
   Role: Enables users to comment on blog posts, edit their comments, and delete their comments.
4. **Category Management Module (CMM)**
   Role: Organizes blog posts into categories for easier navigation by users.
5. **Database Module (DbM)**
   Role: Handles all data storage and retrieval operations using SQLite3. Contains tables for users, posts, comments, and categories.
6. **Frontend Interface (FeI)**
   Role: User-facing component built with HTML (refer below) and CSS. Responsible for site layout and styling.

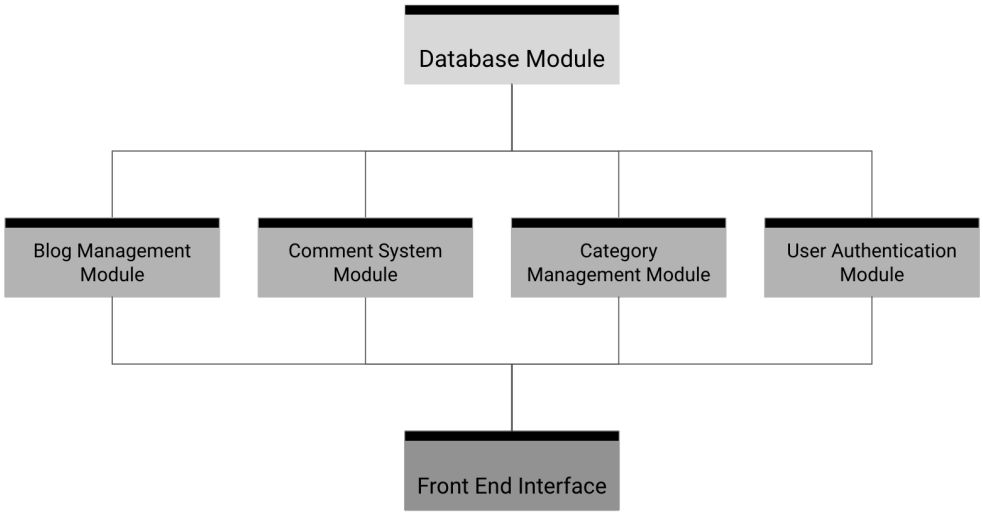
HTML Templates

HTML templates consist of
Homepage
the login page
sign up page
blog post page
post editing page (new post also uses this)
category page
user page

Component Relationships

- **UAM** interacts with the **DbM** to manage user credentials and Flask sessions.
- **PMM** communicates with the **DbM** to create, pull, update, and delete blog entries.
- **CSM** relies on the **DbM** to store and retrieve comments associated with each blog post.
- **CMM** categorizes posts stored in the **DbM** and enables filtering of posts.
- **FeI** integrates all modules to visually represent the data managed by the backend components.

Component Relationship Visualization:



Database:

Tables:

Users:

id	username	password	email
#	(abc) --------	(abc) ---------	-----@--.---

Categories:

id	title
#	(abc)-----

Note: Database Module will use id to link to respective categories, so posts added to existing categories will be automatically grouped within them.

Posts:

post_id	title	date_created	category	username	content
#	(abc)	(YYYY/MM/DD)—-	#category_id	(abc) --	<HTML>

Comments:

id	content	date_created	username	post_comment_is_under
#	(abc)—----	(YYYY/MM/DD)—------	(abc) --------	#post_id

Site Map

Site Map

- Home
  - Login page
  - Blog Pages
    - Blogs by Category
      - Individual Post
        - Comments
        - Edit / Delete
      - Create New Post
    - User Profile
      - My Posts

TASK ASSIGNMENTS

TASK	Will Nzeuton	Daniel Park	Tim Ng	Yinwei Zhang
Set up Flask and SQLite3 environment		**X**	**X**	
Build User Authentication Module, handle passwords, logins, etc.	**X**			
Build Post Management System			**X**	
Build Comment Management System	**X**	**X**		
Build Categories			**X**	**X**
Build Database		**X**		**X**
Frontend (HTML)				**X**
Frontend (CSS)				**X**
Final Testing and Bug Fixing	**X**	**X**	**X**	**X**