Relatório Técnico: Sistema de Gestão de Hoteis - Versão 0.14.18

Disciplina: Técnicas de Linguagens de Programação (T.L.P.)

Curso: Técnico Médio de Informática – ITEL

Ano Letivo: 2024-2025

Líder do Grupo: Adolfo Figueiredo

1. Introdução

Este relatório técnico detalha o desenvolvimento da primeira versão do nosso Sistema de Gestão de Hoteis, elaborado como projeto final da disciplina de Técnicas de Linguagens de Programação (T.L.P.). O sistema foi concebido em Java, utilizando uma interface via terminal, e aplicando os princípios fundamentais da programação orientada a objetos. Nosso principal objetivo foi simular as operações essenciais de um hotel, abrangendo desde o cadastro de hóspedes e usuários (clientes e administradores), reservas, até o check-in e check-out, além de funcionalidades direcionadas tanto para clientes quanto para administradores.

2. Ferramentas Utilizadas

Para o desenvolvimento desta versão do sistema, empregamos as seguintes ferramentas:

- **IDEs**: IntelliJ IDEA e Visual Studio Code
- **Linguagem de Programação**: Java puro (a integração com Spring Boot está planejada para a versão 2)
- **Banco de Dados**: PostgreSQL (totalmente integrado para persistência de dados nesta versão)
- **Controle de Versão**: Git/GitHub (utilização limitada nesta versão por desafios operacionais, mas será reintroduzido na versão 2)

3. Ferramentas Aprendidas

Durante a execução deste projeto, a equipe teve a oportunidade de aprofundar o conhecimento e a aplicação prática de diversas ferramentas e tecnologias, que foram cruciais para o desenvolvimento do Sistema de Gestão de Hotéis. Algumas ferramentas, como **Java**, **VS Code**, **IntelliJ** e conceitos de **Programação Orientada a Objetos (POO)**, já faziam parte do nosso conhecimento prévio, mas foram consolidadas com a aplicação prática no projeto. Além disso, adquirimos novas habilidades com as seguintes ferramentas:

- **Git/GitHub**: Mesmo com desafios iniciais, a prática de controle de versão com Git e o gerenciamento de repositórios no GitHub foram fundamentais para organizar o código e, futuramente, facilitar a colaboração.
- **Docker**: A utilização de Docker para ambientes de desenvolvimento e implantação trouxe uma compreensão valiosa sobre a criação e gerenciamento de contêineres, facilitando a portabilidade e a padronização do ambiente.
- **Banco de Dados (PostgreSQL)**: A integração do PostgreSQL para persistência de dados proporcionou um aprendizado prático e aprofundado sobre modelagem de banco de dados, consultas SQL e gerenciamento de dados em um contexto real de aplicação.

4. Participação dos Integrantes

A colaboração e dedicação de cada membro foram cruciais para o desenvolvimento do projeto. Abaixo, detalhamos a contribuição individual:

- **Adolfo Figueiredo (Líder do Grupo)** - 70% Concluído Coordenou o projeto desde a concepção até a entrega, supervisionando a integração dos módulos para garantir padronização, clareza e funcionalidade. Foi o responsável pela implementação das funcionalidades de login e cadastro, além da integração e gestão do banco de dados.
- **Albertina Samuel** - 60% Concluído Atuou no desenvolvimento da área administrativa do sistema, incluindo a criação de menus de controle interno e funcionalidades como visualização de pedidos e gerenciamento de serviços.
- **Ana Pedro** - 60% Concluído Trabalhou também no desenvolvimento da área administrativa, com foco na lógica de controle da receção e na estrutura de funcionamento interno do sistema.
- **Alessandro da Silva** - 70% Concluído Desenvolveu a interface voltada ao cliente, implementando menus, validações de entrada e mecanismos de feedback. Também foi o responsável pela compilação e organização da documentação.
- **Ângela Maria** - 75% Concluído Implementou a lógica de dados relacionada ao cliente, como o controle de estado individual de cada hóspede e as ações disponíveis para os utilizadores do lado do cliente.

5. Dificuldades Enfrentadas

O processo de desenvolvimento apresentou desafios técnicos e de gestão. As principais dificuldades foram:

1. **Validação Abrangente**: Embora tenhamos implementado validações para garantir entradas de usuário válidas e seguras, algumas ainda necessitam de aprimoramento para cobrir 100% dos casos, o que será abordado na próxima versão.
2. **Gestão de Estado**: Manter a consistência dos dados em tempo de execução, utilizando ArrayLists, exigiu atenção constante, especialmente com alterações simultâneas em objetos interrelacionados (ex: cliente ↔ quarto).
3. **Tratamento de Exceções**: Embora tenhamos iniciado a implementação de blocos try-catch, o tratamento de exceções ainda não foi aplicado de forma completa, o que pode resultar em erros diante de entradas inesperadas. Essa funcionalidade será amplamente melhorada na versão 2.
4. **Controle de Versão**: O uso do GitHub foi descontinuado nesta fase devido a dificuldades técnicas e limitações de tempo. Para a próxima etapa, o projeto será migrado para um repositório com gerenciamento de branches, issues e pull requests.
5. **Colaboração e Distribuição de Tarefas**: Embora a divisão de tarefas tenha sido clara, houve sobrecarga em alguns membros devido a atrasos ou dificuldades técnicas de outros. Isso impactou os prazos de entrega internos.
6. **Aplicação de Frameworks (Spring Boot)**: Tentativas iniciais de integração do Spring Boot revelaram desafios na configuração do ambiente, estruturação de pacotes e compreensão de conceitos como Inversão de Controle (IoC) e Injeção de Dependências. Devido à falta de tempo e à necessidade de maior familiaridade com a ferramenta, sua implementação completa foi adiada para a versão 2, onde pretendemos incorporar persistência em banco de dados, serviços REST e maior modularização.

6. Aumento do Aprendizado

O desenvolvimento desta versão do sistema, apesar dos desafios, foi uma oportunidade valiosa para aprofundar nossos conhecimentos e habilidades práticas. Cada dificuldade enfrentada se tornou uma lição aprendida e um catalisador para o crescimento da equipe:

- **Programação Orientada a Objetos (POO)**: A aplicação dos princípios de POO (encapsulamento, herança e polimorfismo) foi fundamental para a estrutura do sistema, consolidando nosso entendimento sobre como construir software modular e reutilizável.
- **Integração de Banco de Dados**: A experiência com PostgreSQL nos ensinou sobre persistência de dados, modelagem de banco de dados e a importância de transações para manter a integridade das informações.
- **Resolução de Problemas Complexos**: Enfrentar problemas como a validação abrangente e a gestão de estado exigiu raciocínio lógico e a capacidade de encontrar soluções eficazes, mesmo sob pressão.
- **Trabalho em Equipe e Comunicação**: A colaboração no projeto reforçou a importância da comunicação clara, da divisão de tarefas e da ajuda mútua para superar obstáculos e garantir o progresso.
- **Gerenciamento de Projetos e Prazos**: A necessidade de ajustar o escopo e o cronograma diante das dificuldades ensinou sobre a importância do planejamento e da flexibilidade em projetos de software.

- **Visão de Evolução de Software**: A experiência com a versão 0.14.18 nos proporcionou uma visão clara dos próximos passos, como a refatoração com Spring Boot e a implementação de novas funcionalidades, demonstrando a natureza iterativa do desenvolvimento de software.

  Este projeto não apenas nos permitiu aplicar os conhecimentos teóricos adquiridos em T.L.P., mas também nos preparou para desafios mais complexos e para a contínua evolução no campo da programação.

7. Próximos Passos e Funcionalidades Pendentes

A versão atual representa a base do sistema. As seguintes funcionalidades estão planejadas para a versão 2, com um cronograma de implementação definido para após a avaliação final do semestre:

- **UI/UX Aprimorado**: Melhorias significativas na interface do usuário, tornando-a mais intuitiva e amigável.
  - Para o Cliente: Um portal web ou aplicativo móvel para reservas, check-in/check-out, histórico, solicitação de serviços, etc.
  - Para o Administrador: Um painel de controle web ou GUI para gerenciar quartos, hóspedes, funcionários, relatórios, etc., com visualizações claras e interativas.
- **Gestão de Colaboradores**:
  - Perfis de Funcionários: Cadastro de informações dos funcionários.
  - Controle de Acesso: Definição de permissões baseadas no cargo (rececionista, gerente, limpeza).
  - Escalas/Turnos: Um sistema básico para gerenciar os horários de trabalho.
- **Automação de Confirmações**: Envio automático de confirmações de reserva, check-in, check-out por e-mail ou SMS (com integração de API).
- **Notificações em Tempo Real**: Alertar a equipe de limpeza sobre quartos liberados, ou a receção sobre chegadas iminentes.
- **Métodos de Pagamento**: Implementação de opções de pagamento para transações no sistema.
- **Integração com APIs Externas**: Conexão com APIs adicionais para aprimorar o funcionamento e a usabilidade do sistema, potencialmente incluindo serviços de terceiros (ex: geolocalização, serviços de e-mail).
- **Relatórios**: Geração de relatórios detalhados sobre o funcionamento do hotel.
- **Interface Gráfica**: Desenvolvimento de uma interface gráfica (GUI) ou web para aprimorar a experiência do usuário.
- **Refatoração com Spring Boot e Camadas MVC**: Reestruturação do código utilizando o framework Spring Boot e o padrão de arquitetura MVC (Model-View-Controller) para maior escalabilidade e manutenibilidade.

8. Conclusão

Apesar das limitações técnicas e do tempo reduzido, conseguimos atingir o objetivo central desta fase: construir uma aplicação funcional, modular e orientada a objetos, capaz de simular um sistema de reservas hoteleiras. Com a integração do banco de dados PostgreSQL e as funcionalidades de login e cadastro já operacionais, a versão atual demonstra nosso compromisso com boas práticas de programação e com a divisão de tarefas eficaz. Esta experiência reforçou nossas habilidades técnicas e de trabalho em grupo. Estamos plenamente cientes das áreas que necessitam de melhoria e já traçamos os próximos passos para a evolução do sistema na versão 2, visando entregar uma solução ainda mais robusta e completa, com a adição de métodos de pagamento, integração com APIs, melhorias de UI/UX e funcionalidades de gestão de colaboradores.