Api de Rastreamento

1. Introdução

A API de Rastreamento é uma interface de programação que permite o acesso programático aos dados e funcionalidades do sistema de gerenciamento de frota GPSCorporativo. Esta API fornece endpoints REST para autenticação, monitoramento de veículos em tempo real, recuperação de históricos de percursos, telemetria avançada, envio de mensagens para dispositivos de bordo, e controle de atuadores do veículo.

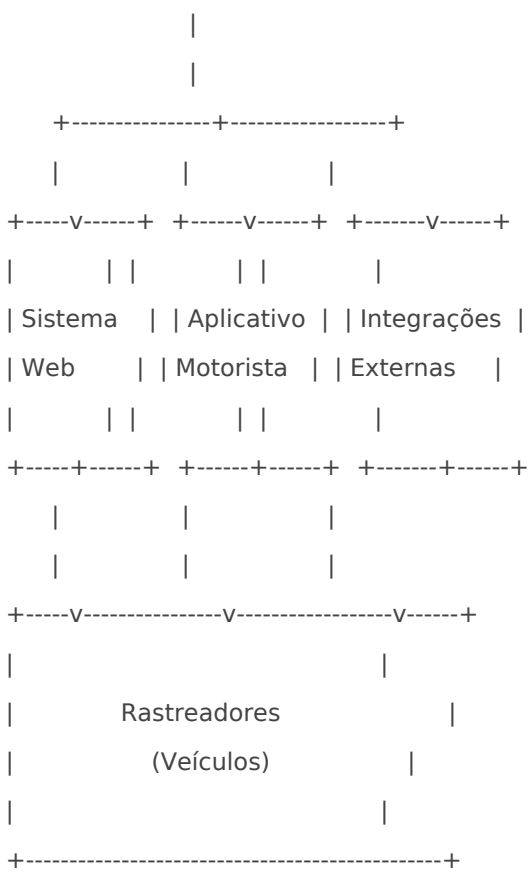
A API utiliza o formato JSON para comunicação de dados e segue o padrão REST com métodos HTTP como GET e POST. Ela requer autenticação via login e implementa controles anti-abuso que limitam a frequência de requisições.

Esta documentação descreve uma coleção do Postman denominada "Testes", que contém requisições para interagir com a API do sistema de rastreamento GPSCorporativo. A coleção foi projetada para facilitar o teste e a integração com os serviços de rastreamento, monitoramento e comunicação com veículos através de uma interface REST.

Informações da Coleção do Postman

- **Nome da Coleção**: Testes
- **ID Postman**: 65270104-d3fd-4853-a9e9-6f3062fbfb22
- **Esquema**: https://schema.getpostman.com/json/collection/v2.1.0/collection.json
- **Servidor Base**: rastreamento.concept.inf.br (definido como variável de ambiente)
- **Base Path**: /GPSCorporativo/rest/

2. Diagrama Conceitual do Sistema



O diagrama acima ilustra como os diferentes componentes do sistema GPSCorporativo se comunicam entre si. A API de Rastreamento atua como interface central que permite a comunicação bidirecional entre:

1. O Sistema Web (usado pelos gestores de frota)
2. O Aplicativo do Motorista (usado pelos condutores dos veículos)
3. Integrações Externas (sistemas de terceiros)
4. Os Rastreadores dos Veículos (que enviam dados de telemetria e recebem comandos)

Este fluxo de comunicação permite tanto o monitoramento quanto o controle remoto da frota.

3. Glossário Técnico

Termo	Definição
Rotograma	Planejamento detalhado de rota que inclui pontos de parada, tempos estimados e distâncias a percorrer.
Itinerário	Sequência ordenada de locais a serem visitados pelo veículo, com horários previstos.
Roteirização	Processo de planejamento e otimização de rotas para uma frota de veículos.

Termo	Definição
Telemetria	Conjunto de dados técnicos transmitidos pelo veículo, como velocidade, RPM, nível de combustível, etc.
Atuador	Dispositivo que controla mecanismos do veículo remotamente (ex: bloqueio de motor).
Ponto de Interesse (POI)	Local geograficamente definido que possui relevância para a operação.
Teclado	Dispositivo de bordo que permite comunicação bidirecional entre a central e o motorista.
JSESSIONID	Token de sessão usado para autenticação nas requisições após o login inicial.
Avisos	Notificações automáticas geradas pelo sistema (ex: excesso de velocidade, desvio de rota).
Identificador de Posição	Código único que identifica um registro específico de localização no sistema.

4. Arquitetura e Características Técnicas

4.1. Características Gerais

- **Tipo**: REST API
- **Formato de Dados**: JSON
- **Autenticação**: Baseada em sessão (JSESSIONID)
- **Tempo da sessão**: 120 segundos. Após 120 segundos, a sessão é expirada e precisa de uma nova autenticação.
- **Controle de Taxa**: Implementado em todos os endpoints (rate limiting)
- **Versão Atual**: 1.5 (Setembro 2024)

4.2. Controle Anti-Abuso

A API implementa um mecanismo de controle anti-abuso que limita as requisições por usuário:

- Armazena timestamps das últimas requisições em um hashmap
- Verifica o intervalo entre requisições (varia de 5 a 60 segundos dependendo do endpoint)
- Retorna mensagem de erro se novas requisições forem feitas dentro do intervalo mínimo

4.3. Fluxo de Trabalho da API

1. **Autenticação**: O cliente deve primeiro se autenticar usando o endpoint de autenticação
2. **Obtenção do Token de Sessão**: Após autenticação bem-sucedida, um JSESSIONID é retornado
3. **Chamadas às APIs**: O cliente pode fazer chamadas aos diversos endpoints usando o JSESSIONID
4. **Controle de Sessão**: A sessão tem duração limitada e pode precisar ser renovada

5. Endpoints e Operações

5.1. Autenticação

Descrição: Autentica o usuário no sistema de rastreamento, obrigatório para utilização dos demais métodos.

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/autenticar

Método: POST

Content-Type: application/x-www-form-urlencoded

Parâmetros:

- `login` (String): CPF ou CNPJ do cliente
- `senha` (String): Senha de acesso ao sistema
- `tipoAcesso` (String): Tipo de acesso ("master" ou "grupo")

Validações:

- Login precisa ser um CPF ou CNPJ válido
- Intervalos entre requisições deve ser superior a 5 segundos
- Intervalos entre autenticações do mesmo usuário deve ser superior a 120 segundos

Retorno em caso de sucesso:

```
{
  "id": 123,
  "nome": "Nome do Cliente",
  "idVeiculo": 456,
  "idGrupoVeiculo": 789,
```

```
  "mensagemParaCliente": "Mensagem opcional para o cliente",
  "jsesionId": "ABC123XYZ"
}
```


Retorno em caso de erro:


```
{
  "id": 0
}
```


Instruções passo a passo:

1. Prepare os parâmetros `login`, `senha` e `tipoAcesso`
2. Envie uma requisição POST para o endpoint de autenticação
3. Armazene o valor de `jsesionId` retornado para utilizar nas próximas requisições
4. Inclua o `jsesionId` como cookie HTTP em todas as requisições subsequentes

5.2. Posição Atual dos Veículos

Descrição: Retorna a posição atual de todos os veículos associados ao usuário autenticado.

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/rastrearVeiculos

Método: GET

Content-Type: application/x-www-form-urlencoded

Parâmetros: Nenhum (utiliza o JSESSIONID para identificação)

Validações:

- Intervalo entre requisições deve ser superior a 30 segundos
- Usuário deve estar autenticado

Retorno em caso de sucesso:


```
[
  {
    "idVeiculo": "123",
    "latitude": "-19.9322",
    "longitude": "-43.9352",
    "ignicao": "Ligado",
    "modelo": "F-4000",
```

```
    "tipoVeiculo": "CAMINHAOA",
    "velocidadeAtual": "60",
    "dataUltimaAtualizacao": "10:25 12/03/2025",
    "tempoParado": "00:00:00",
    "condutor": "João da Silva",
    "cpfCondutor": "123.456.789-00",
    "placa": "ABC1234",
    "ultimaLeituraSensorRS232": "Sensor data",
    "ultimoOdometro": "12345",
    "ultimoHorimetro": "987",
    "statusEntrada1": "false",
    "statusEntrada2": "true",
    "statusEntrada3": "false",
    "statusEntrada4": "false",
    "statusSaida1": "true",
    "statusSaida2": "false",
    "statusSaida3": "false"
  }
]
```


Retorno em caso de erro:


```
{
  "codigo": 99
}
```


Instruções passo a passo:

1. Certifique-se de estar autenticado e possuir um JSESSIONID válido
2. Envie uma requisição GET para o endpoint
3. Processe a lista de veículos retornada
4. Respeite o intervalo mínimo entre requisições (30 segundos)

5.3. Percursos Anteriores

Descrição: Recupera o histórico de percurso de um veículo específico em um intervalo de datas.

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/rastrearVeiculoPorData

Método: POST

Content-Type: application/x-www-form-urlencoded

Parâmetros:

- `idVeiculo` (String): Identificador do veículo
- `dataHoraInicio` (String): Data e hora de início no formato YYYYMMDDHHmmss
- `dataHoraFim` (String): Data e hora de fim no formato YYYYMMDDHHmmss
- `somenteParadas` (String, opcional): "true" para retornar apenas paradas

Validações:

- Intervalo entre requisições deve ser superior a 60 segundos
- `idVeiculo` deve ser um número inteiro válido
- Datas devem estar no formato YYYYMMDDHHmmss
- `dataHoraInicio` deve ser anterior a `dataHoraFim`
- O intervalo máximo permitido é de 7 dias

Retorno em caso de sucesso:


```
[
  {
    "indice": "1",
    "latitude": "-19.9322",
    "longitude": "-43.9352",
    "dataHoraGPS": "10:25 12/03/2025",
    "velocidadeAtual": "60",
    "odometro": "12345",
    "horimetro": "987",
    "tempoParado": "00:00:00",
    "ignicao": "Ligado",
    "descricaoPol": "Nome do ponto de interesse"
  }
]
```


Retorno em caso de erro:


```
{
  "mensagem": "Mensagem de erro específica"
}
```


Instruções passo a passo:

1. Prepare os parâmetros `idVeiculo`, `dataHoraInicio` e `dataHoraFim`
2. Envie uma requisição POST para o endpoint
3. Processe a lista de pontos de rastreamento retornada

4. Atenção ao limite de 7 dias para o intervalo de pesquisa

5.4. Consulta de Posição por Identificador

Descrição: Recupera registros de rastreamento a partir de um identificador de posição para um ou mais veículos.

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/rastrearPorIdentificador

Método: POST

Content-Type: application/x-www-form-urlencoded

Parâmetros:

- `idVeiculo` (String, opcional): Identificador do veículo
- `identificadorPosicao` (String): Identificador único da posição

Validações:

- Intervalo entre requisições deve ser superior a 30 segundos
- `identificadorPosicao` deve ser fornecido
- Se `idVeiculo` for fornecido, deve ser um número inteiro

Retorno em caso de sucesso:


```
[
 [
  {
   "indice": "1",
   "idVeiculo": "123",
   "identificadorPos": "789012345",
   "latitude": "-19.9322",
   "longitude": "-43.9352",
   "dataRecebimentoGMT_3": "10:25 12/03/2025",
   "dataHoraGPSGMT_3": "10:24 12/03/2025",
   "velocidade": "60",
   "ignicao": "Ligado",
   "odometro": "12345",
   "horimetro": "987",
   "orientacao": "180"
  }
 ]
```

```
]
```

Retorno em caso de erro:

```
{
  "mensagem": "Mensagem de erro específica"
}
```

Instruções passo a passo:

1. Prepare o parâmetro `identificadorPosicao` (e opcionalmente `idVeiculo`)
2. Envie uma requisição POST para o endpoint
3. Processe a matriz de pontos de rastreamento retornada
4. Observe que o retorno é uma matriz de matrizes, com cada matriz interna representando um veículo

5.5. Consulta de Dados de Telemetria

Descrição: Recupera dados detalhados de telemetria de um veículo em um intervalo de datas.

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/listarTelemetriaPorData

Método: POST

Content-Type: application/x-www-form-urlencoded

Parâmetros:

- `idVeiculo` (String): Identificador do veículo
- `dataHoraInicio` (String): Data e hora de início no formato YYYYMMDDHHmmss
- `dataHoraFim` (String): Data e hora de fim no formato YYYYMMDDHHmmss

Validações:

- Intervalo entre requisições deve ser superior a 30 segundos
- Todos os parâmetros são obrigatórios
- `idVeiculo` deve ser um número inteiro
- Datas devem estar no formato YYYYMMDDHHmmss
- `dataHoraInicio` deve ser anterior a `dataHoraFim`
- O intervalo máximo permitido é de 7 dias

Retorno em caso de sucesso:

```
[
  {
    "indice": "1",
    "latitude": "-19.9322",
    "longitude": "-43.9352",
    "dataHoraGPS": "10:25 12/03/2025",
    "velocidadeAtual": "60",
    "odometro": "12345",
    "horimetro": "987",
    "ignicao": "Ligado",
    "rpm": "1800",
    "temperaturaMotor": "90",
    "combustivelUtilizado": "1.2",
    "nivelCombustivel": "45",
    "nivelCombustivelPercentual": "75",
    "pressaoPedalAcelerador": "80",
    "tempoEmMovimentoTotal": "120",
    "totalCombustivelUtilizado": "230.5",
    "totalCombustivelUtilizadoParado": "12.3",
    "pesoEixos": "2500",
    "informacaoTacografo": "Info do tacógrafo",
    "indicadores": "Indicadores diversos",
    "portaMotoristaAberta": "false",
    "portaPassageiroAberta": "false",
    "portaTraseiraEsquerdaAberta": "false",
    "portaTraseiraDireitaAberta": "false",
    "portaMalasAberto": "false",
    "caputAberto": "false",
    "cintoSegurancaLigado": "true",
    "arCondicionadoLigado": "true",
    "farolLigado": "true",
    "isExcessoVelocidade": "false",
    "corHexaFaixaRPM": "#00FF00",
    "pressaoOleoMotor": "3.5"
  }
]
```


Retorno em caso de erro:

```
{
  "mensagem": "Mensagem de erro específica"
}
```

Instruções passo a passo:

1. Prepare os parâmetros `idVeiculo`, `dataHoraInicio` e `dataHoraFim`
2. Envie uma requisição POST para o endpoint
3. Processe a lista de dados de telemetria retornada
4. Atenção ao limite de 7 dias para o intervalo de pesquisa

5.6. Listar Avisos por Veículo e Data

Descrição: Lista os avisos gerados por um veículo em uma data específica.

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/listarAvisos

Método: POST

Content-Type: application/x-www-form-urlencoded

Parâmetros:

- `idVeiculo` (String): Identificador do veículo
- `data` (String): Data no formato YYYYMMDDHHmmss

Validações:

- Intervalo entre requisições deve ser superior a 5 segundos
- Ambos os parâmetros são obrigatórios

Retorno em caso de sucesso:

```
{
  "lista": [
    {
      "id": "123",
      "tipo": "ExcessoVelocidade",
      "tipoDescricao": "Excesso de Velocidade",
      "condutor": "João da Silva",
      "dataHora": "10:25 12/03/2025",
      "latitude": "-19.9322",
      "longitude": "-43.9352",
```

```
    "descricao": "Veículo ultrapassou o limite de velocidade",
    "dataHoraEnvioEmail": "10:26 12/03/2025"
   }
  ]
}
```


Retorno em caso de erro:


```
{
 "codigo": 99
}
```


Instruções passo a passo:

1. Prepare os parâmetros `idVeiculo` e `data`
2. Envie uma requisição POST para o endpoint
3. Processe a lista de avisos retornada
4. O retorno inclui um resumo e detalhes dos avisos no período

5.7. Acionar Atuadores

Descrição: Controla atuadores do veículo (bloqueio, saída 2 e saída 3).

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/acionarAtuadores

Método: POST

Content-Type: application/x-www-form-urlencoded

Parâmetros:

- `idVeiculo` (String): Identificador do veículo
- `comando` (String): Comando a ser enviado
- `senhaContato` (String): Senha de contato para autorização

Comandos disponíveis:

- Saída 1 (Bloqueio): "BLOQUEAR" / "DESBLOQUEAR"
- Saída 2: "SPC_OUTPUT2_ACTIVATE" / "SPC_OUTPUT2_DESACTIVATE"
- Saída 3: "SPC_OUTPUT3_ACTIVATE" / "SPC_OUTPUT3_DESACTIVATE"

Validações:

- Intervalo entre requisições deve ser superior a 5 segundos

- Todos os parâmetros são obrigatórios
- Comando deve ser um dos valores permitidos

Retorno em caso de sucesso:

```
{
  "mensagem": "Comando enviado com sucesso"
}
```

Retorno em caso de erro:

```
{
  "codigo": 99
}
```

Instruções passo a passo:

1. Prepare os parâmetros `idVeiculo`, `comando` e `senhaContato`
2. Envie uma requisição POST para o endpoint
3. Verifique a mensagem de retorno para confirmar o envio
4. Aguarde pelo menos 5 segundos antes de enviar outro comando

5.8. Listar Mensagens do Teclado

Descrição: Recupera mensagens trocadas com o dispositivo de bordo (teclado).

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/listarMensagensTeclado

Método: POST

Content-Type: application/x-www-form-urlencoded

Parâmetros:

- `idVeiculo` (String): Identificador do veículo
- `cpfMotorista` (String, opcional): CPF do motorista
- `dataInicioMensagem` (String, opcional): Data de início da mensagem (YYYYMMDDHHmmss)
- `dataFimMensagem` (String, opcional): Data de fim da mensagem (YYYYMMDDHHmmss)
- `dataInicioRecebidaMensagem` (String, opcional): Data de início do recebimento (YYYYMMDDHHmmss)
- `dataFimRecebidaMensagem` (String, opcional): Data de fim do recebimento (YYYYMMDDHHmmss)

Validações:

- Intervalo entre requisições deve ser superior a 5 segundos
- `idVeiculo` é obrigatório
- Pelo menos um conjunto de datas (mensagem ou recebimento) deve ser informado
- Datas devem estar no formato correto

Retorno em caso de sucesso:


```
[
 {
  "placa": "ABC1234",
  "dataHora": "10:25 12/03/2025",
  "codigo": "123",
  "mensagem": "Texto da mensagem",
  "origem": "Motorista",
  "statusComandoMensagem": "Entregue",
  "odometroMensagem": "12345",
  "horimetroMensagem": "987",
  "latitudeMensagem": "-19.9322",
  "longitudeMensagem": "-43.9352",
  "localizacaoMensagem": "Av. Amazonas, 1234",
  "cpfMotorista": "123.456.789-00",
  "nomeMotorista": "João da Silva",
  "codigoIdentificaoLeitor": "XYZ123",
  "matriculaMotorista": "M1234"
 }
]
```


Retorno em caso de erro:


```
{
 "mensagem": "Mensagem de erro específica"
}
```


Instruções passo a passo:

1. Prepare o parâmetro `idVeiculo` e pelo menos um conjunto de datas
2. Envie uma requisição POST para o endpoint
3. Processe a lista de mensagens retornada
4. Observe que o veículo precisa estar associado ao cliente autenticado

5.9. Enviar Mensagem para Teclado

Descrição: Envia uma mensagem para o dispositivo de bordo (teclado) do veículo.

URL: http://rastreamento.concept.inf.br/GPSCorporativo/rest/enviarMensagemTeclado

Método: POST

Content-Type: application/x-www-form-urlencoded

Parâmetros:

- `idVeiculo` (String): Identificador do veículo
- `mensagem` (String): Texto da mensagem a ser enviada

Validações:

- Intervalo entre requisições deve ser superior a 5 segundos
- Ambos os parâmetros são obrigatórios

Retorno em caso de sucesso:

```
{
  "mensagem": "Mensagem enviada com sucesso"
}
```

Retorno em caso de erro:

```
{
  "codigo": 99
}
```

Instruções passo a passo:

1. Prepare os parâmetros `idVeiculo` e `mensagem`
2. Envie uma requisição POST para o endpoint
3. Verifique a mensagem de retorno para confirmar o envio
4. Aguarde pelo menos 5 segundos antes de enviar outra mensagem

6. Regras de Negócio e Validações

6.1. Regras Gerais

1. **Autenticação e Autorização**:

- Usuário deve estar autenticado para acessar qualquer endpoint (exceto o de autenticação)
- Usuário só pode acessar veículos associados à sua conta
- Existem dois tipos de acesso: "master" e "grupo"

2. **Controle de Taxa de Requisições**:
   - Cada endpoint tem um limite de frequência de chamadas
   - O IP e o login do usuário são usados para identificação
   - Requisições feitas antes do intervalo mínimo são rejeitadas
3. **Validação de Datas**:
   - Datas devem seguir o formato YYYYMMDDHHmmss
   - Data inicial deve ser anterior à data final
   - Intervalo máximo em consultas históricas é de 7 dias
   - Datas são armazenadas e processadas em UTC mas exibidas em GMT-3
4. **Formato de IDs e Parâmetros**:
   - IDs de veículos devem ser números inteiros
   - Identificadores de posição devem ser números inteiros ou strings
   - Comandos de atuadores devem ser um dos valores predefinidos

6.2. Validações Específicas

1. **Bloqueio de Veículo**:
   - Requer senha de contato para autorização
   - A operação é registrada em log para auditoria
   - Limitado a uma operação a cada 5 segundos
2. **Mensagens para Teclado**:
   - Tamanho da mensagem pode ser limitado pelo dispositivo de bordo
   - Enviadas em tempo real quando o veículo está conectado
3. **Consulta de Histórico**:
   - Opção para filtrar apenas paradas (economiza dados)
   - Limite de 7 dias para evitar sobrecarga do sistema

7. Troubleshooting e Resolução de Problemas

7.1. Diagnóstico de Problemas Comuns

Problema	Possíveis Causas	Soluções

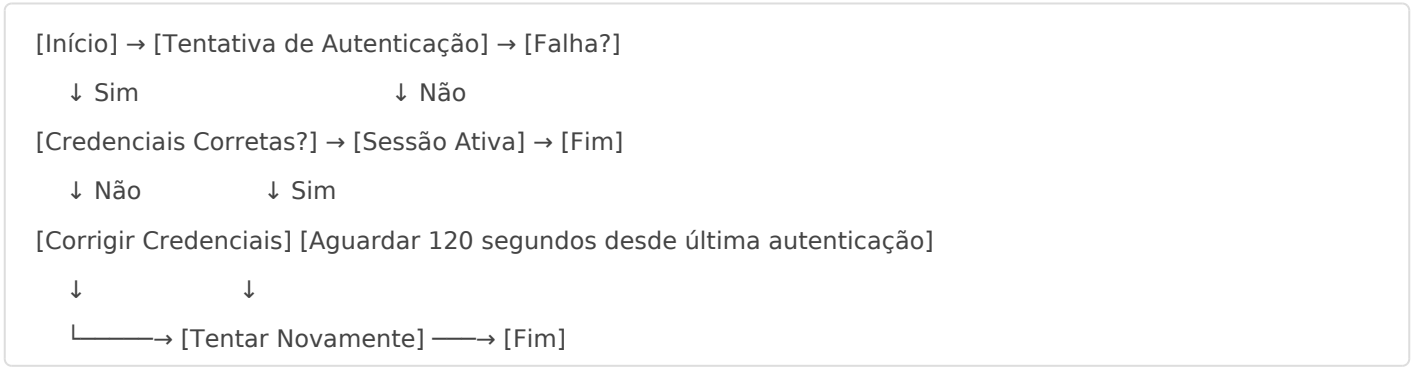
Autenticação sempre retorna 0	1. CPF ou CNPJ errados 2. Senha incorreta 3. Não informou os pontos, hífem e barra no CPF ou CNPJ	1. Informar o CPF e CNPJ com pontos, hífem e barra 2. Confirmar a senha na opções da plataforma WEB em Cofnguração -> Controle de Acesso 3. Verificar se o tipo usado é grupo para as senhas criadas no controle de aceso

7.2. Tabela de Códigos de Erro

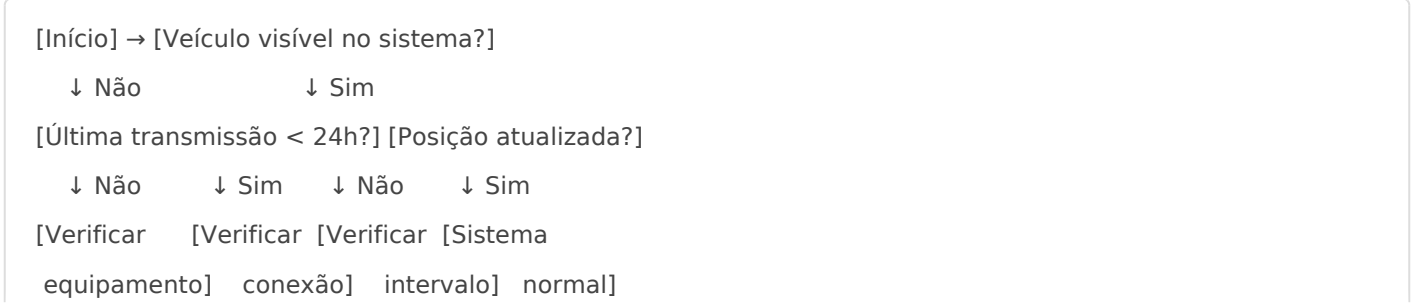
Código	Descrição	Procedimento de Resolução
99	Usuário não autenticado ou sessão expirada	Realizar nova autenticação através do endpoint correspondente
101	Múltiplas requisições em intervalo proibido	Aguardar o tempo mínimo entre requisições para o endpoint específico
102	Parâmetros inválidos ou ausentes	Verificar documentação e corrigir parâmetros da requisição

7.3. Fluxogramas de Decisão

7.3.1. Diagnóstico de Falhas de Autenticação



7.3.2. Resolução de Problemas de Rastreamento



```
↓       ↓       ↓       ↓
└──────────┴──────────┴──────────→ [Fim]
```

8. Versões

Versão	Data de Lançamento	Principais Mudanças
1.5	Setembro 2024	Adicionado suporte para sensores adicionais de telemetria e melhorias na segurança
1.4	Março 2024	Implementada telemetria avançada e relatórios de consumo de combustível
1.3	Outubro 2023	Adicionado controle remoto de atuadores e comunicação bidirecional
1.2	Maio 2023	Melhorias na performance e suporte a múltiplos veículos por requisição
1.1	Janeiro 2023	Correções de bugs e expansão da documentação
1.0	Setembro 2022	Versão inicial com funcionalidades básicas de rastreamento

9. Limitações Conhecidas do Sistema

1. **Intervalo de Consultas Históricas**: Consultas de histórico são limitadas a um período máximo de 7 dias para evitar sobrecarga do sistema.
2. **Frequência de Atualização**: A posição atual dos veículos é atualizada conforme a configuração do dispositivo, tipicamente entre 30 segundos e 5 minutos.
3. **Conexão com Dispositivos**: Comandos enviados para veículos só são processados quando o dispositivo está online e com conectividade.
4. **Tamanho de Mensagens**: Mensagens enviadas para o teclado do motorista são limitadas a 160 caracteres por limitações de hardware.
5. **Simultaneidade de Comandos**: O sistema não permite o envio simultâneo de múltiplos comandos para o mesmo veículo. É necessário aguardar a confirmação do primeiro comando.
6. **Retenção de Dados**: Os dados históricos detalhados ficam disponíveis por até 5 anos. Depende do plano contratado pelo cliente. Após esse período,os dados são apagados.

7. **Precisão Geográfica**: A precisão da localização depende do dispositivo GPS instalado no veículo e das condições ambientais, podendo variar entre 2 e 15 metros.
8. **Compatibilidade de Navegadores**: A interface web é otimizada para Chrome, Firefox e Edge. Outros navegadores podem apresentar limitações visuais.
9. **Atuação em Áreas sem Cobertura**: Veículos em áreas sem cobertura móvel não receberão comandos até retornarem a uma área com sinal.

10. Perguntas Frequentes (FAQ)

1. **Como obtenho o identificador (ID) de um veículo?**
   - O ID do veículo pode ser obtido através do endpoint "Posição Atual dos Veículos" ou ao realizar a autenticação, caso o usuário tenha um veículo único associado.
2. **Quanto tempo dura uma sessão após autenticação?**
   - A sessão tem duração de 120 segundos conforme documentado na seção 4.1. Após esse período, é necessário autenticar-se novamente.
3. **Por que recebo um erro de "múltiplas requisições"?**
   - Cada endpoint tem um limite de frequência de chamadas. Você deve respeitar o intervalo mínimo entre requisições (de 5 a 60 segundos, dependendo do endpoint).
4. **Como enviar um comando de bloqueio/desbloqueio de veículo?**
   - Utilize o endpoint "Acionar Atuadores" com o parâmetro `comando` igual a "BLOQUEAR" ou "DESBLOQUEAR" e forneça a senha de contato para autorização.
5. **Quais formatos de data são aceitos pela API?**
   - A API aceita apenas datas no formato YYYYMMDDHHmmss (exemplo: 20250312102530 para 12/03/2025 10:25:30).
6. **Por que minha consulta de histórico retorna "Rota máxima pesquisável é no intervalo de 7 dias"?**
   - Para evitar sobrecarga do sistema, as consultas de histórico estão limitadas a um intervalo máximo de 7 dias.
7. **O que significa o código de erro 99?**
   - O código 99 indica que o usuário não está autenticado ou que a sessão expirou.
8. **Como faço para consultar os avisos de um veículo?**
   - Utilize o endpoint "Listar Avisos por Data" fornecendo o ID do veículo e a data desejada.
9. **Posso monitorar múltiplos veículos simultaneamente?**
   - Sim, o endpoint "Posição Atual dos Veículos" retorna informações de todos os veículos associados ao usuário autenticado.
10. **Quais dados de telemetria estão disponíveis?**
    - A API fornece dados detalhados como velocidade, RPM, temperatura do motor, nível de combustível, estado das portas, uso do cinto de segurança, entre outros.
11. **O que fazer quando um veículo não aparece no mapa?**
    - Verifique se o dispositivo está ligado, se o veículo tem permissão associada ao seu usuário e consulte a última transmissão para determinar se há problemas de conectividade.

12. **Como resolver erros de sincronização no aplicativo do motorista?**
    - Verifique a versão do aplicativo, a conexão de internet do dispositivo e tente limpar o cache. Se persistir, reinstale a aplicação.
13. **Por que o roteirizador não está considerando todas as restrições configuradas?**
    - Verifique se não há conflitos entre restrições e se todas estão corretamente configuradas. Algumas restrições podem ter precedência sobre outras.
14. **Como posso saber se um comando foi recebido pelo veículo?**
    - Os comandos enviados são registrados no sistema e você pode verificar o status através do histórico de comandos na interface web ou consultando os logs do sistema.

11. Exemplos de Implementação

11.1. Exemplo de Rastreamento em Tempo Real

```
// Exemplo de código JavaScript para rastreamento em tempo real
async function monitorarVeiculosEmTempoReal() {
    // 1. Autenticar no sistema
    const credenciais = {
        login: "empresa@exemplo.com.br",
        senha: "senhaSegura123",
        tipoAcesso: "grupo"
    };

    const respAuth = await fetch('http://rastreamento.concept.inf.br/GPSCorporativo/rest/autenticar', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/x-www-form-urlencoded'
        },
        body: new URLSearchParams(credenciais)
    });

    const authData = await respAuth.json();
    const jsessionId = authData.jsesionId;

    // 2. Configurar intervalo para atualização a cada 30 segundos
```

```
    setInterval(async () => {
        // 3. Obter posição atual dos veículos
        const respVeiculos = await fetch('http://rastreamento.concept.inf.br/GPSCorporativo/rest/rastrearVeiculos',
{
            method: 'GET',
            headers: {
                'Cookie': `JSESSIONID=${jsessionId}`
            }
        });

        const veiculos = await respVeiculos.json();

        // 4. Processar e exibir os dados
        veiculos.forEach(veiculo => {
            console.log(`Veículo ${veiculo.placa} - Posição: ${veiculo.latitude},${veiculo.longitude}`);
            console.log(`Velocidade: ${veiculo.velocidadeAtual} km/h - Status: ${veiculo.ignicao}`);

            // Aqui você adicionaria código para atualizar um mapa ou interface
        });
    }, 30000); // 30 segundos
}

monitorarVeiculosEmTempoReal();
```

11.2. Exemplo de Análise Histórica

```
// Exemplo de código JavaScript para análise histórica de percurso
async function analisarPercursoHistorico(idVeiculo, dataInicio, dataFim) {
    // 1. Autenticar no sistema
    const credenciais = {
        login: "empresa@exemplo.com.br",
        senha: "senhaSegura123",
        tipoAcesso: "grupo"
    };

    const respAuth = await fetch('http://rastreamento.concept.inf.br/GPSCorporativo/rest/autenticar', {
        method: 'POST',
        headers: {
```

```
            'Content-Type': 'application/x-www-form-urlencoded'
        },
        body: new URLSearchParams(credenciais)
    });

    const authData = await respAuth.json();
    const jsessionId = authData.jsesionId;

    // 2. Consultar histórico de percurso
    const params = {
        idVeiculo: idVeiculo,
        dataHoraInicio: dataInicio, // Formato: YYYYMMDDHHmmss
        dataHoraFim: dataFim,       // Formato: YYYYMMDDHHmmss
        somenteParadas: "false"
    };

    const respHistorico = await
fetch('http://rastreamento.concept.inf.br/GPSCorporativo/rest/rastrearVeiculoPorData', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/x-www-form-urlencoded',
            'Cookie': `JSESSIONID=${jsessionId}`
        },
        body: new URLSearchParams(params)
    });

    const historico = await respHistorico.json();

    // 3. Processar os dados históricos
    let distanciaTotal = 0;
    let tempoMovimento = 0;
    let tempoParado = 0;

    for (let i = 1; i < historico.length; i++) {
        const pontoAtual = historico[i];
        const pontoAnterior = historico[i-1];

        // Calcular distância entre pontos (exemplo simplificado)
        const distancia = calcularDistanciaEntrePontos(
```

```
            pontoAnterior.latitude, pontoAnterior.longitude,
            pontoAtual.latitude, pontoAtual.longitude
        );

        distanciaTotal += distancia;

        // Analisar tempos de movimento e parada
        if (pontoAtual.velocidadeAtual > 0) {
            tempoMovimento++;
        } else {
            tempoParado++;
        }
    }

    // 4. Retornar análise
    return {
        idVeiculo: idVeiculo,
        periodo: {
            inicio: dataInicio,
            fim: dataFim
        },
        pontos: historico.length,
        distanciaTotal: distanciaTotal.toFixed(2) + " km",
        tempoMovimento: formatarTempo(tempoMovimento),
        tempoParado: formatarTempo(tempoParado),
        velocidadeMedia: (distanciaTotal / tempoMovimento * 3600).toFixed(2) + " km/h"
    };
}

// Funções auxiliares
function calcularDistanciaEntrePontos(lat1, lon1, lat2, lon2) {
    // Implementação da fórmula de Haversine para cálculo de distância
    // entre coordenadas geográficas
    // Código simplificado para o exemplo
    return 0.5; // km
}

function formatarTempo(minutos) {
    const horas = Math.floor(minutos / 60);
```

```
    const min = minutos % 60;
    return `${horas}h ${min}min`;
}

// Exemplo de uso
analisarPercursoHistorico(
    "3375",
    "20250310000000", // 10/03/2025 00:00:00
    "20250310235959"  // 10/03/2025 23:59:59
).then(analise => {
    console.log("Análise do percurso histórico:", analise);
});
```

12. Referências Cruzadas

Componente	Documentação Relacionada	Seção Relevante
Sistema Web	Manual do Sistema Web	Capítulo 4: Monitoramento de Frota
Aplicativo do Motorista	Guia do Aplicativo	Seção 2.3: Comunicação com a Central
Roteirizador	Manual do Roteirizador	Capítulo 5: Integração com Rastreamento
API de Integração	Documentação de Integração	Seção 3.1: Autenticação e Segurança

Vínculos entre Componentes e Funcionalidades

- **Monitoramento em Tempo Real**: Para implementações avançadas de monitoramento em tempo real, consulte a seção 5.2 deste documento e a seção 4.2 do Manual do Sistema Web.
- **Roteirização com Rastreamento**: Para otimizar rotas com base em dados históricos, consulte a seção 5.3 deste documento e o Capítulo 3 do Manual do Roteirizador.
- **Comunicação com Motoristas**: Para configurar mensagens e alertas automáticos, consulte as seções 5.8 e 5.9 deste documento e a Seção 2.3 do Guia do Aplicativo.
- **Telemetria e Manutenção Preventiva**: Para implementar manutenção baseada em dados de telemetria, consulte a seção 5.5 deste documento e o Capítulo 6 do Manual do Sistema Web.

13. Representações Estruturadas das Entidades

13.1. Estrutura de Veículo (JSON)


```
{
  "id": 3375,
  "placa": "ABC1234",
  "tipo": "CAMINHAOA",
  "modelo": "F-4000",
  "ano": 2022,
  "chassi": "9BFFT35P3CD123456",
  "capacidade": {
    "peso": 3500,
    "volume": 24,
    "passageiros": 3
  },
  "dispositivo": {
    "id": 98765,
    "modelo": "GP1500",
    "imei": "123456789012345",
    "dataInstalacao": "20230215"
  },
  "telemetria": {
    "sensores": [
      "rpm",
      "temperatura",
      "combustivel",
      "pressao_oleo",
      "cinto",
      "portas"
    ],
    "intervaloPadrao": 30,
    "intervaloParado": 300
  },
  "limites": {
```

```
    "velocidadeMaxima": 80,
    "rpmMaximo": 2500,
    "horasOperacao": 12
  }
}
```


13.2. Estrutura de Motorista (JSON)


```
{
  "id": 1234,
  "nome": "João da Silva",
  "cpf": "123.456.789-00",
  "matricula": "M1234",
  "cnh": {
    "numero": "12345678900",
    "categoria": "D",
    "validade": "20260531"
  },
  "contato": {
    "telefone": "11999998888",
    "email": "joao.silva@email.com"
  },
  "acessoApp": {
    "login": "joao.silva",
    "ultimoAcesso": "20250311103045",
    "versaoApp": "2.3.7"
  },
  "veiculoAtual": 3375,
  "status": "EmRota"
}
```


13.3. Estrutura de Rota (JSON)


```
{
  "id": 5678,
  "nome": "Entrega Centro - 12/03/2025",
  "veiculo": 3375,
  "motorista": 1234,
```

```
"status": "EmAndamento",
"criacao": "20250311153045",
"inicio": {
  "planejado": "20250312080000",
  "real": "20250312075532"
},
"fim": {
  "planejado": "20250312180000",
  "real": null
},
"paradas": [
  {
    "ordem": 1,
    "cliente": "Empresa A",
    "endereco": "Av. Amazonas, 1234",
    "coordenadas": {
      "latitude": -19.9322,
      "longitude": -43.9352
    },
    "horario": {
      "planejado": "20250312090000",
      "previsto": "20250312092500",
      "real": "20250312093012"
    },
    "status": "Concluido"
  },
  {
    "ordem": 2,
    "cliente": "Empresa B",
    "endereco": "Rua dos Carijós, 456",
    "coordenadas": {
      "latitude": -19.9182,
      "longitude": -43.9372
    },
    "horario": {
      "planejado": "20250312110000",
      "previsto": "20250312114500",
      "real": null
    },
    "status": "EmAndamento"
  }
```

```
  ],
  "metricas": {
    "distanciaPlanejada": 28.5,
    "distanciaPercorrida": 15.7,
    "consumoPlanejado": 8.5,
    "consumoReal": 9.2
  }
}
```

Revision #11
Created 22 March 2025 13:34:30 by Moises Reis Filho
Updated 16 May 2026 13:48:20 by Moises Reis Filho