

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS E INFORMÁTICA
UNIDADE EDUCACIONAL CORAÇÃO EUCARÍSTICO
Bacharelado em Engenharia de Software

Gustavo Azi Prehl Gama - 754737

Guilherme Henrique Moreira Costa - 855826

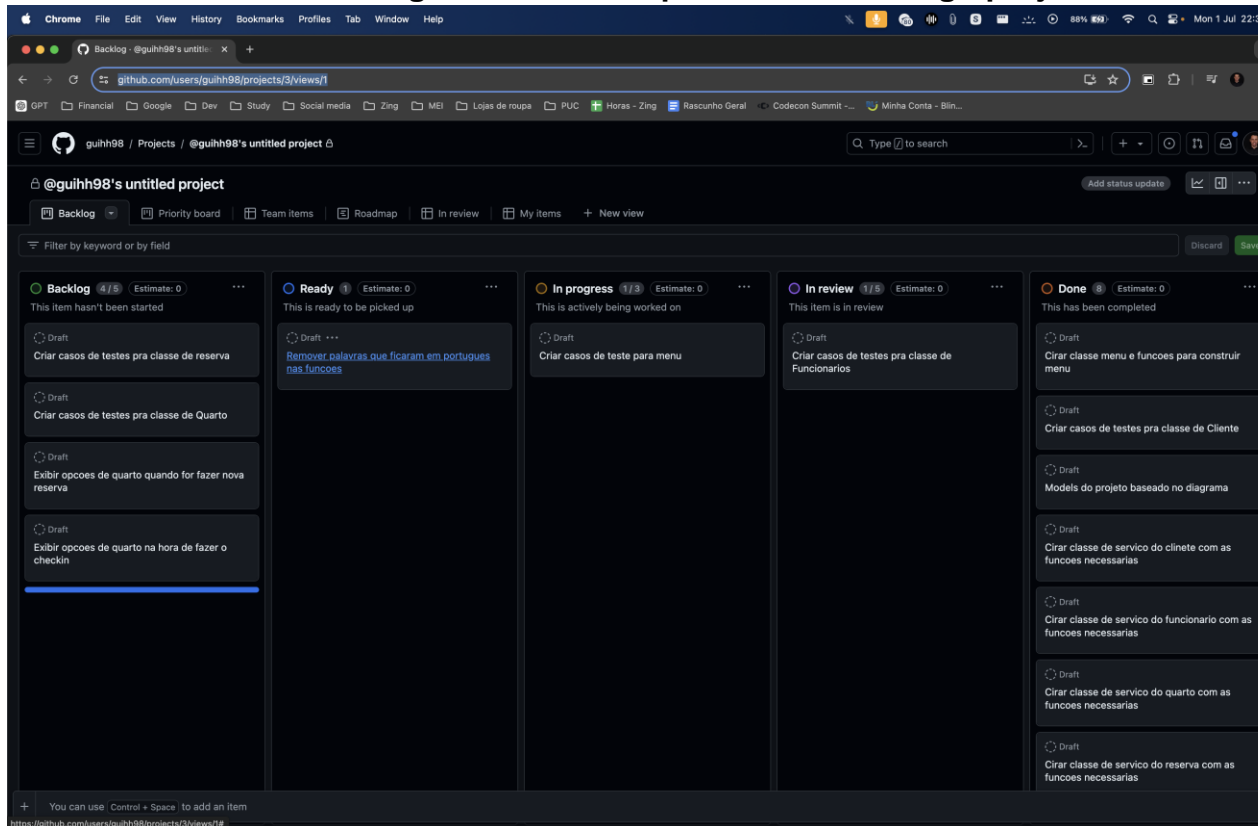
Hotel - Descanso Garantido

Apresentação:

O sistema tem objetivo de gerenciar um sistema hoteleiro, permitindo o usuário criar e salvar clientes, quartos, funcionários e reservas. Todos os dados são salvos em arquivos json.

A Figura 1 apresenta o board do produto com todas as tasks

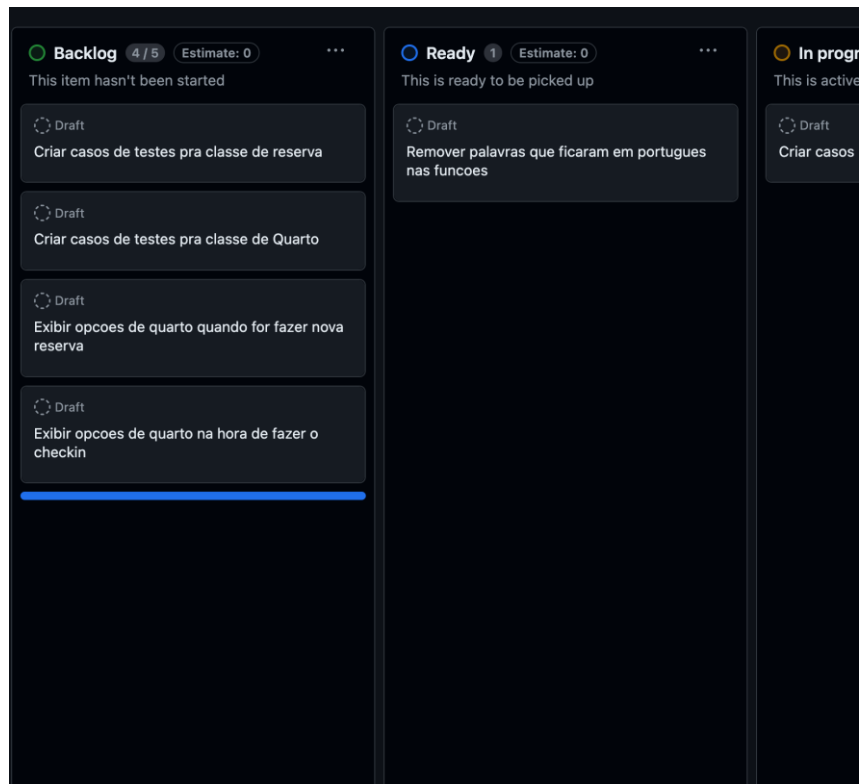
Figura 1 – Time e quadros criado no git projetos



A Figura 2 apresenta o protótipo de tela realizado no início do projeto mostrando no *backlog* geral a divisão das funções e quais tarefas os integrantes deveriam finalizar em cada sprint.

Este quadro serviu como base para cada um dos desenvolvedores criar o seu próprio *backlog*.

Figura 2 - Backlog geral



Observação: É possível acessar esse quadro no projeto do repositório: <https://github.com/guihh98/projects/3/views/1>

Lista de assinaturas das funções e parâmetros

Explicação da estrutura de dados principal do programa.

Apresentação das assinaturas das funções

As funções e parâmetros utilizados no programa foram:

1. registerClient (Client client)

A função permite cadastrar um novo cliente no sistema. Solicita ao usuário informações como código, nome, endereço, telefone e CPF do cliente e armazena esses dados na estrutura clientes.

Esta função não possui retorno, pois modifica diretamente o array de clientes e o número total de clientes via ponteiro.

Parâmetros:

- `Cliente`: É uma classe definida no projeto

2. checkIn (int clienteId, int quartoId)

A função permite cadastrar uma nova estadia no sistema. Solicita ao usuário informações como o código do cliente, verifica a disponibilidade do quarto e atualiza as informações necessárias.

Esta função não possui retorno, pois modifica diretamente os arrays de estadias, clientes e quartos, além do número total de estadias via ponteiro.

Parâmetros:

- **clienteId**: Id do cliente que fara a reserva
- **quartoId**: Id do quarto que sera hospedado.

3. registerEmployee (Employee newEmployee)

A função permite cadastrar um novo funcionário no sistema. Solicita ao usuário informações como código, nome, CPF, e-mail, telefone e armazena esses dados na estrutura funcionários.

Parâmetros:

- `newEmployee`: É uma classe modelo feita no projeto

4. registerNewRoom (Room room)

A função permite cadastrar um novo quarto no sistema. Solicita ao usuário informações como número do quarto, valor da diária e status (ocupado/desocupado) do quarto, e armazena esses dados na estrutura quartos.

Parâmetros:

- Room: Classe modelo feita no projeto

5. checkout ()

A função realiza o checkout de uma estadia específica, alterando o status do quarto para desocupado e removendo a estadia do array de estadias.

Parâmetros:

- `quartoId`: Id do quarto

6. getAllReserves ()

A função lista todas as estadias, exibindo detalhes como data de entrada, data de saída.

7. searchClientById (int clientId)

A função pesquisa um cliente no array de clientes com base em uma chave fornecida e uma função de comparação. Retorna um ponteiro para o cliente encontrado ou NULL se não for encontrado.

Parâmetros:

- `clientId`: Id do cliente pra ser buscado

8. searchEmployeeById(int id)

A função pesquisa um funcionário no array de funcionários com base em uma chave fornecida e uma função de comparação. Retorna um ponteiro para o funcionário encontrado ou NULL se não for encontrado.

Parâmetros:

- `employeeId`: Id do funcionario

9. `getAllRooms ()`

A função exibe a lista de todos os quartos cadastrados, mostrando informações como número do quarto e status (ocupado/desocupado).

Esta função não possui retorno, pois apenas exibe as informações dos quartos cadastrados.

TESTES

Casos de teste do software:

Os casos de teste englobam todo o código, uma vez que as funções devem receber parâmetros que podem estar dentro de outras funções ou do código principal.

Entradas Função: <code>registerClient</code> (<code>Client client</code>)	Classes Válidas	Resultado Esperado	Classes Inválidas	Resultado Esperado

<p>Cliente com código único, nome, endereço, telefone, CPF</p>	<p>Cliente com dados válidos, exemplo: {1, "João", "Rua A", "123456789", "111.222.333-44"}</p>	<p>Cliente cadastrado com sucesso, incremento no número de clientes</p>	<p>Cliente com código duplicado, campos vazios ou inválidos</p>	<p>Mensagem de erro, cliente não cadastrado</p>
<p>Entradas Função: registerEmployee (Employee newEmployee)</p>	<p>Classes Válidas</p>	<p>Resultado Esperado</p>	<p>Classes Inválidas</p>	<p>Resultado Esperado</p>
<p>Funcionario com código único, nome, telefone, cargo, salário</p>	<p>Funcionário com dados válidos, exemplo: {1, "Maria", "987654321", "Gerente", 3000.0}</p>	<p>Funcionário cadastrado com sucesso, incremento no número de funcionários</p>	<p>Funcionário com código duplicado, campos vazios ou inválidos</p>	<p>Mensagem de erro, funcionário não cadastrado</p>
<p>Entradas Função: registerNewRoom (Room room)</p>	<p>Classes Válidas</p>	<p>Resultado Esperado</p>	<p>Classes Inválidas</p>	<p>Resultado Esperado</p>

Quarto com número único, quantidade de hóspedes, valor diária, status	Quarto com dados válidos, exemplo: {101, 2, 150.0, "desocupado"}	Quarto cadastrado com sucesso, incremento no número de quartos	Quarto com número duplicado, campos vazios ou inválidos	Mensagem de erro, quarto não cadastrado
Entradas Função: <code>checkIn (int clienteId, int quartoId)</code>	Classes Válidas	Resultado Esperado	Classes Inválidas	Resultado Esperado
Estadia com código de cliente válido, quantidade de hóspedes, data de entrada e saída, quarto disponível	Estadia com dados válidos, exemplo: {1, 2, "2023-06-01", "2023-06-05", 101}	Estadia cadastrada com sucesso, alteração do status do quarto	Estadia com cliente ou quarto inexistente, datas inválidas, quarto ocupado	Mensagem de erro, estadia não cadastrada
Entradas Função: <code>public static void checkout (int quartoId)</code>	Classes Válidas	Resultado Esperado	Classes Inválidas	Resultado Esperado





Estadia existente, cliente válido, quarto válido	Estadia com dados válidos, exemplo: {1, 2, "2023-06-01", "2023-06-05", 101}	Checkout realizado, cálculo do valor total, alteração do status do quarto	Estadia inexistente, cliente ou quarto inválido.	Mensagem de erro, checkout não realizado
Entradas Função: public static void getAllReserves ()	Classes Válidas	Resultado Esperado	Classes Inválidas	Resultado Esperado
Código de cliente existente	Cliente existente com estadias, exemplo: 1	Listagem de todas as estadias do cliente	Cliente inexistente, código inválido	Mensagem de erro, estadias não listadas
Entradas Função: public static List<Cliente> searchAllClients ()	Classes Válidas	Resultado Esperado	Classes Inválidas	Resultado Esperado

Chave de pesquisa válida (código ou nome), clientes existentes	Chave válida, exemplo: código 1 ou nome "João"	Cliente encontrado, dados exibidos	Chave inválida, cliente inexistente	Mensagem de erro, estadias não listadas
Entradas public static Employee searchEmployeeById(int id)	Classes Válidas	Resultado Esperado	Classes Inválidas	Resultado Esperado
Chave de pesquisa válida (código), funcionários existentes	Chave válida, exemplo: código 1	Funcionário encontrado, dados exibidos	Chave inválida, funcionário inexistente	Mensagem de erro, funcionário não encontrado

Relatório de Execução de Testes

Segue um print dos resultados dos testes unitários, e dentro do repositório tem o arquivo com relatório detalhado dos testes.

<https://github.com/guihh98/TrabalhoFundamentos/tree/main/documents>

- ✓  TrabalhoFundamentos.Tests (9 tests) Failed: 1 test failed
 - ✓  {} TrabalhoFundamentos.Tests.Service (9 tests) Failed: 1 test failed
 - ✓ ✓ ClienteServicesTest (3 tests) Success
 - ✓ TestReGisterClient Success
 - ✓ TestSearchClient Success
 - ✓ TestUnkowClnet Success
 - ✓  EmployeeServiceTest (4 tests) Failed: One or more child tests failed: 1 test failed
 -  TesteBuscarTodosFuncionarios Failed: Assert.Equal() Failure: Values differ
 - ✓ TestRegisterNewEmployee Success
 - ✓ TestSearchEmployee Success
 - ✓ TestUnkowEmployee Success
 - ✓ ✓ MenuTest (1 test) Success
 - ✓ METHOD Success
 - ✓ ✓ ReserveServiceTest (1 test) Success
 - ✓ METHOD Success