Documentação da API de Questões

Este projeto é uma aplicação Spring Boot para gerenciamento de questões, alternativas e temas. Utiliza MongoDB como banco de dados e segue um padrão de arquitetura em camadas.

Estrutura do Projeto

Controllers (Controladores)

AlternativaController

- Gerencia requisições HTTP para alternativas
- Endpoints para operações CRUD em alternativas
- Utiliza ResponseEntity para códigos de status HTTP apropriados
- Caminho base: /api/alternativas

QuestaoController

- Gerencia requisições HTTP relacionadas a questões
- Fornece endpoints para criar, ler, atualizar e deletar questões
- Inclui filtragem por tema e nível de dificuldade
- Caminho base: /questoes

TemaController

- Gerencia operações relacionadas a temas
- Fornece operações CRUD básicas para temas
- Caminho base: /temas

Models (Modelos)

Alternativa

- Representa uma alternativa de questão
- Propriedades:
    - id: Identificador único
    - texto: Texto da alternativa
    - correto: Booleano indicando se é a resposta correta
    - dataCriacao: Data de criação
    - dataAtualizacao: Data da última atualização
    - ativo: Status de ativo

Questao

- Representa uma entidade de questão
- Propriedades:
    - id: Identificador único
    - titulo: Título da questão
    - enunciado: Texto da questão
    - temaId: ID do tema associado
    - dificuldade: Nível de dificuldade
    - alternativas: Lista de alternativas
    - visivel: Status de visibilidade
    - dataCriacao/dataAtualizacao: Datas
    - ativo: Status de ativo

Tema

- Representa um tema/categoria de questão
- Propriedades:
    - id: Identificador único
    - nome: Nome do tema
    - descricao: Descrição do tema
    - visivel: Status de visibilidade
    - dataCriacao/dataAtualizacao: Datas
    - ativo: Status de ativo

DTOs (Objetos de Transferência de Dados)

AlternativaDTO

- Objeto de transferência para alternativas
- Usado para requisições/respostas da API
- Inclui todas as propriedades do modelo Alternativa

QuestaoDTO

- DTO para questões
- Inclui anotações de validação
- Contém objetos AlternativaDTO aninhados

TemaDTO

- DTO para temas
- Usado para comunicação da API

Repositories (Repositórios)

AlternativaRepository

- Repositório MongoDB para entidades Alternativa
- Estende MongoRepository

QuestaoRepository

- Repositório MongoDB para entidades Questao
- Métodos personalizados para filtrar questões por:
    - Nível de dificuldade
    - Visibilidade
    - ID do tema

TemaRepository

- Repositório MongoDB para entidades Tema
- Operações CRUD básicas

Services (Serviços)

AlternativaService

- Lógica de negócio para alternativas
- Gerencia conversão entre DTOs e entidades
- Implementa operações CRUD
- Valida dados de alternativas

QuestaoService

- Lógica de negócio principal para questões
- Gerencia relacionamentos com alternativas
- Implementa operações de filtragem e busca
- Realiza validação de dados

TemaService

- Lógica de negócio para temas
- Implementa operações CRUD
- Gerencia validação e conversão de dados

Tratamento de Exceções

GlobalExceptionHandler

- Tratamento centralizado de exceções
- Fornece respostas de erro consistentes
- Trata exceções específicas:
    - ResourceNotFoundException
    - BadRequestException
    - ResourceAlreadyExistsException
    - Exception genérica

Exceções Personalizadas

- BadRequestException: Dados de requisição inválidos

- ResourceNotFoundException: Recurso não encontrado
- ResourceAlreadyExistsException: Recurso duplicado
- ErrorResponse: Estrutura padronizada de resposta de erro

Aplicação Principal

QuestoesApiApplication

- Ponto de entrada da aplicação Spring Boot
- Configura conexão com MongoDB
- Configura contexto da aplicação

Configuração

Propriedades da Aplicação

yaml
Copy
spring:
data:
mongodb: uri:
mongodb+srv://[usuario]:[senha]@cluster0.nvtdo.mongodb.net/ database: questoes-api
server:
port: 8080

Exemplos de Uso da API

Criando um Tema

http
Copy
POST http://localhost:8080/temas
{
"nome": "Astronomia",
"descricao": "Questões relacionadas ao estudo do universo."
}

Criando uma Questão

http
Copy
POST http://localhost:8080/questoes
{
"titulo": "Qual é o maior planeta do sistema solar?",

```
    "enunciado": "Escolha a alternativa correta sobre o maior planeta do sistema solar.",
    "temaId": "679d5971f407da6328b55882",
    "dificuldade": 2,
    "alternativas": [
        {
            "texto": "Terra", "isCorreto":
            false
        },
        {
            "texto": "Júpiter", "isCorreto":
            true
        }
    ],
    "visivel": true, "ativo":
    true
}
```


Tratamento de Erros

A API fornece respostas de erro detalhadas com:

- Timestamp (data e hora)
- Código de status HTTP
- Mensagem de erro
- Caminho onde o erro ocorreu

Documentação da API de Questões

Este projeto é uma aplicação Spring Boot para gerenciamento de questões, alternativas e temas. Utiliza MongoDB como banco de dados e segue um padrão de arquitetura em camadas.

Estrutura do Projeto

Controllers (Controladores)

AlternativaController

- Gerencia requisições HTTP para alternativas
- Endpoints para operações CRUD em alternativas
- Utiliza ResponseEntity para códigos de status HTTP apropriados
- Caminho base: /api/alternativas

QuestaoController

- Gerencia requisições HTTP relacionadas a questões
- Fornece endpoints para criar, ler, atualizar e deletar questões
- Inclui filtragem por tema e nível de dificuldade
- Caminho base: /questoes

TemaController

- Gerencia operações relacionadas a temas
- Fornece operações CRUD básicas para temas
- Caminho base: /temas

Models (Modelos)

Alternativa

- Representa uma alternativa de questão
- Propriedades:
    - id: Identificador único
    - texto: Texto da alternativa
    - correto: Booleano indicando se é a resposta correta
    - dataCriacao: Data de criação
    - dataAtualizacao: Data da última atualização
    - ativo: Status de ativo

Questao

- Representa uma entidade de questão
- Propriedades:
    - id: Identificador único
    - titulo: Título da questão
    - enunciado: Texto da questão
    - temaId: ID do tema associado
    - dificuldade: Nível de dificuldade
    - alternativas: Lista de alternativas
    - visivel: Status de visibilidade
    - dataCriacao/dataAtualizacao: Datas
    - ativo: Status de ativo

Tema

- Representa um tema/categoria de questão
- Propriedades:
    - id: Identificador único
    - nome: Nome do tema
    - descricao: Descrição do tema
    - visivel: Status de visibilidade
    - dataCriacao/dataAtualizacao: Datas
    - ativo: Status de ativo

DTOs (Objetos de Transferência de Dados)

AlternativaDTO

- Objeto de transferência para alternativas
- Usado para requisições/respostas da API
- Inclui todas as propriedades do modelo Alternativa

QuestaoDTO

- DTO para questões
- Inclui anotações de validação
- Contém objetos AlternativaDTO aninhados

TemaDTO

- DTO para temas
- Usado para comunicação da API

Repositories (Repositórios)

AlternativaRepository

- Repositório MongoDB para entidades Alternativa
- Estende MongoRepository

QuestaoRepository

- Repositório MongoDB para entidades Questao
- Métodos personalizados para filtrar questões por:
    - Nível de dificuldade
    - Visibilidade
    - ID do tema

TemaRepository

- Repositório MongoDB para entidades Tema
- Operações CRUD básicas

Services (Serviços)

AlternativaService

- Lógica de negócio para alternativas
- Gerencia conversão entre DTOs e entidades
- Implementa operações CRUD
- Valida dados de alternativas

QuestaoService

- Lógica de negócio principal para questões
- Gerencia relacionamentos com alternativas
- Implementa operações de filtragem e busca
- Realiza validação de dados

TemaService

- Lógica de negócio para temas
- Implementa operações CRUD
- Gerencia validação e conversão de dados

Tratamento de Exceções

GlobalExceptionHandler

- Tratamento centralizado de exceções
- Fornece respostas de erro consistentes
- Trata exceções específicas:
    - ResourceNotFoundException
    - BadRequestException
    - ResourceAlreadyExistsException
    - Exception genérica

Exceções Personalizadas

- BadRequestException: Dados de requisição inválidos
- ResourceNotFoundException: Recurso não encontrado
- ResourceAlreadyExistsException: Recurso duplicado
- ErrorResponse: Estrutura padronizada de resposta de erro

Aplicação Principal

QuestoesApiApplication

- Ponto de entrada da aplicação Spring Boot
- Configura conexão com MongoDB
- Configura contexto da aplicação

Configuração do banco de dados no MongoDB

Utilizamos o banco de dados MongoDB, por ser um sistema que trabalha com questões que podem ter diferentes números de alternativas. Temas podem ter descrições e estruturas variáveis.Outra questão importante pela opção do MongoDB é que nosso aplicativo de quiz de questões de história, caso tenha aumento de questões, o MongoDB é um sistema escalável que pode crescer horizontalmente conforme aumenta o número de questões, alternativas e temas.

O banco de dados é composto por 3 classes implementadas no CRUD, Alternativa, questões e temas.

O quiz é dividido por temas, que têm um grupo de questões cadastradas com alternativas relacionadas a o tema proposto.

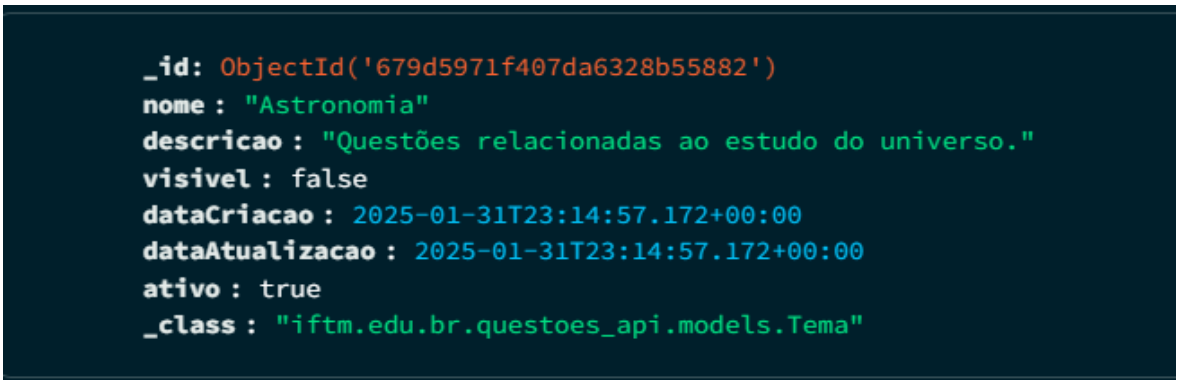
No mongo para cadastrar um tema utilizamos Postman, como exemplo de implementação da alternativa.

Para criar o tema em JSON devemos passar os seguintes atributos:

post

```
{
 "nome": "Astronomia",
 "descricao": "Questões relacionadas ao estudo do universo."
}
```

Após o cadastro podemos visualizar no mongo da seguinte forma:



Para cada questão criada temos alternativas correlacionadas a questão, na implementação das classes utilizamos a anotação @DBRef que cria uma referência entre documentos de diferentes collections, esta notação é similar a uma chave estrangeira em banco de dados SQL. O código é usado para relacionar questões com suas alternativas.

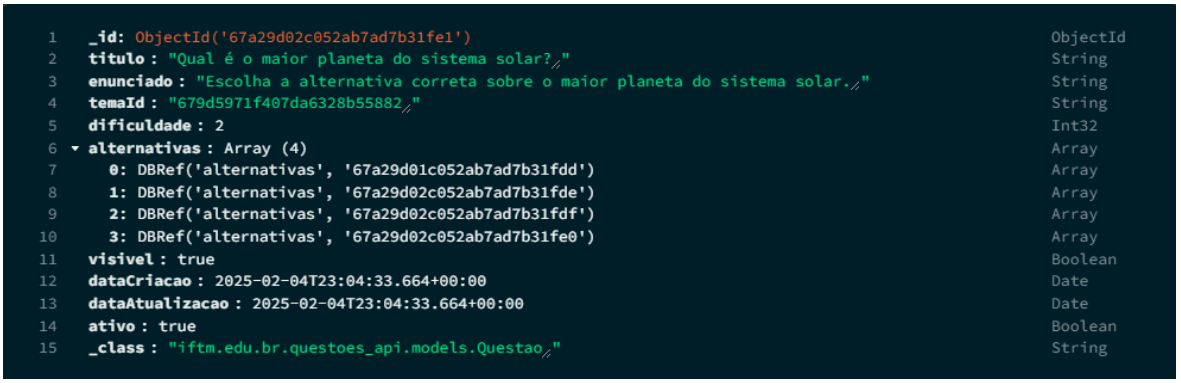
Para criação de uma questão é necessário que já tenha um tema cadastrado, e na criação passamos o ID do tema. E também optamos pelo efeito em cascata, quando é criado uma questão criamos também as alternativas, e também ser for deletar uma questão também irá deletar as alternativas correlacionadas a questão.

Exemplo de criação de uma questão:

```
{
  "titulo": "Qual é o maior planeta do sistema solar?",
  "enunciado": "Escolha a alternativa correta sobre o maior planeta do sistema solar.",
  "temaId": "679d5971f407da6328b55882",
  "dificuldade": 2,
  "alternativas": [
    {
      "texto": "Terra",
      "isCorreto": false
    },
    {
      "texto": "Júpiter",
      "isCorreto": true
    },
    {
      "texto": "Vênus",
      "isCorreto": false
    },
    {
      "texto": "Saturno",
      "isCorreto": false
    }
```

```
  ],
  "visivel": true,
  "ativo": true
}
```

Após a criação da questão a consulta no mongoDB retornará:



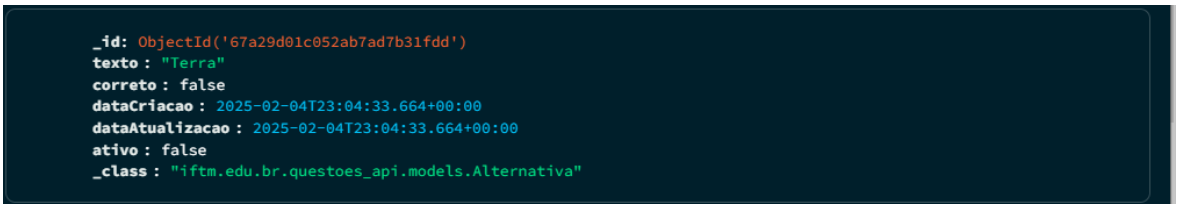
Caso o usuário queira deletar uma questão em formato JSON deve passar o ID da questão cadastrada para ser deletada.

db.questoes.deleteOne({ "_id": ObjectId("67a29d02c052ab7ad7b31fe1") })

No postman utilizamos o seguinte comando:

http://localhost:8080/questoes/67a29d02c052ab7ad7b31fe1

Alternativas após cadastra junto com a questão no mongoDB podemos visualizar da seguinte forma:



Propriedades da Aplicação

yaml
Copy
spring:
data:
mongodb: uri:
mongodb+srv://[usuario]:[senha]@cluster0.nvtdo.mongodb.net/

database: questoes-api server:
port: 8080

A uri define a String de conexão com o cluster MongoDB, incluindo credenciais e endereço do servidor. No código usa uma conexão com MongoDB Atlas.

Esta estrutura NoSQL se mostra adequada para o sistema pois:

Oferece flexibilidade para evolução do modelo de dados

Permite consultas eficientes através de índices

Escala horizontalmente conforme o crescimento do sistema

Simplifica o desenvolvimento através das anotações Spring Data MongoDB

Tratamento de Erros

A API fornece respostas de erro detalhadas com:

- Timestamp (data e hora)
- Código de status HTTP

- Mensagem de erro
- Caminho onde o erro ocorreu