New User Signup

Acceptance

Who is this for?

New users that want to sign up for our algorithm speed test app.

What problem are we solving?

We want to make the user sign up flow as quick and easy as possible.

How do we measure success?

- Increase in new user signups
- Increased conversion through sign up flow

Details

Users need to ability to create an account for our app before they submitting their algorithms. From a front end, customer facing perspective, this is the New User Sign Up feature. From an infrastructure perspective, we're creating authenticated db entries that allow our users to interact fully with our application.

Signing Up

New Users Sign Up will be the default homepage for the application. Users will be shown the input field of Username and Password, a Sign Up User button, and a link below the form to log in if they previously created an account.

Elements of a New User Sign Up:

- Username
- Password
- Sign Up Button
- Link to login for returning users.

Algorithm Input Page

Acceptance

Who is this for?

Users that want to run speed tests on their algorithm.

What problem are we solving?

Bragging rights for the cohort. Code Golf + algorithm efficiency with some arbitrary scoring system for the two.

How do we measure success?

- Number of algorithms submitted
- Number of algorithms submitted per language
- Repeated submissions for the same algorithm by the same user.

Details

Users need a way to understand what the app does in a quick and succinct way. Let's have the web code editor in the middle of the screen with some instructions above. "Welcome to Algorithm Golf. Below is a problem we'd like you to solve. "We will award points based on algorithm efficiency in ms and its character length, with less characters receiving more points."

On the technical level we'll need a way to store coding problems and randomize the output for which one gets displayed. We'll also have to parse the inputted code and display the score based on a few different criteria:

- Parsed code should be categorized into a certain language (JS, Python, Ruby, etc..).
- Character count for the code should be recorded and saved to the unique inputted code's id. Score will be based on a static scale, no time to figure out how to make it relative.
- Above stated data should be passed into a congratulations window once the code is run.

Instructions

On the top of the screen will be instructions of how the app works. See above for sample copy. Below the instructions will be the coding challenge to solve and the web code editor for the user to input their algorithm. Using a dropdown(I think this is what CodingMirror does) the user can choose the language they want to submit their algorithm.

Algorithm Input

Coding editor… but on the web. Input code, doesn't run if code is broken, will run it if it works. We'll need to figure out a way to parse the code inputted for a few different reasons. We need to run it through a performance library and speed test it (if it's not a feature build into CodingMirror already) and parse the code to get a character count. These speed test results and character count will need to be saved for the inputted code's unique ID and be made available for the congrats modal.

Congrats Modal

After the user submits their algorithm successfully, a modal should pop up that congratulates the user on a successful submission. The screen should include their speed test results in ms(?) , the total character count of the submission, and the total score for their submission.

There will be links on the modal that directs the user to the leaderboard or to another coding challenge.

Elements of Algorithm Input Page:

- Leaderboard link(top left)
- Profile LInk(top right)
- Instructions
- Coding challenge
- Algorithm input page.

Leaderboard

Acceptance

Who is this for?

Users that want to check out the leaderboard.

What problem are we solving?

Making it easy to view leaderboards for each different language. Also a way to quickly check the user’s personal standing.

How do we measure success?

- Visits to leaderboard page
- # of clicks on “My Rank” button.

Details

Users need a way to check the rankings and their standings. The Leaderboard initially gives the overall rankings, with the different points in various languages tallied up for each unique user. The leaderboard can be broken down further by language, to see how each user fares in individual languages.

Select Language Dropdown
Lists available languages on the platform. When user selects the language from the dropdown the leaderboard should refresh to the rankings for that particular language (yay SPA).

My Rank Button
On both the overall and individual language leaderboards there should be a my rank button that either highlights or scrolls down the page to the unique user’s standing in the leaderboard.

Elements of a New User Sign Up:

- Language selector
- Leaderboard(defaulted to all scores)

- My Rank button
- Accounts will not be viewable in MVP.

Log In Page

Acceptance

Who is this for?

Returning users

What problem are we solving?

Simple login page.

How do we measure success?

- Retention

Details

Replace sign up w/ log in for the Sign Up User Story.

Elements of a New User Sign Up:

- Username
- Password
- Log In Button
- Link to Sign Up for new users.

Profile Page

Acceptance

Who is this for?

Checking personal scores

What problem are we solving?

Too lazy to look at leaderboards

How do we measure success?

- Doesn't matter

Details

When a user clicks the my profile button, there will be a list displayed with the languages available on the platform along with the user's score in said language.

Elements of a New User Sign Up:

- Username
- Languages
- Scores in those languages

Scoring Algorithm

Max points @ 250ms. === 100 points

Points decrease by 5 points for every additional 20 ms.

Score of 1 @ 600ms.

Multiplier is started @ 10x. After 20 characters, decreases by .1 every 2 characters.