تسک BACKEND

جواب سوالای زیر رو با دقت جواب بده وهر سوال داشتی ایمیل بزن بپرس و حتما **5 روز** دیگه حتما بفرست.

سلام دوست عزیز

قبل از توضیح پروژه این نکات را در نظر بگیر.

در اجرای این پروژه می‌توان از هر تکنولوژی و زبانی استفاده کرد، ولی پیشنهاد ما استفاده از زبان سی‌شارپ و فریم‌ورک دات‌نت کور است.

در صورت نیاز به نمونه id کتاب می‌توان، از روی سایت طاقچه id کتاب‌ها را برداری.

توضیح مساله:

یک سرویس طراحی شود که اطلاعات مربوط به کتاب‌ها را برگرداند.

Endpoint: /api/book/{id}

این اندپوینت برای دریافت اطلاعات کتاب، از api زیر استفاده می‌کند:

https://get.taaghche.com/v2/book/{id}

قسمت اول:

می‌خواهیم یک api بنویسیم که در آن دو لایه کش داشته باشیم؛ لایه‌ی اول روی memory و لایه‌ی دوم روی redis. بنابراین وقتی یک درخواست به این اندپوینت می‌رسد، اول memory را بررسی می‌کند. اگر داده‌ای در آنجا وجود داشت آن را برمی‌گرداند. اگر وجود نداشت سراغ redis می‌رود. باز هم اگر وجود نداشت، از api بالا استفاده می‌کند و داده را در هر دو لایه‌ی memory و redis کش می‌کند که دفعات بعدی بتواند از آن استفاده کند. دیتای کتاب در redis بدون انقضا کش شود. این API نوشته شده بر روی آدرس زیر در دسترس باشد.

api/book/{id}/

قسمت دوم:

میخواهیم برای کش در لایه مموری و ردیس زمان منقضی شدن منحصر برای هرکدام در نظر بگیریم. به نحوی که پس از گذشت این زمان، کش در لایه مربوطه در دسترس نباشد. این مقدار زمان را بتوانیم از تنظیمات و کانفیگ پروژه عوض کنیم.

فیچر1:

قسمت سوم:

حال میخواهیم یک consumer بر بستر RabbitMQ داشته باشیم که در صورت تغییرات اطلاعات کتاب در سایر پروژه‌های شرکت، یک پیغام با ارسال BookId، برای این صف در Rabbit ارسال کنند و این consumer بعد از دریافت پیام، کش کتاب مورد نظر در هر دو لایه را پاک کند.

قسمت چهارم:

آیا در قسمت سوم به جای پاک کردن کش، می‌توانیم کار دیگری انجام دهیم؟ لطفا کارهایی که می‌توان بعد از دریافت پیام انجام داد را بنویسیم و با هم مقایسه کنیم.

قسمت پنجم:

قسمت ربیت، ردیس و پروژه نوشته شده داکرایز شود و با نوشتن تنها یک دستور docker compose up -d ربیت، ردیس و پروژه (بر روی پورت ۵۰۰۰) بالا بیاید و با صدا زدن آدرس زیر بتوان اطلاعات هر کتابی را دریافت کرد.

localhost:5000/api/book/{id}

این سرویس شرایط زیر را داشته باشد:

1. معماری خوبی داشته باشد و لایه‌ها به‌خوبی از هم مجزا باشند.
2. کد تمیز نوشته شود و قابلیت نوشتن unit test را داشته باشد.
3. امکان تغییر لایه‌های کش در آینده وجود داشته باشد؛ یعنی مثلاً بعداً بتوان به‌جای استفاده از redis از فایل استفاده کرد.

موارد امتیازی:

1. پیاده‌سازی unit test
2. تنظیمات مربوط به مقدار زمان معتبر بودن کش را بتوانیم از فایل Docker compose بدهیم.
3. پروژه Readme داشته باشد و توضیحات مربوط به پروژه را در آن داشته باشیم.

اگر در مورد صورت پروژه سوالی داشتی، می‌توانی از طریق همین ایمیل با ما در ارتباط باشی.

اگر سوالی در مورد تسک داری از راه های زیر می‌تونی بپرسی.

و لطفا جواب تسک رو به ایمیل m.motiee@hasin.recruitee.com بفرست.

خوشحال می‌شیم اگه در مورد تسک بهمون فیدبک بدی.