Objectivo

Implementar uma API server side de comentários que suportem "infinite scroll" do lado do UI em ASP.Net Core, e que permitam ao utilizador saltar para comentários novos.

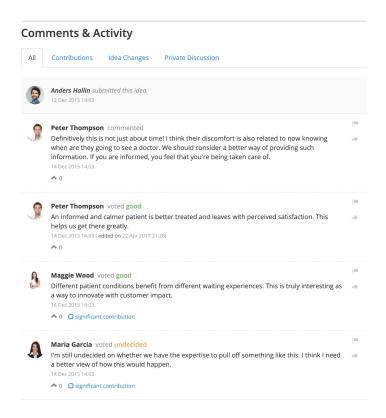
Contamos no final do desafio ter um projecto de ASP.Net core num repositório de git que consigamos descarregar, compilar e correr para testar. Se for um repositório público, não incluas o nome InnovationCast nem IC para evitar que apareça em pesquisas.

Preparamos um esqueleto de ASP.Net Core que podes usar como base em https://github.com/InnovationCast/InnovationCast.Backend.Challenge. Na pasta Requests tens um ficheiro com um exemplo de chamada à API que podes usar para testar e demonstrar o uso do que desenvolveres.

Estamos à procura de uma solução que seja adequada à nossa equipa de 5 a 8 pessoas, com uma arquitectura intermédia entre o código todo num só ficheiro e vinte projectos separados.

Contexto

Um dos componentes críticos do InnovationCast é o que suporta discussões. Permite a um utilizador comentar uma ideia, um artigo, ou outras entidades, e ler os comentários já existentes. Também permite carregar todos os comentários novos de uma entidade para um determinado utilizador.



Uma vez que o InnovationCast é um software desenvolvido para suportar um modelo de inovação colaborativo, queremos melhorar a experiência do componente de discussão, que permite aos utilizadores principalmente comentarem e darem feedback sobre ideias.

O componente também é utilizado noutras áreas do software para além das ideias uma vez que é possível comentar notícias, sinais e outro tipo de entidades.

Âmbito do desafio

Um comentário tem obrigatoriamente:

- o identificador da entidade comentada
- um campo de texto
- um autor
- uma data de publicação

Pode ter mais campos que sejam tidos como necessários ou opcionais.

Os identificadores da entidade comentada e o autor são representados por uma string. O campo de texto é HTML, e a data de publicação é uma data.

O HTML viria de um campo de edição <u>WYSIWYG</u>, para ser renderizado numa feed de comentários, e a data virá em UTC, podendo depois ser transformada numa data no fuso horário de um cliente.

Casos de uso a implementar:

- Listar comentários de uma entidade.
- Listar comentários novos de uma entidade (comentários que um utilizador ainda não viu).
- Submeter novo comentário.

O que está dentro de âmbito:

- Serviços/API
- Estruturas de dados
- Persistência

A API deverá receber dados JSON e a persistência deverá usar RavenDB. O servidor de RavenDB está disponivel online para descarregar (https://ravendb.net/download), existem imagens de docker (https://hub.docker.com/r/ravendb/ravendb/), e cloud (https://cloud.ravendb.net/pricing).

Estamos interessados em explorar a tua aproximação ao problema, a discutir posteriormente na entrevista Tech onde vamos abordar as vantagens e desvantagens da tua solução.

innovationcast[®]

Não precisas de implementar os seguintes pontos, mas vamos falar sobre como abordarias os mesmos na entrevista Tech:

- Validação e sanitização dos dados
- Segurança
- Testes automáticos
- Uso do software em vários países culturas e fusos horários

O que está fora de âmbito, e que podes assumir que é tratado pela infraestrutura:

Autenticação