ANEXO I

1. Objetivo

O objetivo deste documento é estabelecer o padrão técnico obrigatório para a criação de Interfaces de Programação de Aplicações (APIs) pelas empresas licenciadas pela CGMAC. A finalidade é permitir a consulta e a integração automatizada de dados sobre unidades marítimas (embarcações, plataformas, etc.) em nossa plataforma central de monitoramento para a atividade da APO do FZA-59. A adesão a estas diretrizes garante uma integração rápida, segura e consistente dos dados.

2. Diretrizes para elaboração da API

2.1 Princípios Gerais Obrigatórios

Toda API desenvolvida deve seguir os seguintes princípios:

- Protocolo: A API deve ser acessível via HTTPS para garantir a segurança dos dados em trânsito.
- Arquitetura: A API deve ser construída seguindo os princípios RESTful.
- Formato de Dados: Todas as trocas de dados (requisições e respostas) devem utilizar o formato JSON.
- Versionamento: A API deve incluir um número de versão em sua URL base para gerenciar futuras atualizações sem quebrar a integração (Ex: `/v1`).
- Timestamps: Todos os campos de data e hora devem seguir o padrão ISO 8601 em UTC, finalizando com o designador `Z`.

2.2. Autenticação (OAuth 2.0)

As requisições à API devem ser autenticadas utilizando o fluxo "*OAuth 2.0 Client Credentials"*, que é o padrão de mercado para comunicação segura entre servidores.

- Processo de Obtenção de Acesso:

1. Credenciais Iniciais: A empresa deverá fornecer um `client_id` e um `client_secret` únicos.
2. Obtenção do Token de Acesso: Nosso sistema fará uma requisição `POST` a um endpoint `/auth/token` para trocar as credenciais por um `access_token` temporário.

- Endpoint: `POST /auth/token`
- Request Body (`application/x-www-form-urlencoded`):

- `grant_type`: "client_credentials"
- `client_id`: (seu client_id)
- `client_secret`: (seu client_secret)

- Exemplo de Resposta:

JSON

```
{
    "access_token": "um_token_jwt_secreto_e_temporario",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

3. Uso do Token: Para acessar qualquer rota de dados, nosso sistema enviará o `access_token` no cabeçalho HTTP `Authorization`.

Exemplo: `Authorization: Bearer um_token_jwt_secreto_e_temporario`

2.3. Rate Limiting (Limite de Requisições)

Para garantir a estabilidade e o uso justo da API, deve haver um limite no número de requisições. A API deve retornar um código de status `429 Too Many Requests` se o limite for excedido.

- Diretriz: Recomenda-se um limite razoável, como 100 requisições por minuto por credencial. A documentação da API deve informar o limite exato.

2.4. Modelos de Dados (Schemas)

Os objetos JSON retornados pelas rotas da API devem seguir estritamente as estruturas abaixo.

2.4.1. Objeto `UnidadeMaritima`

Representa as informações cadastrais e de disponibilidade de uma unidade.

Atributo	Obrigatório?	Tipo	Descrição
nome	Sim	String	Nome comercial ou de operação da unidade.
imo	Não	String	Número IMO - Organização Marítima Internacional (7 dígitos). Nulo se não aplicável.
mmsi	Sim	String	(Chave Principal) Número MMSI - Identidade do Serviço Móvel Marítimo (9 dígitos).
tipoUnidade	Sim	Enum (String)	Categoria da unidade. Deve ser um dos valores listados na Seção 2.4.3

2.4.2. Objeto `PosicaoAIS`

Representa a última posição geográfica conhecida da unidade.

Atributo	Obrigatório?	Tipo	Descrição
mmsi	Sim	String	MMSI da unidade a qual a posição se refere.
latitude	Sim	Number	Coordenada de latitude em formato decimal.
longitude	Sim	Number	Coordenada de longitude em formato decimal.
timestampAquisicao	Sim	Timestamp	Data e hora exata do registro da coordenada pelo AIS.

2.4.3. Enum `tipoUnidade`

O campo `tipoUnidade` no objeto `UnidadeMaritima` deve conter exatamente um dos seguintes valores:

Valor	Descrição
EMBARCACAO_EMERGENCIA	Embarcação dedicada para as atividades de emergência.

EMBARCACAO_APOIO_FAUNA	Embarcação que faz atividades de apoio às atividade de fauna.
EMBARCACAO_EMERGENCIA_APOIO	Embarcação que faz atividades de apoio e atividades de emergência.
PLATAFORMA_PERFURACAO	Unidades marítimas que podem fazer perfuração ou intervenções em poços.

2.5. Endpoints Obrigatórios

A API de cada parceiro deve implementar os dois endpoints a seguir.

2.5.1. Lista de Unidades Marítimas

Esta rota deve retornar a lista completa de unidades marítimas gerenciadas pela sua empresa.

`GET /unidades`

2.5.2. Posição Geográfica da Unidade

Esta rota deve retornar os dados de geolocalização mais recentes para uma unidade específica, consultada pelo seu MMSI.

`GET /posicao/{mmsi}`

2.6. Tratamento de Erros

A API deve utilizar os códigos de status HTTP padrão. O corpo da resposta de erro deve ser um objeto JSON seguindo o formato abaixo.

Formato do Corpo do Erro:

JSON

```
{
    "error": "codigo_do_erro",
    "error_description": "Uma descrição clara do que aconteceu."
}
```

Exemplos de Erros:

Status HTTP	**error (código)**	**error_description (exemplo)**
401 Unauthorized	invalid_token	"Token de autenticação inválido ou expirado."
404 Not Found	not_found	"A unidade marítima com mmsi 'XXXXXXXXX' não foi encontrada."
429 Too Many Requests	rate_limit_exceeded	"Limite de requisições excedido. Tente novamente mais tarde."
500 Internal Server Error	internal_server_error	"Ocorreu uma falha inesperada no servidor." - o envio de um corpo de resposta no formato JSON acima é uma recomendação, mas não é obrigatório, pois a falha pode impedir servidor de processar e formatar uma resposta customizada.

2.7. Processo de Onboarding

O processo para integrar a API à nossa plataforma seguirá quatro etapas claras:

1. **Desenvolvimento:** A equipe de TI da empresa desenvolve os endpoints da API, seguindo estritamente todas as especificações técnicas, de segurança e de modelos de dados contidas neste documento.
2. **Documentação Interativa (OpenAPI/Swagger):**

- É obrigatório gerar e disponibilizar uma documentação interativa da API usando o padrão OpenAPI 3.0 (ou superior).
- Essa documentação (geralmente via Swagger UI ou ReDoc) deve ser acessível publicamente através de uma URL, como por exemplo: https://api.suaempresa.com/v1/docs.
- Ela servirá como a principal ferramenta para os testes de validação.

3. **Disponibilização das Credenciais:** Após o desenvolvimento e a publicação da documentação, a empresa deve nos fornecer as seguintes informações:

- A URL Base da API (ex: https://api.suaempresa.com/v1).
- A URL da documentação OpenAPI/Swagger.

- As credenciais client_id e client_secret para autenticação OAuth 2.0.

4. **Validação e Homologação:**

- A CGMAC utilizará a documentação interativa para realizar uma bateria de testes, verificando cada endpoint, o formato das respostas e o tratamento de erros.
- Uma vez que a API seja validada e comprovada como 100% conforme a esta nota técnica, ela será homologada e a integração em nosso ambiente de produção será informada.