**Documentação Técnica**

**Sistema de Previsão de Eventos Logísticos com NeuralProphet**

**Índice**

1. Introdução
2. Arquitetura do Sistema  
   2.1 Visão Geral  
   2.2 Diagrama de Arquitetura
3. Componentes  
   3.1 Backend  
   3.2 Frontend
4. Fluxo de Funcionamento
5. Detalhamento Técnico  
   5.1 Tratamento dos Dados  
   5.2 Treinamento do Modelo  
   5.3 API REST – Detalhes dos Endpoints  
   5.4 Consumo pelo Frontend
6. Deploy e Atualização  
   6.1 Backend (Render Web Service)  
   6.2 Frontend (Site Estático)  
   6.3 Atualização de Componentes
7. Segurança e Boas Práticas
8. Exemplo de Uso
9. Roadmap e Melhorias Futuras
10. FAQ
11. Créditos e Licenciamento

**1. Introdução**

Esta documentação detalha o sistema de previsão de eventos logísticos baseado em modelos de séries temporais com NeuralProphet, disponibilizado através de uma API RESTful em Python/Flask e interface web frontend.

O objetivo do sistema é antecipar, para uma data escolhida, o número de eventos operacionais, além de fornecer rankings detalhados por motorista e tipo de evento. O sistema visa apoiar a tomada de decisão e estratégias de redução de riscos.

**2. Arquitetura do Sistema**

**2.1 Visão Geral**

O sistema segue arquitetura web desacoplada, com backend e frontend em deploys independentes e comunicação via HTTP/REST.

**2.2 Diagrama de Arquitetura**

[Usuário] <-- navegador --> [Frontend HTML/CSS/JS]

|

| (fetch)

v

[API REST Flask + NeuralProphet] <--> [Arquivo CSV de Dados]

|

[Hospedado na Render]

**3. Componentes**

**3.1 Backend**

**Tecnologias Utilizadas:**

* Linguagem: Python 3.11
* Principais bibliotecas: Flask, Gunicorn, NeuralProphet, pandas, numpy, torch
* Hospedagem: Render.com (Web Service)
* Serialização: torch.serialization

**Estrutura e Funcionalidades:**

* Carregamento, tratamento e agregação de dados via pandas.
* Treinamento do modelo de séries temporais NeuralProphet ao iniciar o backend.
* Exposição do endpoint /predict, que recebe uma data por GET e retorna um objeto JSON com as previsões e análises.
* Preenchimento automático de valores ausentes; tratamento robusto de erros e inconsistências.
* O backend está pronto para escalonamento e aceita conexões públicas somente na rota /predict.

**Principais Endpoints:**

* GET /predict?date=YYYY-MM-DD
  + Parâmetro: date (opcional, padrão = amanhã)
  + Retorno: JSON contendo:
    - data\_previsao (string, data consultada)
    - previsao\_total\_yhat1 (float, quantidade prevista de eventos)
    - top10\_motoristas\_geral (array de objetos, ranking dos 10 motoristas)
    - probabilidade\_eventos\_especificos (objeto, detalhamento por tipo de evento)
  + Possíveis status HTTP: 200, 404, 500
* Rota / não utilizada (Retornará 404 se chamada)

**Segurança:**

* CORS habilitado para garantir integração com frontend hospedado separadamente.
* API expõe apenas o necessário, retornando códigos de status apropriados para erros e exceções.

**3.2 Frontend**

**Tecnologias Utilizadas:**

* HTML5, CSS3, JS (ES6+)
* Hospedagem: Render Static Site, Netlify, Vercel ou GitHub Pages

**Estrutura e Funcionalidades:**

* Permite ao usuário escolher uma data de previsão.
* Exibe o resultado da previsão geral, o ranking dos 10 motoristas mais propensos e os 5 motoristas mais associados a cada tipo específico de evento.
* Exibe mensagens amigáveis durante o carregamento ou em caso de erros.
* Paleta de cores institucional da Ritmo Logística.

**4. Fluxo de Funcionamento**

1. Backend inicializa, lê CSV, executa tratamento e treina modelo.
2. Usuário acessa frontend, escolhe a data de previsão.
3. Frontend envia requisição para /predict?date=YYYY-MM-DD.
4. Backend processa, executa previsão e retorna JSON.
5. Frontend exibe resultados de forma clara e organizada.

**5. Detalhamento Técnico**

**5.1 Tratamento dos Dados**

* Leitura do arquivo CSV basedadosseguranca.csv.
* Preenchimento de valores ausentes por moda (categóricos) e média (numéricos).
* Conversão de datas para pandas datetime e agregação diária.
* Dados agregados por dia para treinar o modelo NeuralProphet.

**5.2 Treinamento do Modelo**

* Treinamento do NeuralProphet para ajuste dos eventos ao longo do tempo.
* Frequência diária (freq='D').
* O modelo é carregado e treinado automaticamente ao iniciar o backend.

**5.3 API REST – Detalhes dos Endpoints**

**/predict**

* Request:  
  GET /predict?date=YYYY-MM-DD
* Response:

jsonCopiar

{

"data\_previsao": "2025-06-15",

"previsao\_total\_yhat1": 19.12,

"top10\_motoristas\_geral": [

{"Motorista": "João", "Probabilidade": 3.01},

...

],

"probabilidade\_eventos\_especificos": {

"Excesso de Velocidade": [

{"Motorista": "Carlos", "Probabilidade": 0.25},

...

],

...

}

}

* Erros:
  + Data fora do intervalo: retorna mensagem + intervalo (404)
  + Falha no modelo: erro detalhado (500)

**5.4 Consumo pelo Frontend**

* O JS do frontend utiliza fetch para solicitar os dados, exibindo resultados ou mensagens de erro conforme retorno.

**6. Deploy e Atualização**

**6.1 Backend (Render Web Service)**

* Deploy contínuo via push no GitHub.
* Atualizações em Python ou dependências ativam deploy automático no Render.

**6.2 Frontend (Site Estático)**

* Deploy contínuo: ao atualizar arquivos HTML, CSS ou JS e fazer push, uma nova versão é publicada.

**6.3 Atualização de Componentes**

* Visual (CSS/HTML/JS): basta atualizar e subir no Git.
* Lógica (Python/modelo): atualizar, versionar e re-deploy.

**7. Segurança e Boas Práticas**

* Apenas routing essencial exposto na API.
* CORS habilitado restritamente.
* Tratamento explícito de erros.
* Recomenda-se repositórios privados para dados sensíveis e uso futuro de autenticação.

**8. Exemplo de Uso**

1. Usuário acessa a URL do frontend.
2. Seleciona uma data.
3. Obtém previsão, ranking dos motoristas e análise detalhada de eventos.

**9. Roadmap e Melhorias Futuras**

* Autenticação em endpoints sensíveis.
* Dashboards analíticos interativos.
* Upload de CSV e retreinamento via interface.
* Alertas automáticos para casos críticos.

**10. FAQ**

* **Preciso reiniciar o backend para novo modelo/dados?**  
  Sim, recomendo redeploy ao trocar CSV ou alterar modelo.
* **Suporta outras fontes de dados?**  
  Sim, pode adaptar a rotina de ingestão.
* **Posso personalizar frontend?**  
  Sim, altere os arquivos e faça novo deploy.

**11. Créditos e Licenciamento**

* **Autor:** Seu Nome
* **Contato:** [Seu Email ou GitHub]
* **Licenciamento:** [Informe aqui, ex. MIT]
* **Frameworks:** NeuralProphet, pandas, Flask, etc.