Introdução

Nome do Projeto

iteraBus – Rastreamento de Ônibus em Tempo Real

Descrição

O iteraBus é um sistema de rastreamento de ônibus em tempo real, desenvolvido para fornecer informações precisas sobre a localização dos veículos, ajudando passageiros e operadores de transporte a planejarem melhor seus deslocamentos. O sistema combina um backend robusto em .NET Core com um frontend interativo em React, além da integração com a API do Google Maps para exibição das rotas.

Qual problema resolve?

Você já precisou pegar um ônibus, chegou ao ponto e se perguntou: será que meu ônibus já passou por aqui?

Ou se perguntou: será que ele realmente passa por esse ponto?

Essa incerteza é muito comum entre os passageiros. A falta de informações claras sobre a localização dos ônibus dificulta o planejamento de deslocamentos e gera frustração. Por isso nasceu o iteraBus, que resolve essa questão fornecendo dados em tempo real, permitindo que os usuários saibam exatamente onde o ônibus está e quanto tempo levará para chegar ao ponto desejado.

Motivação para o projeto

No dia a dia das cidades, o transporte público é essencial para a mobilidade das pessoas. No entanto, um dos maiores desafios enfrentados pelos passageiros é a falta de previsibilidade sobre a chegada dos ônibus.

A falta de informações em tempo real causa atrasos, incertezas e dificulta o planejamento da rotina dos usuários. O **iteraBus** surge para solucionar esse problema, oferecendo uma plataforma que permite aos passageiros **rastrear ônibus em tempo real**, visualizar rotas atualizadas e planejar melhor seus deslocamentos.

Tecnologias Utilizadas

- **Linguagens de Programação:** C#, JavaScript
- **Frameworks e Bibliotecas:** .NET Core, React
- **Banco de Dados:** SQL Server, utilizando Entity Framework Core para ORM
- **APIs Externas:** Google Maps API para exibição do mapa e rastreamento de veículos
- **Ferramentas de Desenvolvimento:** Visual Studio Code, Swagger, GitHub
- **Arquitetura:** Aplicação baseada em arquitetura de camadas (Domínio, Aplicação, Infraestrutura e API)

Requisitos do Sistema

1. **Cadastro e Gestão de Ônibus**
    - Permitir o cadastro, edição e remoção de ônibus.
    - Associar cada ônibus a uma rota específica.
2. **Gerenciamento de Rotas**
    - Criar, editar e excluir rotas de ônibus.
    - Associar pontos de parada às rotas.
3. **Rastreamento de Ônibus em Tempo Real**
    - Exibir a posição dos ônibus no mapa utilizando a API do Google Maps.
    - Atualizar a posição dos ônibus dinamicamente.
4. **Consulta de Informações pelos Passageiros**
    - Permitir que os passageiros visualizem no mapa a posição atual dos ônibus.
    - Exibir previsão de chegada dos ônibus aos pontos de parada.
5. **Autenticação e Controle de Acesso**
    - Permitir login e cadastro de usuários.
    - Diferenciar níveis de acesso (passageiro e administrador)

Histórias de Usuário

1. **Gerenciamento de Ônibus**
    - Como administrador, eu quero cadastrar um novo ônibus para que ele possa ser rastreado no sistema.
    - Como administrador, eu quero editar as informações de um ônibus para que eu possa manter os dados sempre atualizados.
    - Como administrador, eu quero remover um ônibus do sistema para que eu possa manter apenas informações relevantes.
2. **Gestão de Rotas**
    - Como administrador, eu quero criar uma nova rota para que os ônibus possam ser associados a um trajeto específico.
    - Como administrador, eu quero editar os detalhes de uma rota para que as informações reflitam mudanças no trajeto.
    - Como administrador, eu quero excluir uma rota do sistema para que somente rotas ativas fiquem disponíveis.
3. **Rastreamento e Visualização no Mapa**
    - Como passageiro, eu quero ver no mapa a localização dos ônibus em tempo real para que eu saiba quando ele passará pelo meu ponto.
    - Como passageiro, eu quero visualizar os detalhes da rota de um ônibus para que eu saiba quais pontos ele atende.
    - Como passageiro, eu quero receber uma previsão de chegada do ônibus ao meu ponto para que eu possa planejar meu deslocamento.
4. **Autenticação e Controle de Acesso**
    - Como usuário, eu quero fazer login no sistema para que eu tenha acesso às funcionalidades personalizadas.
    - Como administrador, eu quero cadastrar novos usuários e definir permissões para que somente pessoas autorizadas possam gerenciar ônibus e rotas.

Plano de Desenvolvimento

Fase 1 - Estruturação do Projeto (2 dias)

Objetivo: Definir a base do projeto, garantindo organização e escalabilidade.

1.1 Criar repositório no GitHub

1.2 Configurar ambiente de desenvolvimento

1.3 Definir arquitetura do backend (camadas: domínio, aplicação, API)

1.4 Criar banco de dados e definir entidades iniciais (Ônibus, Localização, Rota)

Fase 2 - Desenvolvimento Backend (1 semana)

Objetivo: Criar APIs para manipulação dos dados e garantir a comunicação entre o frontend e o banco de dados.

2.1 Criar APIs de CRUD para Ônibus, Rota e Localização

2.2 Implementar serviços e repositórios (Interface x Implementação)

2.3 Configurar Entity Framework Core com SQL Server

2.4 Testes e correções

Fase 3 - Desenvolvimento Frontend (2 semanas)

Objetivo: Criar a interface do usuário e garantir uma experiência fluida ao visualizar os dados de rastreamento.

3.1 Criar estrutura do projeto React

3.2 Implementar telas básicas (Login, Dashboard, Mapa)

3.3 Consumir API do Google Maps para exibir o mapa

3.4 Consumir API do backend

3.5 Exibir ônibus em tempo real no Google Maps com base nas coordenadas enviadas pelo backend

Fase 4 - Integração e Testes (1 semana)

Objetivo: Garantir que todas as partes do sistema funcionem corretamente e de forma segura.

4.1 Testes de integração entre backend e frontend

4.2 Ajustes de segurança na API

4.3 Melhorias no fluxo de atualização do mapa

4.4 Simulação de uso com múltiplos usuários para validação de performance

Fase 5 - Implementação de Inteligência Artificial com Ollama (Opcional - 2 a 4 semanas)

Objetivo: Utilizar IA para análise preditiva e otimização do rastreamento dos ônibus.

5.1 Criar modelo preditivo para estimar tempos de chegada usando Ollama

5.2 Implementar detecção de padrões de tráfego para otimizar rotas

5.3 Integrar o modelo ao backend para fornecer previsões em tempo real

5.4 Testar e validar a precisão dos modelos com dados históricos

Fase Final - Deploy e Entrega

Objetivo: Disponibilizar o sistema em um ambiente de produção e documentar o projeto para futuras melhorias.

6.1 Configurar ambiente de produção

6.2 Criar documentação detalhada do projeto

6.3 Publicar backend e frontend no GitHub

6.4 Realizar primeira apresentação

6.5 Realizar apresentação final com base no feedback da primeira apresentação

Conclusão e Próximos Passos

O **iteraBus** está planejado de forma a garantir um desenvolvimento eficiente e escalável. O projeto inclui a integração com o **Google Maps** para rastreamento visual dos ônibus e o uso de **inteligência artificial (IA)** com **Ollama** para otimização de rotas e estimativas de chegada. A fase de testes garantirá a estabilidade do sistema antes do **deploy** final.

Melhorias futuras incluem:

- Escalabilidade e performance (como cache e indexação no banco).
- Segurança (autenticação e proteção contra ataques).
- Funcionalidades avançadas como notificações em tempo real e mais IA.
- Melhorias na interface de usuário, incluindo um **app móvel**.

O cronograma está bem definido, com etapas claras para a integração com o Google Maps, implementação da IA e testes de integração, visando um **deploy** eficiente e seguro.