**Codewars Private Leaderboard**

The purpose of this project is to create a small SPA to display a private Codewars leaderboard.

The leaderboard will be empty to begin with and populated with Codewars users added by our

SPA users. Please take a look at the requirements below.

Requirements

These requirements should be seen as a lower bound to the final product, which means you’re

being given creative space to make your SPA more interesting if you have the time.

Functional

● Account

○ Register

○ Login

○ Logout

○ Display logged in user name

● Display leaderboard (2 views)

○ General leaderboard view based on honor (username | name | honor)

○ Rank leaderboard view based on score (username | name | score | “overall” OR

<language>) - default “overall” + language filter

○ View pagination

● Add Codewars user to leaderboard

● Remove Codewars user from leaderboard

● Display all available Codewars user details on leaderboard item selection

● Auto refresh (up to 5 min delay in rendering leaderboard changes when honor or rank

change)

● Add comments to Codewars user (to be displayed when showing user details)

Tools

● Jira

● GitHub

● GitHub Actions (at a minimum, CI should be in place, i.e., build + test; CD to a cloud

platform of your choice will give you some bonus points)

Agile Process

● Scrum methodology

● 1 week Sprint

● Day 1

○ Sprint Planning should happen after the initial Project Setup session (includes

user story writing)

○ Teams should elect Scrum Master

● Stand-ups should be held at the beginning of daily “Project Work” sessions from day 2

● Review/Demo session will happen on last day of Sprint

● Since this is a single Sprint project, Retrospective will be skipped

Workflow

● Jira

○ TO DO: ticket has just been created

○ IN PROGRESS: work on ticket has started

○ BLOCKED: work on ticket can’t proceed

○ IN REVIEW: ticket is ready to review (if it’s a code view, a PR must have been

created at this stage and should be reviewed by team members who haven’t

worked on that ticket)

○ IN QA: UAT by team members who haven’t worked on the ticket (having

acceptance criteria for each ticket will help drive QA efforts)

○ DONE: work on ticket has been completed

● Git branching model

○ Our preferred model is still Gitflow, while being mindful that feature branches

shouldn’t be long-lived and there will be frequent merges to the develop branch

(CI branch)

○ Feature branch names should be prefixed with the Jira ticket ID (it makes future

Jira - Git hosting platform integrations easier) -> feature/<Jira ticket

ID>-shortname

Tech Stack

● You’re free to choose a stack (for frontend and backend applications)

● API documentation should be made available via Swagger

Testing

You should be able to demonstrate that your code coverage is > 50% (this is very low!) by

averaging the frontend and backend code coverage.

References

● Jira

● GitHub

● GitHub Actions

● Gitflow

● Codewars API

● Swagger