Relatório Final sobre Java Parkings

Bernardo Melgaço[1], Diogo Henrique[2], Guilherme Lana [3], Gustavo Ignácio[4]
João Almeida[5], Lernardo Amaral[6]

[1]Pontifícia Universidade Católica de Minas Gerais
Belo Horizonte – MG – Brasil

[2]Curso de Engenharia de Software.

Abstract. *The growing scarcity of parking spaces in Brazilian urban centers has proven to be a promising opportunity for innovative companies. In this scenario, Xulambs Inc. plans to launch Xulambs Parking, an integrated system for parking management in large cities. Xulambs Parking will have a network of strategically distributed parking lots, offering a practical and functional solution that prioritizes comfort and ease of use for customers. Furthermore, all data, such as customer records, vehicles and parking space occupancy, will be stored securely in a database, ensuring the efficiency and reliability of the operation.*

Resumo. *A crescente escassez de vagas de estacionamento nos centros urbanos brasileiros tem se mostrado uma oportunidade promissora para empresas inovadoras. Nesse cenário, a Xulambs Inc. planeja lançar o Xulambs Parking, um sistema integrado para a gestão de estacionamentos em grandes cidades. O Xulambs Parking contará com uma rede de estacionamentos distribuídos estrategicamente, oferecendo uma solução prática e funcional que prioriza o conforto e a facilidade de uso para os clientes. Além disso, todos os dados, como registros de clientes, veículos e ocupação de vagas, serão armazenados de forma segura em um banco de dados, garantindo a eficiência e confiabilidade da operação.*

1. Visão Geral das Principais Funcionalidades

O sistema Xulambs Parking foi desenvolvido para oferecer uma solução integrada e eficiente para a gestão de estacionamentos em grandes cidades, atendendo aos requisitos propostos em cada sprint de desenvolvimento. Abaixo, estão descritas as principais funcionalidades:

1.1. Gestão de Estacionamentos

Cadastro e gerenciamento de múltiplos estacionamentos, cada um com um número fixo de vagas identificadas alfanumericamente (ex.: vaga Y08). Classificação das vagas em categorias específicas: Regular, Idoso, PCD e VIP, cada uma com suas regras e características particulares, como descontos ou tarifas diferenciadas.

1.2. Controle de Vagas e Veículos

Registro de veículos por placa, vinculados a clientes identificados ou anônimos. Garantia de exclusividade de uso de uma vaga por cliente, evitando conflitos. Permissão para que um cliente tenha múltiplos veículos cadastrados.

1.3. Histórico e Relatórios de Uso

Consulta de histórico de uso do estacionamento para clientes identificados, com opções de filtro por intervalo de datas. Relatórios gerenciais para os administradores, incluindo: Valor total arrecadado no estacionamento. Arrecadação por mês específico. Valor médio por utilização. Ranking de clientes com maior contribuição para a receita mensal.

2. Principais consultas SQL

As principais consultas SQL realizadas pelo nosso sistema são: **SalvarEstacionamento**: Salva um novo estacionamento no banco de dados.

ListarEstacionamentos: Lista todos os estacionamentos cadastrados.

RemoverEstacionamentoPorId: Remove um estacionamento e suas dependências, como as vagas associadas, usando o ID do estacionamento.

SalvarVagas: Salva uma lista de vagas em um estacionamento específico.

BuscarEstacionamentoPorId: Busca um estacionamento pelo seu ID.

ListarVagasPorEstacionamento: Lista todas as vagas de um estacionamento específico.

BuscarRankingEstacionamentos: Retorna o ranking de estacionamentos baseado no total faturado por cada um.

ListarRankingUtilizacao: Retorna o ranking dos estacionamentos com base na quantidade de utilizações.

CalcularArrecadacaoTotal: Calcula o total arrecadado em um estacionamento específico.

CalcularValorMedioUtilizacao: Calcula o valor médio das utilizações de um estacionamento.

CalcularArrecadacaoMensal: Calcula o total arrecadado mensalmente em um estacionamento.

SalvarTicket: Salva um novo ticket no banco de dados, registrando a entrada de um veículo.

BuscarTicketsAtivos: Busca todos os tickets ativos (sem saída registrada) de um estacionamento.

BuscarVeiculoPorVaga: Busca a placa do veículo que está em uma vaga específica.

RegistrarSaida: Registra a saída de um veículo de uma vaga e calcula o custo do ticket.

BuscarTicketsFechados: Busca todos os tickets fechados (com saída registrada) de um estacionamento.

BuscarTicketPorVaga: Busca o ticket associado a uma vaga específica.

TemTicketAberto: Verifica se há algum ticket aberto (sem saída registrada) para um veículo.

AtualizarStatusVaga: Atualiza o status de ocupação de uma vaga, definindo se está ocupada ou disponível.

3. Diagrama de Classes Final

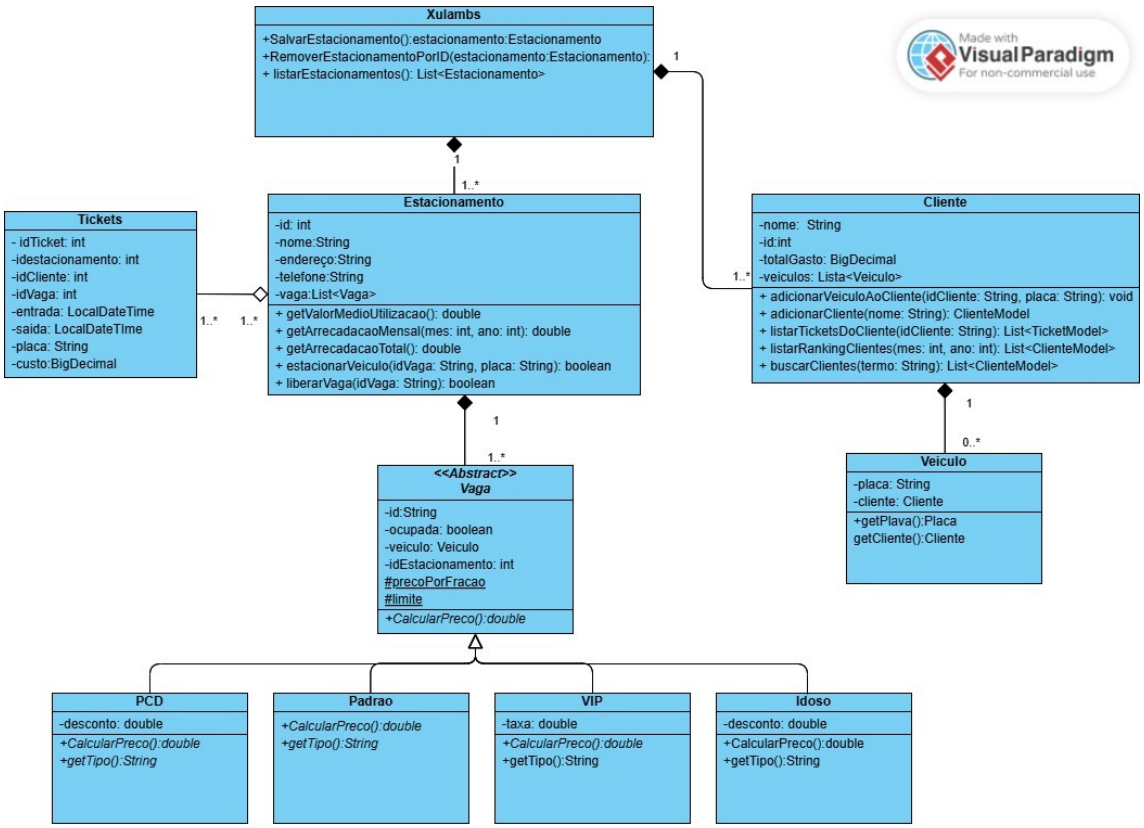

Figure 1. O diagrama de classes representa a estrutura do sistema para gerenciar um estacionamento, destacando as principais entidades: Estacionamento, Vaga, Cliente, Veículo e Ticket. Ele detalha as relações entre essas classes, bem como os atributos e métodos necessários para o funcionamento do sistema. Permite visualizar como cada entidade interage, garantindo a gestão eficiente de vagas, clientes e cobranças.

4. Diagrama Entidade-Relacionamento (DER)

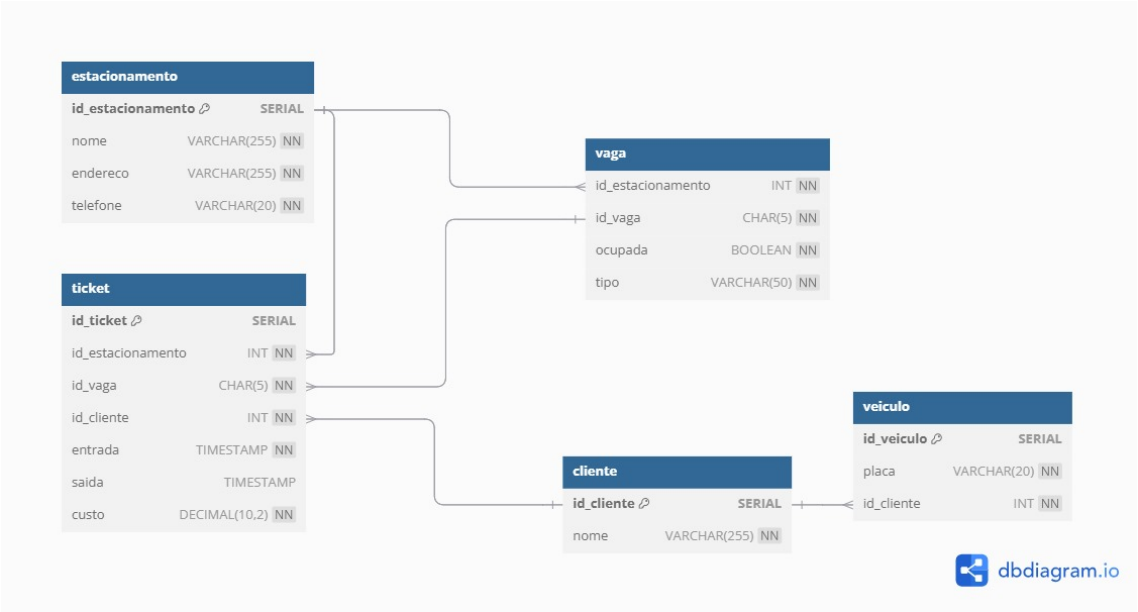

Figure 2. O diagrama Entidade-Relacionamento (DER) modela o banco de dados usado pelo sistema. Ele organiza dados em tabelas como Estacionamento, Vaga, Cliente, Veículo e Ticket. O DER define relacionamentos claros, como a associação entre clientes e veículos, e entre tickets e vagas, garantindo integridade e consistência nos dados. Chaves primárias e estrangeiras são utilizadas para estruturar consultas eficientes, permitindo rastrear ocupação de vagas, histórico de uso e receitas.

5. Decisões do Projeto

Escolhemos o PostgreSQL como SGBD devido à experiência adquirida no trabalho interdisciplinar e à sua solidez em operações transacionais. As coleções da API Java, como ArrayList, foram empregadas para a manipulação dinâmica de dados em memória, possibilitando ordenações, buscas e filtros de forma eficiente. Além disso, a adoção de exceções customizadas foi uma escolha estratégica para aprimorar o gerenciamento de erros e a transparência no manejo de falhas. Exceções como ClienteDAOException e EstacionamentoDAOException foram criadas para envolver erros específicos ligados a operações de banco de dados e normas de negócio. Isso garante que as questões sejam reconhecidas e resolvidas de modo mais direto e exato, prevenindo a disseminação de mensagens genéricas ou irrelevantes para o restante do sistema. A estrutura do projeto adota o padrão MVC (Model-View-Controller), favorecendo a modularidade e tornando mais fácil a manutenção. A distinção nítida entre DAOs, controladores e modelos estrutura o código e possibilita um desenvolvimento mais sustentável do sistema. Além disso, procuramos restringir a quantidade de telas, assegurando a simplicidade no fluxo do programa e tornando a manutenção do código mais fácil.

6. Relatos Pessoais

João Almeida: Particularmente, achei o processo bastante trabalhoso. Não que isso seja algo ruim, mas exigiu que eu me dedicasse com mais foco e reservasse um tempo maior

para este trabalho. Apesar disso, consegui aprender muito ao longo do caminho e alinhei meus objetivos com os do grupo. Ao final, espero atingir as expectativas e contribuir positivamente para o resultado.

Diogo Henrique: Participar do desenvolvimento deste sistema foi uma experiência enriquecedora, permitindo aprimorar habilidades técnicas. No entanto, o trabalho em equipe foi desafiador, assim como seguir o padrão MVC, que, apesar dos benefícios, exigiu bastante atenção na implementação.

Guilherme Lana:Esse trabalho, embora cheio de desafios, foi uma experiência marcante, oferecendo a chance de aprender novas práticas, demandar dedicação e adquirir aprendizados valiosos, tanto no âmbito pessoal quanto no trabalho em equipe

Leonardo Amaral: A disciplina de Laboratório de Programação Modular foi desafiadora, mas muito proveitosa. Aprendi a usar melhor a linguagem Java, organizando o código de forma mais eficiente e aplicando conceitos como modularidade e reutilização. Cada Sprint exigiu dedicação, mas trouxe resultados claros no entendimento das boas práticas de programação. Foi um aprendizado direto e prático, que também mostrou a importância de trabalhar de forma colaborativa e focada.

Bernado Melgaço:Embora tenha sido um desafio em diversos aspectos, considero essa experiência extremamente enriquecedora para todos os envolvidos. Participar desse trabalho proporcionou a chance de aprender novas práticas e demandou muita dedicação em várias etapas, permitindo adquirir importantes lições, tanto no âmbito individual quanto no trabalho em equipe.

Gustavo Ignácio: Apesar de ser um trabalho desafiador em diferentes aspectos, acredito que tenha sido uma experiência muito valiosa para todos. Contribuir neste trabalho ofereceu uma oportunidade de aprender novas práticas e exigiu bastante dedicação em vários momentos, ensinando-me muitas coisas importantes, tanto individualmente quanto em relação ao trabalho em equipe.

7. Principais Dificuldades

Nossa principal dificuldade foi fazer o grupo trabalhar em sintonia e lidar com a persistência dos dados utilizando arquivos TXT. Ambas as tarefas foram bastante desafiadoras, mas, ao final, conseguimos superar os obstáculos e alcançar melhorias significativas.

8. Principais Aprendizados

Aprendemos a trabalhar em equipe, a salvar dados tanto em arquivos TXT quanto em um banco de dados, e aprimoramos nossas habilidades na criação de diagramas, compreendendo sua importância no desenvolvimento. Além disso, desenvolvemos nossa capacidade de organizar o código, aplicando o padrão MVC e o padrão Singleton de forma eficiente.

9. Sugestoes de Mudança

Sugere-se melhorar ainda mais a organização do código, reforçando a aplicação do padrão MVC, além de otimizar as consultas no banco de dados para torná-las ainda mais rápidas e eficientes.