

1) Instruções Gerais

1. Compartilhe o projeto no GitHub com o usuário asoanderson.
 2. O projeto deve conter um README.md com todas as informações necessárias para:
 - ☐ Configuração do ambiente.
 - ☐ Build da solução.
 - ☐ Deploy da aplicação.
 3. Entrega: Compartilhar o repositório até o dia 02/04/2024 às 23:59.
 4. Critérios de Avaliação:
 - ☐ Clareza do código e da lógica aplicada.
 - ☐ Capacidade de organização de ideias.
 - ☐ Estrutura e boas práticas no código.
 - ☐ Performance e escalabilidade.
 - ☐ Implementação da autenticação com JWT.
 - ☐ Desenvolvimento de testes unitários e testes de integração.
-

2) Descrição da Tarefa

Desenvolver uma API REST em Go que gerencie campeonatos esportivos e suas partidas, garantindo segurança com autenticação JWT. O sistema permitirá listar campeonatos e partidas. Além disso, o candidato deve desenvolver testes unitários e testes de integração para garantir o correto funcionamento da aplicação.

3) Requisitos da Aplicação

- A API deve fornecer endpoints protegidos com autenticação JWT.
 - Para acessar os dados, o usuário precisa estar autenticado e obter um token JWT.
 - Os dados dos campeonatos e partidas devem ser obtidos da API externa (ver seção API Externa).
 - Filtros disponíveis para listar partidas:
 - ☐ Campeonato (obrigatório).
 - ☐ Equipe (opcional).
 - ☐ Rodada (opcional).
 - ☐ Os filtros de Equipe e Rodada podem ser usados em conjunto.
 - Desenvolver testes unitários e testes de integração cobrindo os principais fluxos da aplicação.
-

4) Especificação Técnica

- Linguagem: Go.
 - Framework Web: Gin ou Echo.
 - Banco de Dados: Pode ser PostgreSQL.
 - Autenticação: JWT.
 - Testes: Utilizar framework de testes do Go.
 - Containerização: O projeto deve rodar dentro de um Docker (opcional).
-

5) Funcionalidades

5.1 Autenticação JWT

Criar um endpoint REST para login que gere um token JWT.

- Criar um endpoint REST para registrar novos usuários (opcional).

Endpoint para Login:

POST /auth/login

Exemplo de corpo da requisição:

```
{  
  "usuario": "admin",  
  "senha": "123456"  
}
```

Resposta esperada:

```
{  
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."  
}
```

5.2 Listar Campeonatos

- Criar um endpoint REST protegido com JWT para listar campeonatos.
- Os dados devem vir da API externa.

Endpoint:

GET /campeonatos

Resposta esperada:

```
[  
  { "id": "campeonato_001", "nome": "Campeonato Brasileiro", "temporada": "2025" },  
  { "id": "campeonato_002", "nome": "UEFA Champions League", "temporada": "2025" }  
]
```

5.3 Listar Partidas com Filtros

- Criar um endpoint REST protegido com JWT para listar as partidas de um campeonato.
- Os dados devem ser obtidos da API externa.
- O filtro de campeonato é obrigatório.
- Filtros opcionais:
 - Equipe: se informado, exibe apenas jogos da equipe.
 - Rodada: se informado, exibe apenas os jogos da rodada específica.
 - Se ambos os filtros forem usados juntos, exibir apenas os jogos dessa equipe na rodada escolhida.

Endpoint:

GET /campeonatos/{id}/partidas?equipe=Flamengo&rodada=3

Resposta esperada:

```
{
  "rodada": 3,
  "partidas": [
    { "time_casa": "Flamengo", "time_fora": "Palmeiras", "placar": "2-1" }
  ]
}
```

6) API Externa

Os dados dos jogos devem ser consumidos da API:

- URL: <http://api.football-data.org/>
- Documentação: <https://www.football-data.org/documentation/quickstart>
- Token de Autenticação:
Header: X-Auth-Token: 6311a66f5f8746fd8860a5de6173f49f
- Para obter todas as partidas de um campeonato:
GET <http://api.football-data.org/v4/competitions/{id}/matches>
(Onde {id} é o ID da competição.)
- Para retornar os dados dos campeonatos:
GET <http://api.football-data.org/v4/competitions/>

7) Como Entregar

1. Subir o código no GitHub e adicionar o usuário **asoanderson**.
2. O **README.md** deve conter:
 - Passos para rodar o projeto.
 - Exemplos de chamadas à API.
 - Instruções para rodar os testes unitários e testes de integração.
3. O código deve estar estruturado e seguir boas práticas.

Mockup Telas

TELA 1

Tabela do Campeonato

Selecione o campeonato*

TELA 1

Tabela do Campeonato

Selecione o campeonato*

WC | FIFA World Cup

CL | UEFA Champions League

BL | Bundesliga

DED | Eredivisie

BSA | Campeonato Brasileiro Série A

PD | Primera Division

FL1 | Ligue 1

ELC | Championship

PPL | Primeira Liga

EC | European Championship

SA | Serie A

PL | Premier League

CLI | Copa Libertadores

TELA 1

Tabela do Campeonato

Selecione o campeonato *

BSA | Campeonato Brasileiro Série A

Selecione a Equipe

Todas

Selecione a rodada

Todas

Campeonato Brasileiro Série A

1ª Rodada

Equipe A

x

Equipe B

Equipe A

x

Equipe B

Equipe A

x

Equipe B

Equipe A

x

Equipe B

10 Jogos da 1ª Rodada

2ª Rodada

Equipe A

x

Equipe B

Equipe A

x

Equipe B

Equipe A

x

Equipe B

Equipe A

x

Equipe B

10 Jogos da 2ª Rodada

3ª Rodada

Equipe A

x

Equipe B

Equipe A

x

Equipe B

Equipe A

x

Equipe B

Equipe A

x

Equipe B

10 Jogos da 3ª Rodada

.

.

.

TELA 1

Tabela do Campeonato

Selecione o campeonato *

BSA | Campeonato Brasileiro Série A

Selecione a Equipe

Todas

Selecione a rodada

5ª Rodada

Campeonato Brasileiro Série A

5ª Rodada

Palmeiras

x

Cuiabá

Equipe A

x

Equipe B

Equipe A

x

Equipe B

Equipe A

x


Equipe B

10 Jogos da 5ª Rodada

TELA 1

Tabela do Campeonato

Selecione o campeonato *

 BSA | Campeonato Brasileiro Série A

Selecione a Equipe


 Palmeiras

Selecione a rodada

5ª Rodada

Campeonato Brasileiro Série A

5ª Rodada

 Palmeiras

 x

 Cuiabá

Jogos da Equipe

Filtrada na Rodada

Filtrada

Cadastro de Torcedores e Broadcast

1. Cadastro de Torcedores

Objetivo: Permitir que os usuários se cadastrem para receber notificações sobre os jogos do seu time.

- **Endpoint:**
POST /torcedores
- **Descrição:**
Esse endpoint receberá os dados do torcedor e o registrará como destinatário de mensagens. É necessário que sejam enviados, por exemplo, nome, e-mail e o time de interesse.

- **Exemplo de corpo da requisição:**

```
{  
  
  "nome": "João Silva",  
  
  "email": "joao.silva@example.com",  
  
  "time": "Flamengo"  
}
```

- **Resposta esperada:**

```
{  
  
  "id": "torcedor_001",  
  
  "nome": "João Silva",  
  
  "email": "joao.silva@example.com",  
  
  "time": "Flamengo",  
  
  "mensagem": "Cadastro realizado com sucesso"  
}
```

- **Notas:**

- Realizar a validação dos dados (campos obrigatórios, formato de e-mail, etc.).
- Armazenar os dados em uma base de dados ou outro mecanismo de persistência (ou, para testes, em memória).

2. Serviço de Broadcast

Objetivo: Distribuir mensagens aos torcedores cadastrados quando ocorrerem eventos importantes relacionados aos jogos do seu time.

- **Tipos de Mensagens a serem transmitidas:**

1. **Notificação de Início do Jogo:** Informar os torcedores que o jogo vai começar.
2. **Notificação de Fim do Jogo:** Informar o placar final do jogo.

- **Implementação:**

Para essa funcionalidade, pode-se optar por uma das seguintes abordagens:

- **Endpoint REST para Broadcast:**
Criar um endpoint, por exemplo, POST /broadcast, que receba o tipo de evento, time e os detalhes da mensagem, e então distribua as notificações para os torcedores cadastrados que acompanham o time informado.
- **Comunicação em Tempo Real (ex.: WebSockets):**
Implementar uma conexão WebSocket para enviar as mensagens assim que o evento ocorrer.
- **Outros Mecanismos:**
Pode-se utilizar notificações por e-mail, SMS ou push notifications, dependendo do escopo e das tecnologias escolhidas.

- **Exemplo de Payload para Broadcast (Início do Jogo):**

```
{  
  "tipo": "inicio",  
  "time": "Flamengo",  
  "mensagem": "O jogo do Flamengo vai começar em breve!"  
}
```

- **Exemplo de Payload para Broadcast (Fim do Jogo):**

```
{  
  "tipo": "fim",  
  "time": "Flamengo",  
  "placar": "2-1",  
  "mensagem": "O jogo terminou com placar 2-1"  
}
```

- **Fluxo do Processo:**

- Ao receber uma solicitação de broadcast, o sistema identifica todos os torcedores cadastrados que estão interessados no time especificado.
- Em seguida, distribui a mensagem de notificação para cada torcedor.
- Opcionalmente, registra/loga o envio das mensagens para controle e auditoria.

- **Segurança:**

O endpoint ou serviço de broadcast deve ser protegido para que apenas usuários autorizados (por exemplo, administradores) possam enviar essas notificações.

- **Testes:**

Desenvolver testes unitários e testes de integração para:

- Validar o correto cadastro dos torcedores.
 - Verificar se a distribuição das mensagens broadcast ocorre conforme o esperado.
-

3. Considerações Finais

- **Documentação:**

Atualize o **README.md** do projeto com:

- ☐ Instruções para rodar e testar o endpoint de cadastro de torcedores.
- ☐ Exemplos de payloads para envio de mensagens de broadcast.
- ☐ Detalhes sobre a segurança e a forma de envio das notificações.

- **Boas Práticas:**

Garanta que o código esteja estruturado e que as boas práticas de desenvolvimento sejam seguidas, incluindo a escrita de testes que cubram os principais fluxos dessa nova funcionalidade.