

# **DOCUMENTO TESTE PROVISÓRIO EM MODELO ABNT.**

## **Palavras-chave:**

Mobilidade Urbana, Sistema de Transporte, Banco de Dados Relacional, Banco de Dados NoSQL, Otimização de Viagens, Controle de Acesso.

## **1. Introdução**

Com o crescimento das cidades e a consequente sobrecarga dos sistemas de transporte público e privado, a mobilidade urbana tornou-se um dos principais desafios das grandes metrópoles. A busca por soluções eficientes para otimizar o tráfego e melhorar a experiência do usuário é um tema de interesse crescente. Nesse contexto, este artigo apresenta o desenvolvimento de um aplicativo de mobilidade urbana que utiliza uma combinação de tecnologias avançadas para otimizar as viagens dos usuários e melhorar a gestão do tráfego em tempo real. A solução proposta visa integrar diferentes fontes de dados e fornecer uma interface intuitiva, personalizada e escalável.

## **2. Objetivos**

O principal objetivo deste projeto é desenvolver um sistema de mobilidade urbana que permita aos usuários escolherem as rotas mais rápidas e eficientes com base em dados em tempo real, como tráfego e condições meteorológicas. Além disso, busca-se:

- Criar uma plataforma intuitiva e acessível para diferentes perfis de usuários.
- Integrar múltiplas fontes de dados (sensores de tráfego, dados climáticos, etc.).
- Utilizar tecnologia de ponta para garantir a escalabilidade e o desempenho do sistema.
- Implementar um controle de acesso robusto para segurança e personalização dos serviços.

- Utilizar bancos de dados relacionais e NoSQL de maneira estratégica para atender diferentes necessidades do sistema.

### 3. Metodologia

A metodologia utilizada no desenvolvimento do sistema é baseada em uma abordagem ágil, com ciclos de desenvolvimento rápidos e entregas incrementais. As fases incluem:

1. **Análise de requisitos:** Levantamento das necessidades dos usuários e especificações técnicas do sistema.
2. **Desenvolvimento:** Implementação das funcionalidades principais do aplicativo, incluindo integração com APIs externas (tráfego, clima, etc.).
3. **Testes e Validação:** Validação da usabilidade e performance do sistema, além de testes de segurança.
4. **Implantação e monitoramento:** Lançamento da versão inicial e acompanhamento do desempenho do sistema em tempo real.

### 4. Descrição do Sistema

#### 4.1 Funcionalidades

O sistema oferece diversas funcionalidades, com o objetivo de proporcionar uma experiência de usuário otimizada e personalizada:

- **Pesquisa de rotas:** Permite ao usuário pesquisar rotas baseadas em diferentes critérios (tempo, distância, custo).
- **Informações em tempo real:** O sistema atualiza informações de tráfego, clima e outros dados relevantes em tempo real.

- **Personalização:** O aplicativo aprende as preferências de viagem do usuário, oferecendo sugestões baseadas em seu histórico e preferências.
- **Alertas:** Notificações para mudanças no tráfego, condições climáticas adversas, ou interrupções no serviço de transporte.

#### 4.2 Tecnologias utilizadas

- **Frontend:** React Native para a criação de um aplicativo móvel multiplataforma, garantindo a compatibilidade com iOS e Android.
- **Backend:** Node.js com Express, escolhido por sua escalabilidade e facilidade de integração com APIs externas.
- **Banco de Dados Relacional:** PostgreSQL, utilizado para armazenar dados estruturados, como informações de usuários, rotas e históricos de viagens.
- **Banco de Dados NoSQL:** MongoDB, utilizado para armazenamento de dados não estruturados, como informações temporais de tráfego e localização em tempo real.

#### 4.3 Justificativa de cada escolha tecnológica

A escolha do **React Native** foi motivada pela necessidade de desenvolver um aplicativo que funcione em ambas as plataformas (iOS e Android) sem duplicação significativa de código. O **Node.js** foi escolhido devido à sua arquitetura baseada em eventos, que permite uma alta performance na manipulação de grandes volumes de requisições simultâneas, ideal para um sistema de mobilidade urbana que precisa de atualizações em tempo real. O **PostgreSQL** foi selecionado por ser um banco relacional robusto e altamente escalável, capaz de lidar com dados estruturados de forma eficiente. O **MongoDB** foi escolhido para lidar com dados temporais e não estruturados, como registros de localização e tráfego, devido à sua flexibilidade e escalabilidade.

### 5. Modelagem do Banco de Dados

## 5.1 Diagrama Entidade-Relacionamento (DER)

(UTILIZAR DE TESTE OU REAPROVEITAR O DIAGRAMA ANTIGO)

## 5.2 Explicação das Entidades e Relacionamentos

- **Usuário:** Armazena informações sobre os usuários do sistema (nome, e-mail, preferências de viagem).
- **Rota:** Contém as informações sobre as rotas possíveis, como origem, destino e distância.
- **Viagem:** Representa uma viagem realizada pelo usuário, incluindo dados de tempo, duração, rota utilizada e preferências.
- **Tráfego:** Informações sobre o tráfego em tempo real, relacionadas à localização.
- **Clima:** Informações meteorológicas relevantes para as rotas e viagens.

## 5.3 Justificativa para o uso de índices, triggers, views, procedures/functions

- **Índices:** Serão utilizados índices nas colunas mais consultadas, como origem e destino das rotas, para melhorar a performance das buscas.
  - **Triggers:** Usadas para atualizar o status de uma viagem quando o tráfego ou o clima mudam de forma significativa.
  - **Views:** As views serão usadas para combinar informações de diferentes tabelas (por exemplo, combinação de tráfego e clima).
  - **Procedures/Functions:** Serão criadas funções para cálculo de tempo estimado de viagem, levando em conta diferentes variáveis (tráfego, clima).
- 

## 6. Controle de Acesso

### 6.1 Descrição dos usuários e grupos

- **Usuário Comum:** Usuários que utilizam o sistema para planejar suas viagens.
- **Administrador:** Usuários responsáveis por gerenciar dados do sistema, como rotas e informações de tráfego.
- **Operador de Sistema:** Usuários com acesso a dados em tempo real sobre tráfego e clima, responsáveis por monitorar a situação e fornecer informações aos usuários.

## 6.2 Regras de Acesso

- **Usuário Comum:** Acesso restrito às funcionalidades de pesquisa de rotas e histórico de viagens.
  - **Administrador:** Acesso total ao gerenciamento de dados do sistema.
  - **Operador de Sistema:** Acesso aos dados em tempo real, mas sem a capacidade de modificar dados estruturais do sistema.
- 

## 7. Uso do Banco NoSQL

### 7.1 Explicação técnica do banco escolhido

O **MongoDB** foi escolhido para armazenar dados não estruturados, como a localização dos usuários e informações de tráfego em tempo real. Sua arquitetura de documentos JSON permite uma fácil escalabilidade e flexibilidade na manipulação de dados dinâmicos.

### 7.2 Justificativa da aplicação no sistema

MongoDB é ideal para lidar com dados temporais e em grande volume, como os dados de localização e o tráfego em tempo real. Sua natureza NoSQL oferece flexibilidade para armazenar e consultar esses dados de forma eficiente, sem a necessidade de um esquema fixo, o que facilita a integração de diferentes fontes de dados externas (como APIs de tráfego e clima).

---

## **8. Conclusão**

O desenvolvimento do sistema de mobilidade urbana proposto busca atender à crescente demanda por soluções de transporte mais eficientes nas grandes cidades. A combinação de tecnologias de frontend, backend e bancos de dados relacionais e NoSQL permite que o sistema seja altamente escalável, eficiente e seguro. O controle de acesso e a modelagem do banco de dados garantem que as informações dos usuários sejam bem gerenciadas e protegidas. Com a implementação dessas tecnologias, espera-se que o sistema contribua significativamente para a melhoria da mobilidade urbana e para a otimização das viagens dos usuários.