View on GitHub

be-challenge

Backend Assignment

Problem Statement:

You need to build a Rest API where `Instructors` and `Students` can manage their `Courses`. An `Instructor` can manage `Subjects`, `Courses`, `Tags`, `Lessons`, and `Videos`. Below are the relationships between various entities.

- An `Instructor` is a `User` with a flag `is_instructor` = `True`.
- A `Student` is a `User` with `is_instructor` = `False`.
- A `Course` can belong to multiple `Subjects`.
- A `Lesson` can belong to multiple `Courses`.
- A `Video` can belong multiple `Lessons`.
- A `Video` can have multiple attached `Tags`.
- A `Video` should have a `link` field to save `YouTube` or `Vimeo` url.

User Stories

Required

Must implement the below CRUD application with needed support filters.

Instructor

1. As logged in instructor, I can create, view, edit, delete `Tags`.
2. As logged in instructor, I can create, view, edit, delete `Subjects`.
3. As logged in instructor, I can create, view, edit, delete `Courses`.
4. As logged in instructor, I can create, view, edit, delete `Lesson`.
5. As logged in instructor, I can create, view, edit, delete `Videos`.
6. As logged in instructor, I can view analytics - most viewed `Courses` and `Videos`.

Student

1. As a logged in student, I can view all active `Courses`. I can filter `Courses` by `Subject`.
2. As a logged in student, I can view all active `Lessons` for a selected `Course`.
3. As a logged in student, I can view all active `Videos` for a selected `Lesson`. `Videos` Can be filtered by Video title and Tag names.
4. As a logged in student, I can view the selected `Video` details.
5. As a logged in student, I can subscribe, unsubsribe to various `Courses`.

Optional

- `Video` details should have personalized Course recommendations.

Evaluation

The assignment is designed to check your coding and problem-solving skills. Please send us a link to your git repository.

What we evaluate in the code?:

- Domain driven design (usage of DDD concepts: entities, value objects, services, repositories, etc)
- Messaging (Commands and Events)
- Code organization (modularity, dependencies between modules, etc)
- Exception handling and logging
- Writing and organizing tests

What we expect to see in the README?:

- Architectural overview (knowledge of distributed services, cloud platforms)
- What tools and technologies you used (libraries, framework, tools, storage types, cloud platform features)
- What you think that it can be improved and how

be-challenge maintained by proximity-tech

Published with GitHub Pages