

SUMMARY DOCUMENT: 'We-Chat Discussion Forum'

Aryan and Joss

Walkthrough Document for We-Chat... "For Your Discussion Needs"

Regular User Walkthrough:

1. User goes to the website link on the server: <https://cosc360.ok.ubc.ca/joss/WeChat-Discussion-Forum/home.php>
 - a. ADMIN LOGIN: username: [admin24](#) password: [Test1234](#) email: [admin24@gmail.com](#)

REG LOGIN: username: [red](#) password: [Test1234](#) email: [red@gmail.com](#)

phpMyAdmin login: 80772049 for username and password
2. The user will be designated to the home page where they can click login. If they do not have an account, the user can click the 'sign-up' button to redirect them to the create account page.
3. The user fills out their credentials of their choice: Username, email, profile picture, password, and they have to confirm their password as well. Once completed, click 'create account' button.
 - a. User is led to the home page after account creation with their username displayed in the navigation bar
4. If the user already has an account, they simply go to the login page and enter their username and password and click login. This will allow them to enter the website.
5. If the user forgets their password, they can reset their password by clicking 'forgot password'. This prompts them to the reset password page where they enter their email. The email MUST BE VALID meaning they have to enter an email address AND it must already be registered on the website. Then the user is prompted to

enter a verification code sent to their account email which is also verified. If the user wants to start again, they can simply click start over. Or if they remember their password and want to try again, they can click login. User must keep in mind that the code has a timer and they must enter it quick, otherwise, they have to start over. Then the user is prompted to enter a new password and confirm it. Once done, they are redirected to the login page where they can enter their username and new password.

6. Once the user is in the main home page, they have access to viewing their account details/history, accessing admin page (if they are given the admin authentication/role), searching for posts, home page button, settings page, and logout. Whichever page the user on will be highlighted in the nav bar.
7. The home page consists of the top communities sorted by total number of members, communities the user has joined. The user can create posts by clicking the posts button and they are also showcased on the home page, along with posts made by other users. These posts have update and downvote functionality and the colour of the arrow changes to reflect this.
8. To join a community, the user must click 'join' which redirects them to a list of communities they can join. If the user tries to join a community they are already in, it simply redirects them back to the home page. If a user that is not logged in tries to join this community, they will be redirected to login. They can also decide to leave by clicking the exit icon beside the community.
9. To create a community, the user must click 'create' which redirects them to create a community page. They enter all fields: name and description in order to create a community. If the user tries to leave the page with some details in the fields, they are displayed an alert/warning that their changes may not be saved. Once a community is created, user is redirected back to the homepage where they can decide to join the community they made.
10. To create a post, the user must click 'create post' on the home page. They must enter a title, and select a community, the description and image are optional. If some details are inputted and the user decides to go back or leave the page, an alert is displayed that their changes may not be saved.
11. The user can also comment under any post. Each post is displayed on the homepage and to access comments, the user can either click the title of the post

OR the comment logo. From there, the user can enter their comment and click comment to post it. All the other comments are displayed from other users. If the user wishes to hide/show the comments they can do so by clicking the button beside the comments.

12. User can also upvote and downvote posts from the homepage or from viewing the post where the comments are displayed as well. It is updated for other users when clicked as well in 1-3 second intervals using AJAX for asynchronous updates. The score is displayed beside the upvote arrows. Comments are also votable.
13. If the user clicks on the community from the homepage (under the user's name) it redirects to the community info page. It shows everything related to that community such as the name, how many members are in there, what the community description is, and you can also join the community from this page and/or create one. The posts are also displayed related to this community.
14. The user can see their account profile as well by clicking their username on the homepage navigation bar. This page shows a history of all their posts, the date they joined, their username, their profile picture, the communities they are in (can also leave from here as well), and their past comments. If the user clicks any of the field (similar to home page posts section), they will be redirected to that page. For example, if a user clicks a comment section, they will be redirected to that post where that comment was made.
15. If the user wants to change their password/username, they simply click on the settings icon on the home page and enter the fields and click save.

Admin Walkthrough:

1. If the user is given admin authority, the 'Admin' button will appear on the navigation bar in the homepage.
2. The admin can also view graphical representations of Number of posts in each community by clicking the "chart" logo button in the navigation bar. This graph is asynchronously updated with each post made in each community.
3. If the admin hovers over the graph bars, they can see the community name and the number of posts in that community.

4. The admin can delete users, posts, and communities. The Admin can search for specific posts as well as search for specific users in the users section.
5. The admin can click the chart button by the search bar, this will redirect them to the adminGraph page where a live graph using AJAX shows up to date infographics on the number of users in each community.
6. The admin can go back to the admin page from the graph by clicking the 'admin' button in the top left.

Implementation from a Developer's Point of View:

Functionality Implemented:

- Files and their functionality
 - PHP, JavaScript, JSON, XML, CSS, Packages
 - Admin.php file is for admin features such as deleting posts, communities, and users. Queries to show posts, search, are also implemented as in the home.php file.
 - AdminGraph.php file is for showcasing the graph for the admins to view and track number of posts in each community from time to time and beware which communities are having the most activity.
 - checkLogin.php is for checking if a user is logged in using the isset functionality with user_id
 - community, communityList.php are for the communities forms. This is where data is grabbed from the database using user_id and community_id to display on the page.
 - connectDB.php is for the connection details
 - createAccount.php is for creating a new user account, all values are required except profile picture(username, email, password, confirm

password)

- createCommunity.php is for creating a community that also takes into javascript file Postvalidation.js and alert.js for validating posts and warning user if potential information loss if the page is reloaded or exited. Data is collected from the community table from the database
- createPosts.php is similar: uses mysqli_query to prevent sql injection attacks, data is collected from the database. Javascripts file included for validation and warning user
- deleteUser.php and deletePost.php is for admin use for deleting. Uses parameterized queries to prevent sql injection attacks.
- home.php page is for displaying the main home page with a navigation bar, creating/viewing posts, communities (join/create), displaying top communities. Takes in asynchronous updates for upvote/downvote functionality with the javascript file async.js
- joinCommunity.php is for joining communities. This inserts a new row into the user_community table in the db (user_id and community_id)
- leaveCommunity.php is for leaving communities. It checks if user is a member of the selected community, if user is then they are able to delete that community. This is also erased from the viewAccount.php page where it shows history of user's communities.
- Login.php is for logging in. Username is checked if it is in the database. Password is also checked, if wrong it displays "oops, wrong password". Otherwise if user does not exist it displayed "sorry, user is not registered"
 - parameterized queries for prevention of sql injection attacks
- logout.php page destroys the session and redirects to home.php page to view posts. User is able to log back in by clicking login in the navigation bar
- package-lock.json and package.json for chart.js displaying of graphs
- phpunit.xml for tests configuration. These settings make sure tests work the way they are supposed to (colors, stoponfailure, etc)
- resetPassword.php for resetting password that verifies if the email entered is correct and valid in the database. If so, user is sent a verification code to

the email and database with an expiry timer. If time expires or user remembers password, they can start over. If code is incorrect, the user is notified that it is wrong. Switch statements used for each iteration (enter email, enter code, etc). User is then given the next switch statement to occur which is to enter new password. New password is updated in the database and that can be used to login.

- settings.php page is for resetting username and password. User is updated and so is password. Parameterized queries to prevent sql injection attacks are used. Password is also verified previously before updating in the database. If verification passes (correct old password), an alert is displayed saying “password updated successfully” If verification fails (wrong password) an alert is displayed “Error: old password is incorrect”.
 - updateCommentScore.php and updateScore.php updates the score on each comment and post. It gets the post id and vote from the request. Upvote and downvotes are updated, and new score is returned as JSON.
 - viewAccount.php page is for viewing the user’s account history. Previous comments, posts, when they joined, etc. This is done through joining sql queries to show and display the history (posts, communities, comments...)
 - viewPost.php is for viewing posts with multiple select queries that are parameterized to prevent sql injection attacks.
 - tests for phpunit are made for the login.php page to check if login page loads correctly (passes)
 - login.css, resetPassword.css, settings.css, style.css, viewAcc.css for styling of pages. Universal style page is style.css.
 - images folder for storing images/profile pictures etc.
- User Login/Sign-up page
 - The user is prompted to create an account filling out the following credentials: Username, email, password, profile picture and confirm-password.
 - When the user goes on the website, they are able to view the posts, communities, and etc but are not able to create a post, comment or join a community. They must have an account.

- Once a user account is created, their information is stored into the database and then they are able to login. Remember, information/credentials must be correct and valid or error messages will be displayed such as “Incorrect password, email, etc” when a user tries to login.
- Logout page
 - User can logout of their account and are prompted to the login page where they can log in again OR go to home page to view the content but not interact with anything.
- Database Functionality
 - Database is complete with the necessary tables included for retrieving and storing user website data. Discussiondatabase.sql file included in the repo.
- Home Page
 - Logged in user or without registering is able to filter searches by searching for a post, user, or community keyword. Logged in User is also able to create a community and/or join one that is already made by other users. The top communities are listed and ranked by the amount of users joined. The downvotes can also go in the negatives if the comment/post is not liked by others. Asynchronously updated upvotes/downvotes using javascript.
 - Logged in User is also able to create a post by entering a title, which community it relates to, the description and possibly an image (future iteration).
 - Logged in User can decide to leave a comment on a post as well which stacks along with the other user’s comments.
- View Account/User page
 - This is where the user can see the communities they are in currently, when they joined the website (made an account), and view their previous posts. They will also see their profile picture.
- Alerts
 - If user does leaves some information (community creation or Post creation) in the given fields and tries to exit the page, they are given an alert warning them that their changes may be lost. Implemented through javascript through the ‘beforeunload()’ function.

- Settings
 - User is able to change their password. User has to enter old password, new password, and then confirm password. If the password does not match previous password, an error shows up and asks to retry as it does not match the database stored values. User must fill out all the fields before saving.
- Email forgot password
 - Forgot password implementation sends user code and expiry time to the database. Code also sent to the user email. User can start over if the wrong email is entered or if the email is not in the database or if the code timer has expired which the user is notified if it is.
- Create/Join Community and Leave
 - User is able to create a community (community name, description, and create button).
 - User is able to join previous communities made by other users. The community is then displayed on the home page and the user page if they view their account. If a user is not logged in, the community sidebar will tell them to login to join communities
 - User is able to leave the community; updated in the database
- Admin Authentication/functionality
 - 'isAdmin' column in 'users' table in the db. Type is boolean and default value is 0 for not admin, manually changed in database to 1 if given admin authentication.
 - Admin is able to search for user by email, name, or post. Can search for posts, communities and users in general posts column, as well as can search for specific users in their user search sidebar. Admin can press garbage icon to delete a post or user
 - Admin is able to delete a user, this will delete all posts by that user, comments, and their communities joined. If the user has not posted, commented, or joined a community their record will be permanently erased. If the user has posted, their actual account will remain in the database. This is to allow for reactivation

in the future, however this can be changed to completely remove the users record regardless of previous post/comment history.

- Used chart.js to implement graph functionality with AJAX for asynchronous updates on the communities and number of posts in those communities.
- Asynchronous updates
 - Use .AJAX, JavaScript, and JQuery, along with JSON (exchanging data from client and server) to enable the browser to send requests to the server in the background without requiring a full page refresh on upvotes and downvotes (ongoing)
 - When the new data is retrieved from the server database, the JavaScript code updates the relevant parts of the page for a seamless and efficient user experience
 - Real time comment updates using AJAX requests, upvoting and downvoting vote count update in real time, user notifications.
- Image upload
 - Profile picture is uploadable and user can upload pictures to their posts as well
- Server side security and Client Side
 - Required parameters (password, email, user etc)
 - Secure password storage
 - Hashed passwords
 - Parameterized queries
 - Instead of creating a template with placeholders, the parameter values are passed directly to the query as arguments. The database engine then handles the parameterization of the query.
 - Used in admin page for deletion
 - SQL injection attack prevention
 - Input validation through password (specific length, requirements)
 - Used mysqli_query and mysqli_connect to prevent sql injection attacks

- Database privileges
 - Limit privileges to only admins that are manually given the authentication. Prevents hackers from modifying sensitive data.
 - Use of Prepared statements
 - This approach can prevent SQL injection attacks because the parameter values are treated as data and are not executed as code.
-
- User should be able to reply to comments
 - More graphical representations
 - Profile picture should be able to be changed from the viewAccount.php page
 - Settings accessible even if account is not created

Database

Store the user information upon account creation

- Name, login details (hashed password, email, user id)
- Posts
- Date and Time
- Communities
- Upvote/downvote score

Store users and validate upon login

- Once user logs in and is validated, they can view their account
- Posts and comments from this user will be added to their account

Store posts

- Posts include text content, user, community, image(optional), upvotes, and comments

- Retrieved on view account page

Store communities

- communities have to store their corresponding posts
- Store their username/id that are registered to that community
- Retrieved on view account page

Store comments

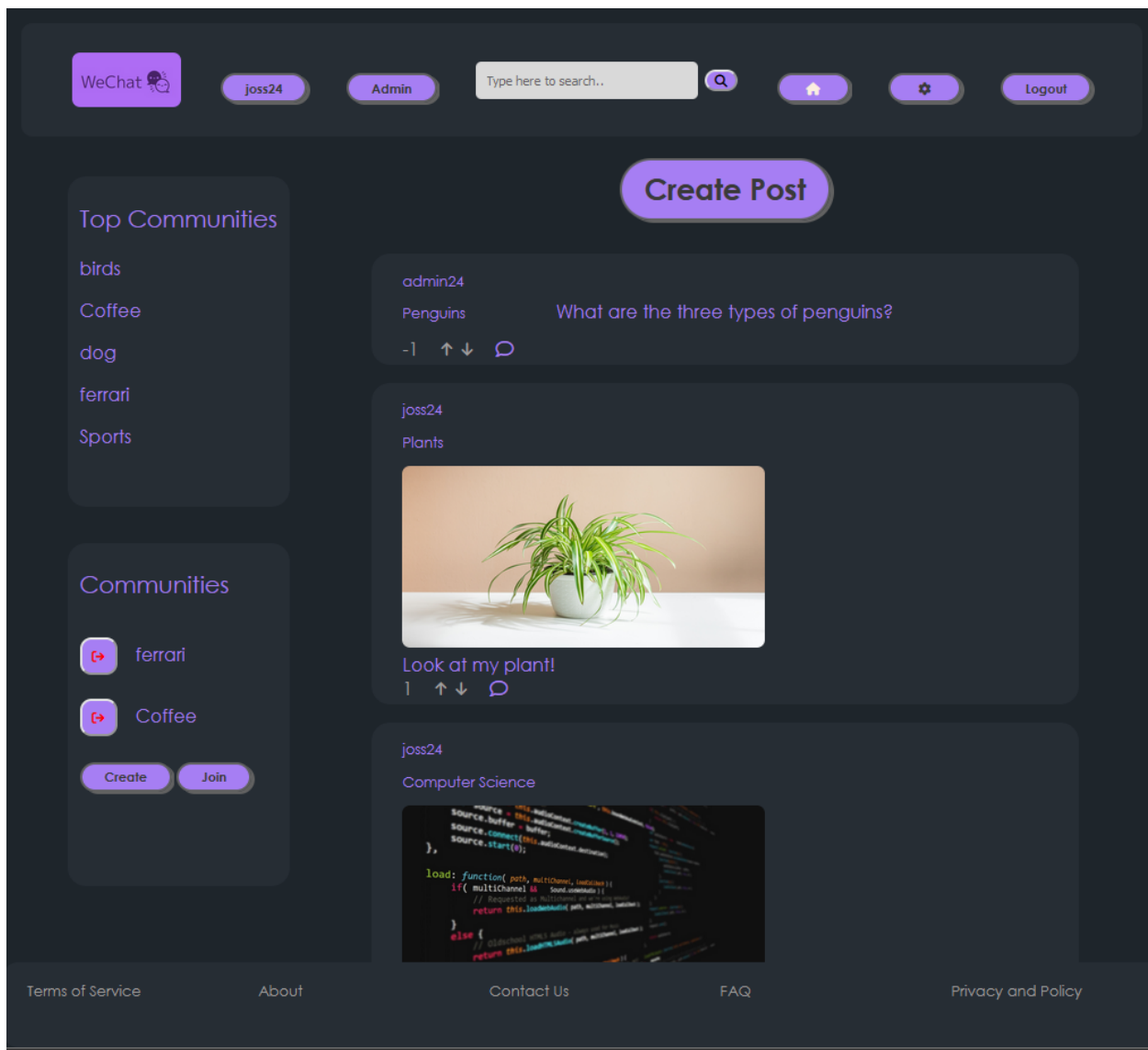
- comments stored in database

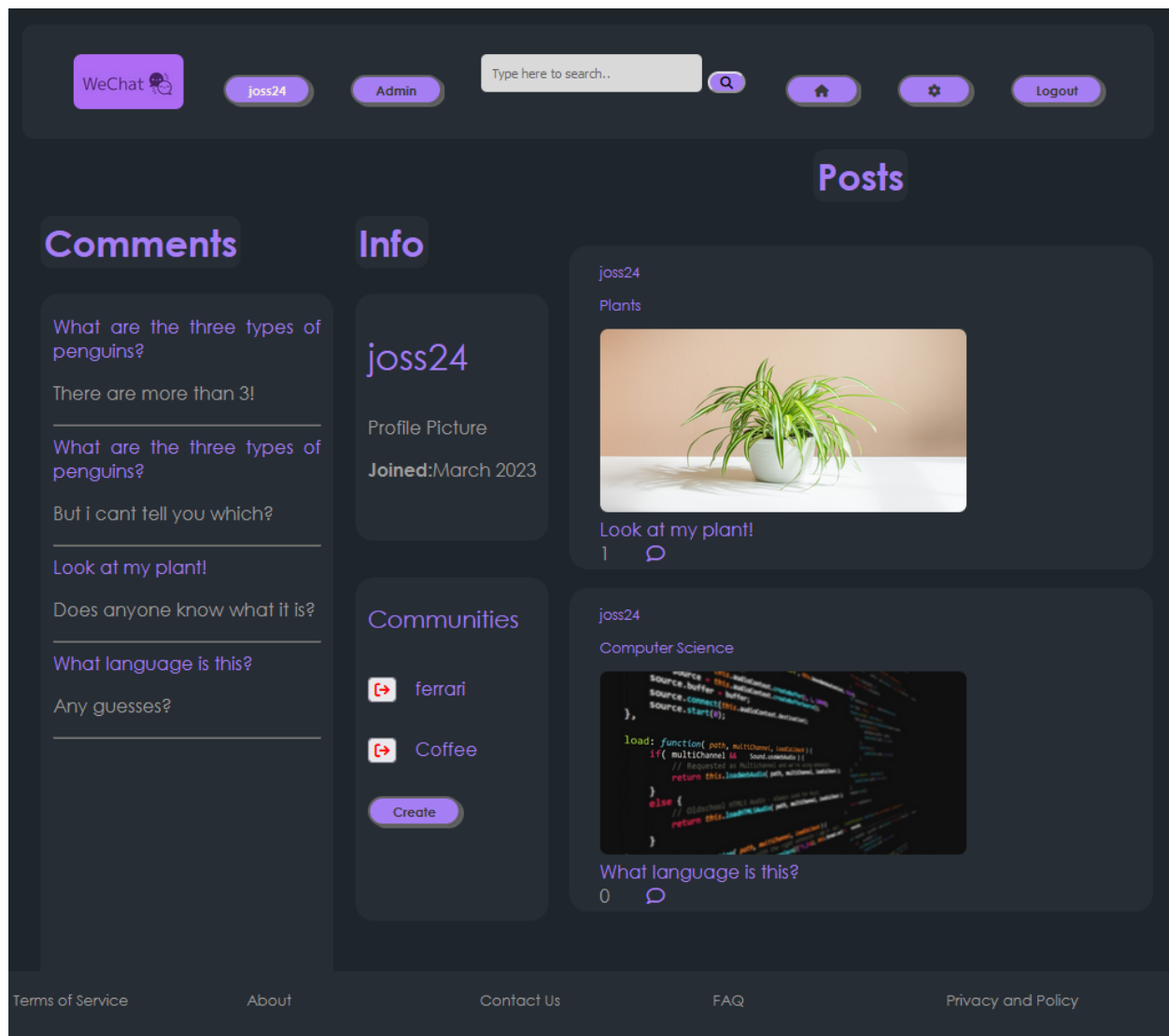
Store forgot password (codes)

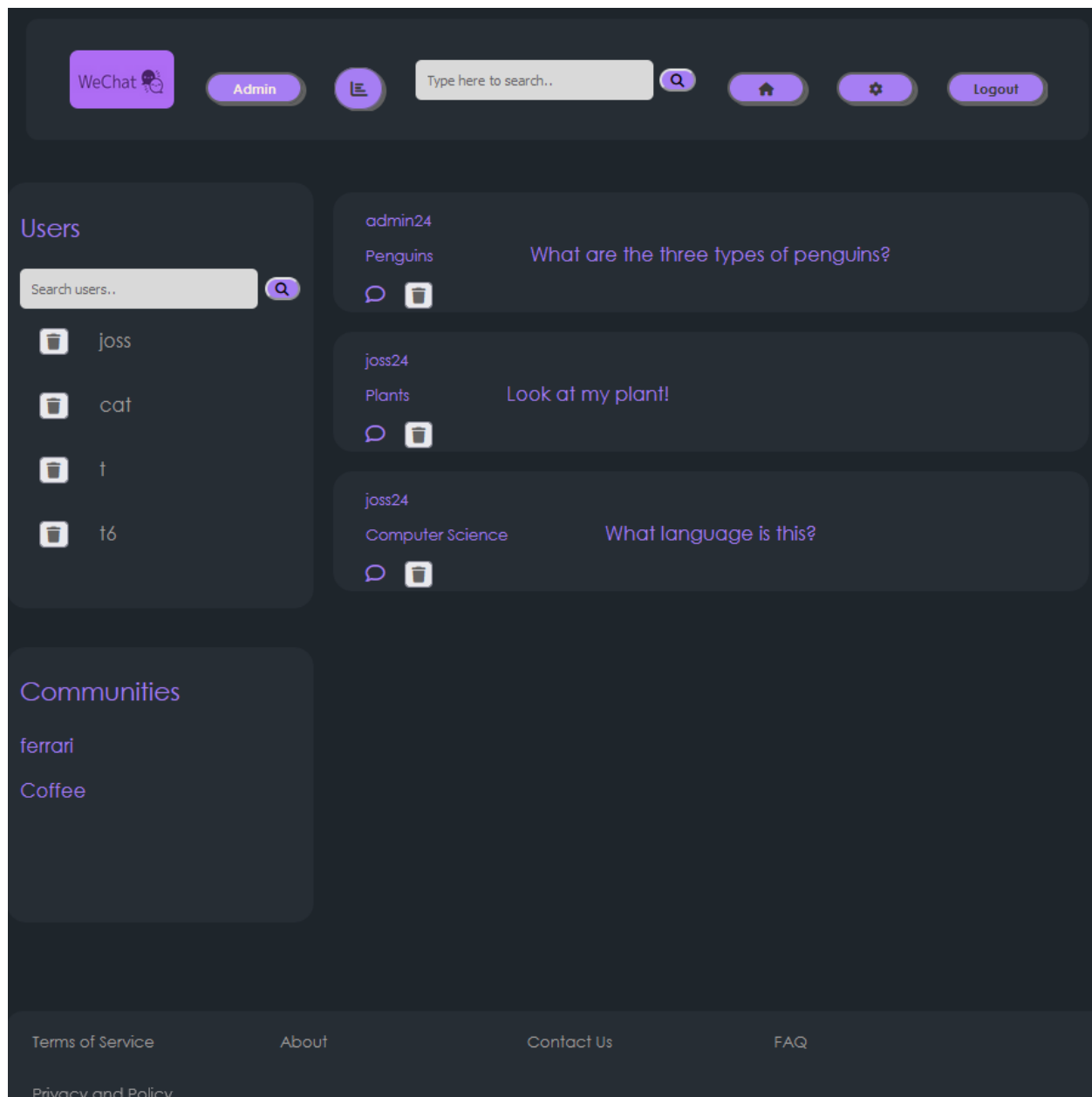
- Store expiry time and verification code and email in users table for forgot password verification details

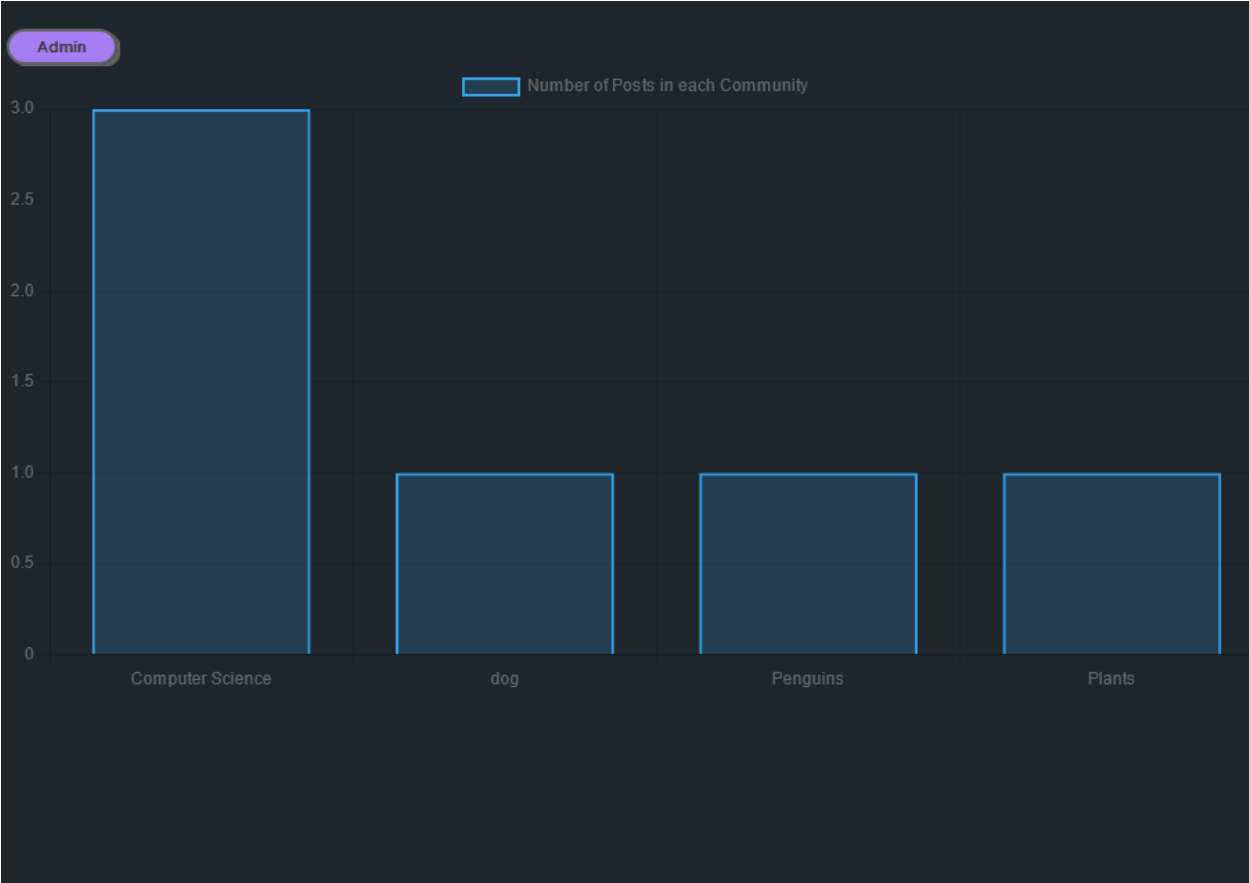
Store Admin → 0 for not admin, 1 → admin

- manually alter in database 0 is default if user not admin, 1 if user admin, thus given admin privileges.










HomeLogout

User Settings

WeChat

New Username:

New Username

Old Password:

Old Password

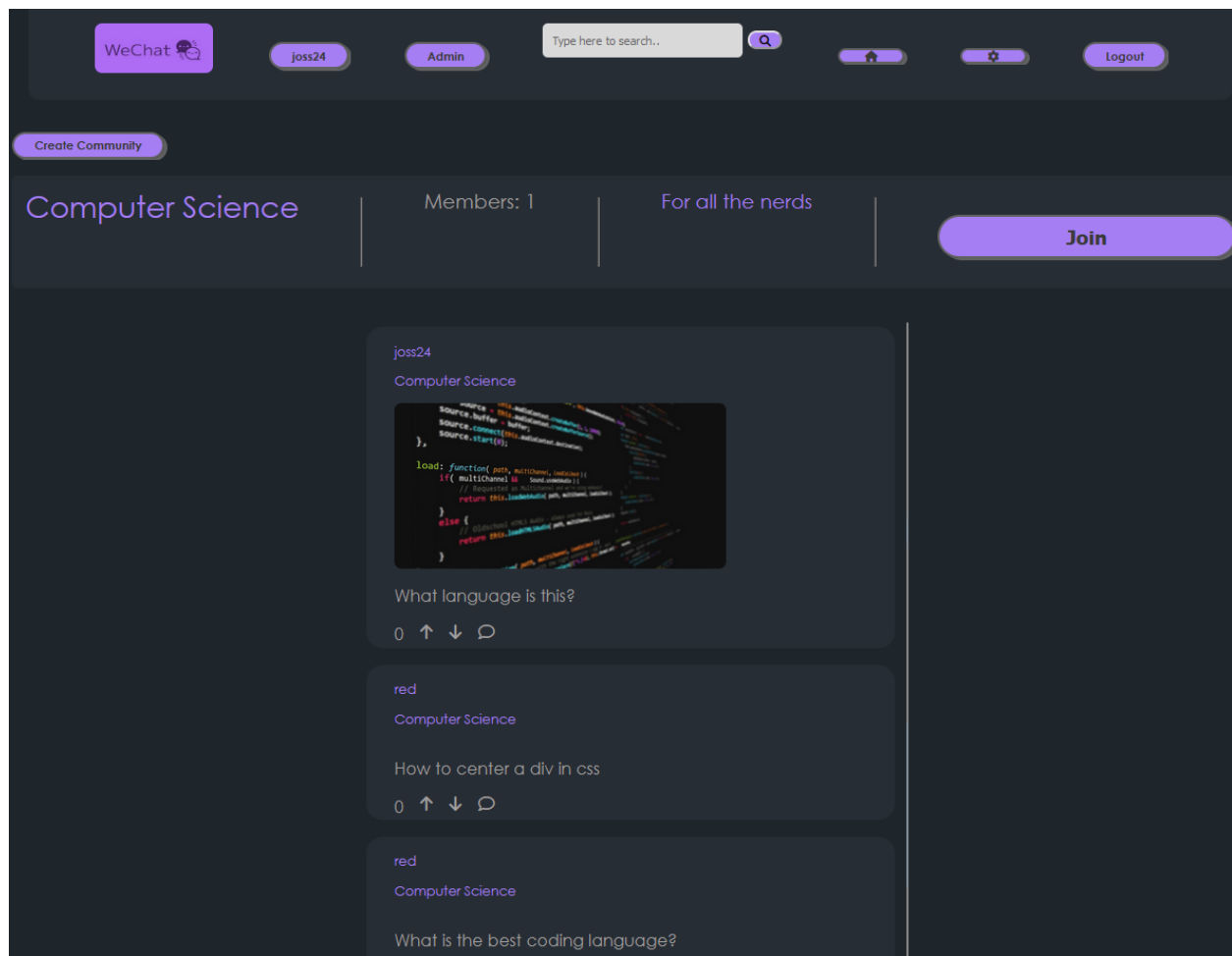
New Password:

New Password

Re-enter New Password:

Re-enter New Password

Save



joss24LoginType here to search..Logout

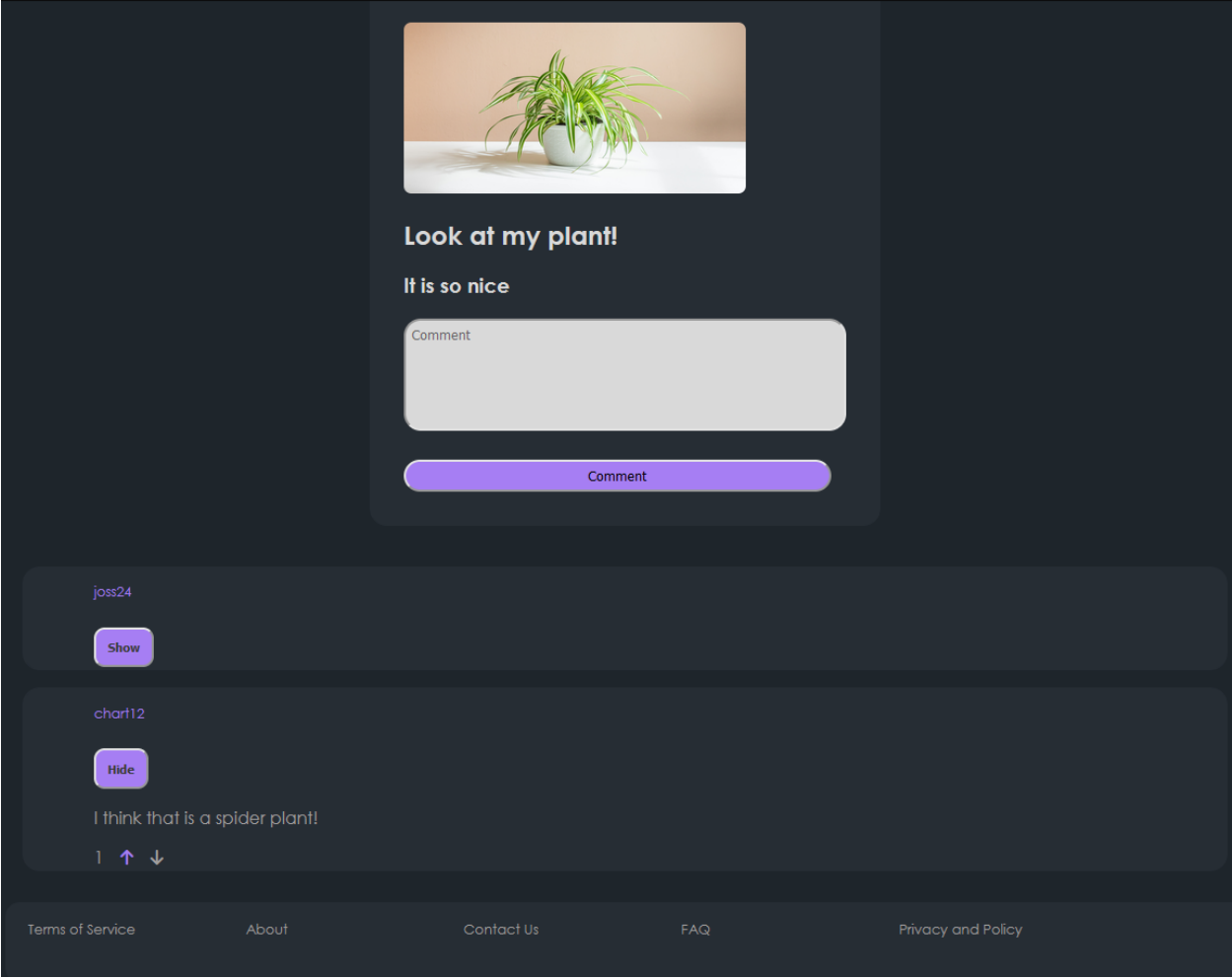
Create Community

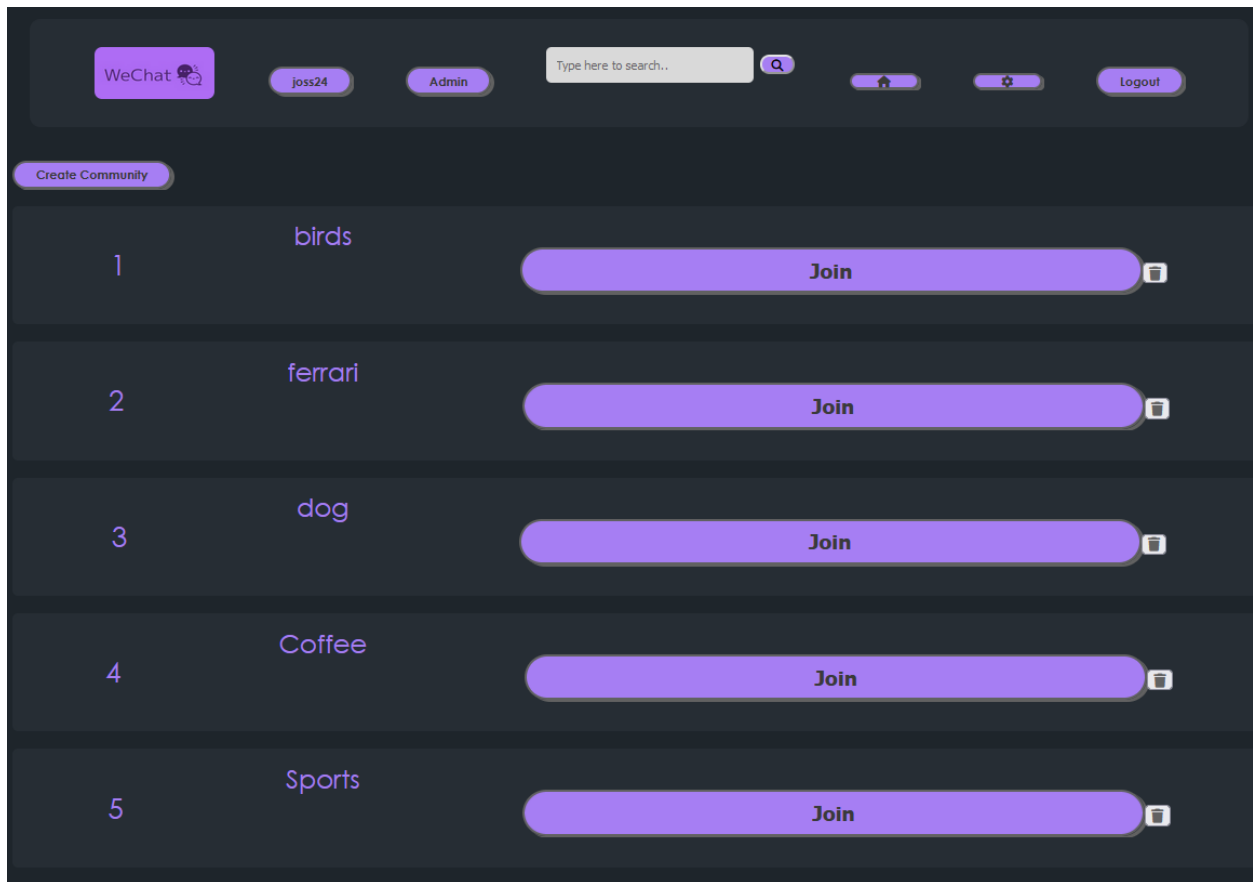
Community Name

Description

Create


Terms of ServiceAboutContact UsFAQPrivacy and Policy






Updates and improvements from minimal core functionality:


Display in graph for Admin: Number of posts in each community, number of users - Aryan


WeChat


joss24

Admin

Type here to search..








Logout

Create Post

Title

Choose Community 

Description (optional)

Browse...

No file selected.

Post

Terms of Service

About

Contact Us

FAQ

Privacy and Policy