Problem Statement

Iris is Pluralsight's learning intelligence platform, an innovative and unique user experience, whose aim is to use data to create a smarter, personalized learning journey.
As a Machine Learning Engineer, you will be responsible for building the infrastructure and implementing the algorithms that make Iris smart. In this role, it is common to encounter tasks such as handling, parsing, transforming, and manipulating data and then constructing endpoints to access a feature vector, similarity metrics, or recommendations.

Instructions and Questions

In this challenge, we present you with a sample of anonymized Pluralsight user data that includes several variables representing our users selected interests as well as how they interact with our courses and our assessments. We have provided you with three csv files (their contents are outlined below).

We would like you to parse the data and calculate a score that represents similarity between users. You may use your judgement as to what makes users "similar" and choose what calculations you deem appropriate to identify similar users. The result should be accessible via a structured, RESTful API. The endpoint should take as an input a user handle and return a summary of the most similar users. Feel free to build the API in the language of your choice. A back-end data store such as SQLite or PostgreSQL is strongly encouraged.

The end result should be runnable and demo-able, and your source code should be available on github where you have committed throughout the process so that we can see your progress as you code the solution. You may either host your application on the internet, or you can send us instructions for running it locally.

In addition to providing your code, please give a written response to the following questions:

1. Tell us about your similarity calculation and why you chose it.
2. We have provided you with a relatively small sample of users. At true scale, the number of users, and their associated behavior, would be much larger. What considerations would you make to accommodate that?
3. Given the context for which you might assume an API like this would be used, is there anything anything else you would think about? (e.g. other data you would like to collect)

Please feel free to take as much time as you like to complete this challenge, but we do not expect candidates to spend more than 6-8 hours on this exercise.

Following submission and pending a successful review, you will be asked to present your solution either in person or via video call, and potentially do some pairing with us to add a new feature to your application.

Data Summary

This document is accompanied by four csv files with the following tables:

user_assessment_scores
user_handle: a unique identifier for each user, consistent across all of our databases
user_assessment_date: the date the user completed this assessment
assessment_tag: the assessment tag (may or may not exactly match the user_interest or course_view tags)
user_assessment_score: the user's score, from a distribution that is roughly normal with a mean of ~140 and a standard deviation of ~60

user_course_views
user_handle: a unique identifier for each user, consistent across all of our databases
view_date: the date the user viewed the course
course_name: self-explanatory
author_handle: a unique identifier for each course's author (or professor or teacher if you prefer)
level: the difficulty level assigned to this course
course_view_time_seconds: the number of seconds this user spent watching this course on this day

course_tags
course_id: course identifier for joining to user_course_views
course_tags: author-applied tags to define what topics this course covers (may or may not exactly match assessment tags)

user_interests
user_handle: a unique identifier for each user, consistent across all of our databases
interest_tag: tags this user has indicated they are interested in (may or may not exactly match assessment_tags)

date_followed: the date this user followed/expressed interest in the tag mentioned above.