CS4530 Final Project: “StackViolet”

Group 607: Grace Armstrong, Khushi Khan, Saanvi Vutukur, Jessica Zhao

User Stories and Features

Community Engagement, Accessibility, Productivity

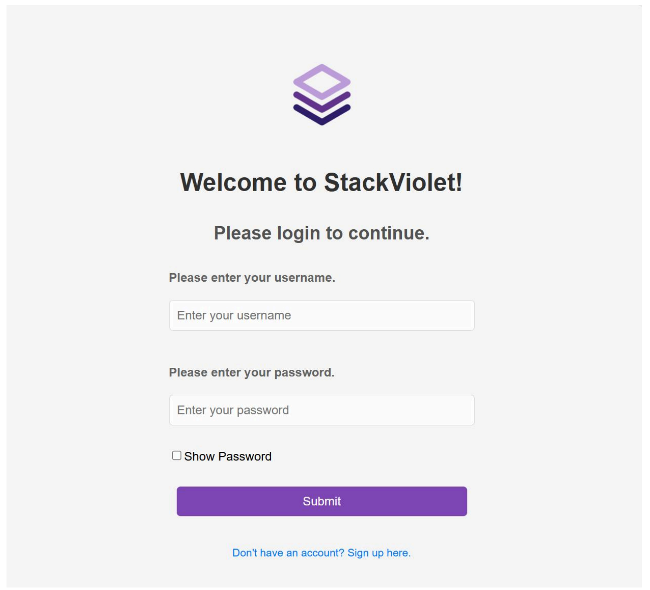
Many Q&A platforms struggle with content discovery and user engagement. Without a clear system to track topics of interest, users often miss discussions or waste time searching. Additionally, accessibility features are often lacking for users with visual impairments, and user contributions are rarely recognized, reducing motivation.

Our platform addresses these challenges with three key focuses: engagement, accessibility, and productivity. We display user activity through a contribution visual and offer community features with topic-based public and private groups handled by moderators. Nim game offers entertainment while waiting for responses, with daily streaks and leaderboards to boost regular usage. We also enhanced accessibility with customizable color schemes, adjustable text sizes, dark/light mode, keyboard shortcuts, and markdown formatting, all saved to user profiles.

For productivity, we provide a centralized homepage for quick access to notifications, starred posts, upvoted/downvoted content, and filtering options. A notification system keeps users updated. These features create an accessible, organized, and community-based platform that encourages continued participation.

Demo and Source

Our demo site is available at https://cs4530-s25-607.onrender.com/ and our code at https://github.com/neu-cs4530/spring25-team-project-spring25-project-group-607/tree/profile-settings.



Welcome to StackViolet!

Please login to continue.

Please enter your username.

Enter your username

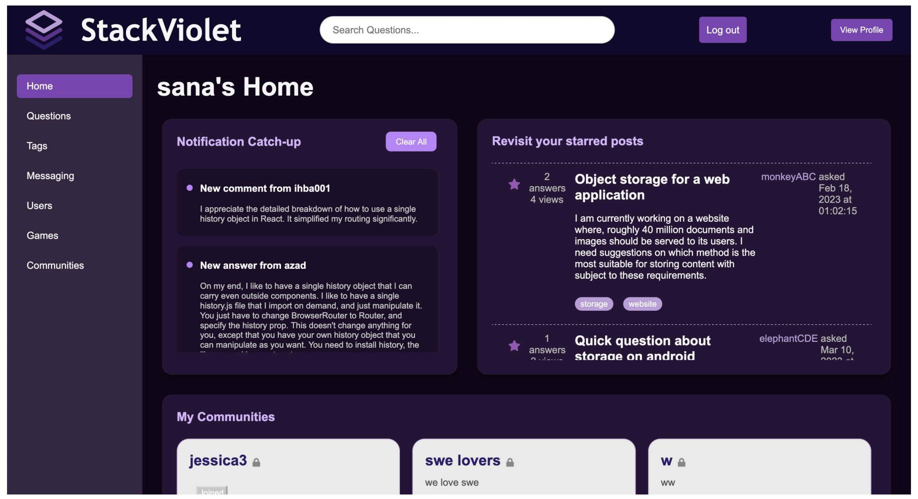
Please enter your password.

Enter your password

Show Password

Submit

Don't have an account? Sign up here.



sana's Home

Notification Catch-up — Clear All

New comment from ihba001

I appreciate the detailed breakdown of how to use a single history object in React. It simplified my routing significantly.

New answer from azad

On my end, I like to have a single history object that I can carry even outside components. I like to have a single history.js file that I import on demand, and just manipulate it. You just have to change BrowserRouter to Router, and specify the history prop. This doesn't change anything for you, except that you have your own history object that you can manipulate as you want. You need to install history, the

Revisit your starred posts

2 answers 4 views — Object storage for a web application — monkeyABC asked Feb 18, 2023 at 01:02:15

I am currently working on a website where, roughly 40 million documents and images should be served to its users. I need suggestions on which method is the most suitable for storing content with subject to these requirements.

storage website

1 answers — Quick question about storage on android — elephantCDE asked Mar 10,

My Communities

jessica3

Joined

swe lovers

we love swe

w

ww

Technology Stack & Design

MongoDB, Express.js, React, Node.js, TypeScript

Our features were implemented on the existing FakeStackOverflow codebase. Many of the tracking features, such as the contribution visualization, game streaks, and leaderboard were implemented through a combination of back-end work, front-end work, and sockets. Many of our accessibility features were created using libraries, such as react-markdown, react-katex, and react-math. For features such as starred posts and communities, we followed the same back-end design of defining mongoose schemas, user models, relevant types, and implementing services and controllers. In our front-end, we implemented services and used them in the necessary hooks for each component. This helped in creating the centralized home page, as it required reusing components that were created for other features. By using Socket.io in both the back-end and front-end, we were able to implement real-time communications within web clients. Notifications were compiled using aggregation operations and displayed on the front-end.

The site is deployed on Render, and continuous integration is done through Github.

Future Implementations

In the future, we plan to look towards using component libraries to quickly enhance our front-end, as it was a time-consuming part of the project. In addition, we struggled with integrating our code towards the end, so we plan to delegate tasks more effectively to account for that.

As we were working on the Communities feature, we realized that there were far more functionalities that we hadn’t planned on doing which made sense to add to our project plan. Due to the scope of the other tasks we had planned out, we were not able to implement extra functionalities, such as leaving the community, removing members, demoting moderators, and updating community information.

In addition, we considered the idea of using recommendation algorithms to suggest content tailored to user interests and past interactions. This would enhance the content seen on the home page, and could help with productivity. Finally, community moderators could be replaced with AI tools that ensure discussions are safe, detect authenticity of posts, and accept users who match a certain criteria, to emphasize respectful communities.