LeeterBoard

Full-Stack University Contest Leaderboard

Team Name: Runtime Terror | GitHub Team: LeeterBoard

Members: Cole Price, Dhruv Patel, Caleb Damron, Brody Curry

Introduction

Overview

LeeterBoard is a full-stack web application that gathers LeetCode contest data and categorizes participants based on their universities. The platform allows users to filter rankings by institution, track their performance relative to peers, and compare standings across universities.

Our motivation stems from the idea that seeing rankings at a university level could enhance the competitive spirit of coding contests. By providing visibility on how students perform within and across institutions, we aim to promote a sense of motivation and engagement in competitive programming.

LeeterBoard is a novel idea, as we are not aware of any existing applications that offer this type of university-based ranking system. It fills a gap by introducing an extra layer of motivation and competition for LeetCode contest participants.

Early on, our scope was much smaller, with the overall concept being purely a ranking system, without much else. A site that would simply show a list that had username, rank, and points, with the ability to search for universities to see specific rankings within schools. The project started very smoothly, enough so that more ideas were added. We chose to add a page for a graph that correlates user scores and data in colleges to compare and contrast scores within universities. Furthermore, additions of custom badges and tiers for high ranking competitors and functionality across several devices.

All goals were met, with the expected challenges that come with the creation of a site and data storage.

Customer Value

(Changes from Proposal: No Changes).

Customer Need

Our primary customers are university students who participate in LeetCode contests and are looking for a structured way to compare their rankings against peers within their institution and across other universities. Currently, LeetCode does not provide a built-in feature for university-specific rankings or peer comparisons, leaving students without an efficient way to benchmark their performance within their academic community.

Proposed Solution

LeeterBoard addresses this gap by providing a comprehensive leaderboard system that categorizes contest participants by university. This feature enables students to assess their performance relative to their peers, fostering a competitive environment that encourages continuous improvement.

Measures of Success

The success of LeeterBoard will be evaluated based on key metrics, including:

- User Adoption: The number of students actively using the platform to track rankings.
- User Friendly: Users are able to navigate through the website from multiple different devices, having seamless experiences.
- Impact on Participation: Increased involvement in LeetCode contests as students strive to improve their rankings.
- User Feedback: Positive responses from students who find the leaderboard beneficial for motivation and performance tracking.

Technology

Architecture

Our software pulls contest data from the LeetCode Leaderboard and filters the data by University. The main components of our system include:

- The data scraper (backend) fetches the contest data, extracting relevant participant information.
- The database stores and organize the contest results, user profiles, and university mappings.
- UI/Display (frontend) displays the leaderboard, search filters, user profile, and rankings in an interactive UI.

The LeeterBoard system has a Flask (Python) frontend, a Python backend, a MongoDB database, and a Python data scraper. The frontend displays rankings and filters data by university, sending requests to the backend, which processes queries and retrieves contest results from the database. The database stores user rankings, contest results, and university mappings. A Python-based data scraper periodically fetches LeetCode contest data by hitting an API workaround, ensuring up-to-date rankings. This setup allows students to track and compare their competitive programming performance efficiently. The line graphs use chart.js, an HTML chart that uses a map and aligns the university information.

Furthermore, several background scripts help optimize the loading of universities, the weekly rankings, and scripts for any new users to be added to the database and their data to be quickly calculated and placed accordingly. Rendering is server-side with Flask and Jinja2 templates.

Team

Roles

Two members, Cole Price and Dhruv Patel, largely led the project, providing the bulk of much of the code and testing. Both members worked heavily on the back and front end, and had several quality of life additions to the site and its design and utility.

Brody Curry largely supported Design and UI concepts during sprint 1, as well as writing documentation on the project.

Caleb Damron supported testing and site/CSS development, as well as overall Python backend work.

Project Management

Schedule

The project largely went smoothly, with everyone working quickly to get the site up and running and get much of the initial sprint and setup fixed. All members worked on different UI elements to ensure a complete and proper design that would be pleasing to users, as well as new implementations of quality of life features to help make the site accessible and easy to use and understand.

All aspects of the project has been completed, with simple enhancements to existing systems to make things smoother, more satisfying to both use and to observe, and to make it as professional as possible, with clickable elements having a slight effect on hover, as well as cleanup for the users per the university page.

Reflection

Overall, the project was a success on all fronts. Planning and discussion on overall goals went smoothly as previously discussed, and the development and testing went smoothly as well. While there were issues with getting certain members set up to work on development in time to support the project during early development, the project itself went extremely smoothly, mostly. As some of the more complex elements of the UI and especially the line graphs, using tools like react would have been extremely helpful. Team management also went well, with heavy discussion on improvements and additions to the site and logic happening weekly, as well as a healthy environment for support with issues and errors. We believe the project was a success, with all site elements running smoothly and on time, as well as looking extremely professional.