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

Bernardo Parreiras Prado e João Paulo Aguiar Prado

Villa Prado Resort - Um Hotel 5 Estrelas

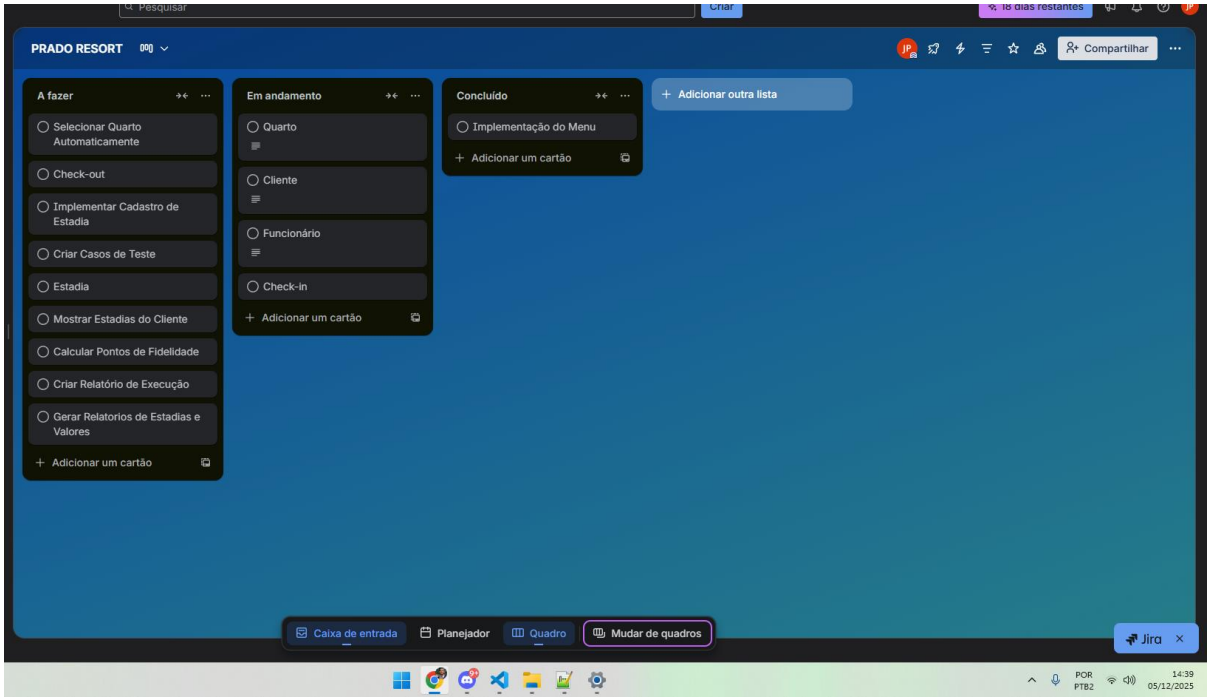
LINK DO VIDEO NO GOOGLE DRIVE : video villa prado resort

Apresentação:

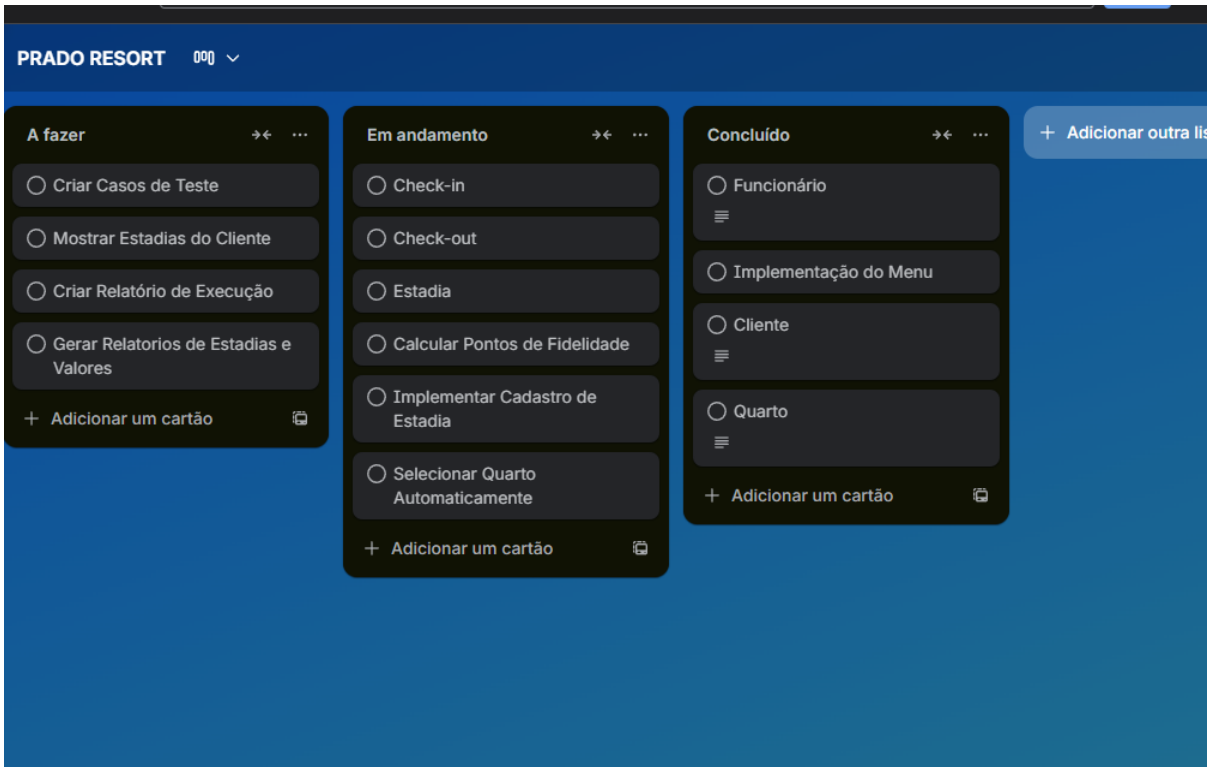
O sistema desenvolvido tem como objetivo otimizar a gestão diária de um hotel por meio da automação de suas principais operações. Ele oferece um cadastro completo de clientes e funcionários, possibilita a realização de reservas com seleção automática do quarto adequado com base no número de hóspedes e na disponibilidade, e gerencia todo o fluxo de estadias, incluindo check-in, check-out e cálculo de valores. Além disso, o sistema gera relatórios automatizados que auxiliam na análise de ocupação e desempenho do hotel, garantindo maior eficiência, organização e controle das atividades internas.

Backlog to produto:

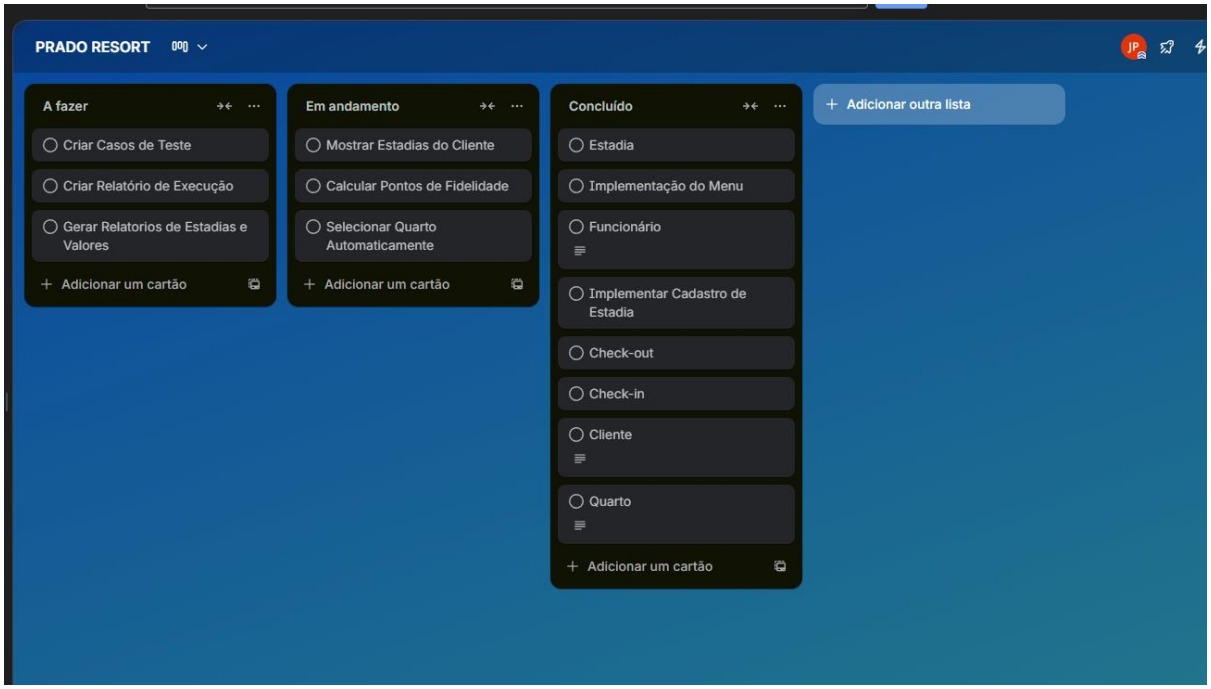
Figura representando o Meio da Sprint 1 (Utilizamos o Trello para monitoramento de backlog entre as sprints.



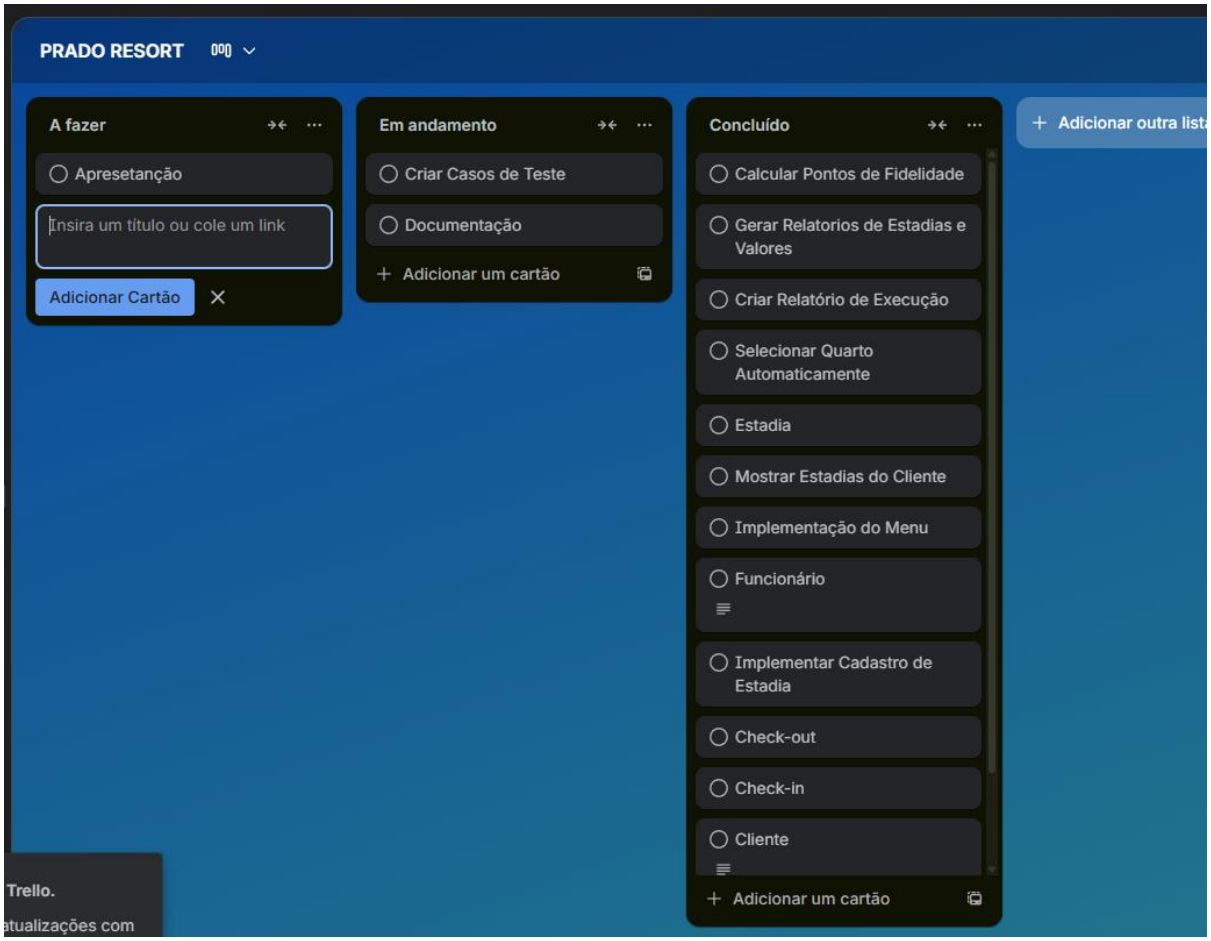
Final Sprint 1, (Em andamento oque foi planejado pela planning para ser executado na sprint 2)



Final Sprint 2, (Em andamento oque foi planejado pela planning para ser executado na sprint 3)



Final Sprint 3, (Em andamento oque foi planejado pela planning para ser executado na sprint 4 (Ultima sprint))



Lista de assinaturas das funções e parâmetros:

ARQUIVO: src/models/hotel.py CLASSE: Hotel

1. cadastrar_cliente(self, nome, endereço, telefone) cadastra um novo cliente no sistema, gerando automaticamente um código único. Retorna o objeto Cliente criado e armazena-o na lista interna de clientes.

2. cadastrar_funcionario(self, nome, telefone, cargo, salario) registra um novo funcionário com cargo, salário e informações pessoais. O código é gerado automaticamente. Retorna o objeto Funcionário criado.

3. adicionar_quarto(self, numero, tipo, quantidade_hospedes, preco_diaria) Inclui um novo quarto no hotel, desde que o número não esteja cadastrado. Retorna True se adicionado com sucesso ou false caso o quarto já exista.
4. cadastrar_estadia(self, codigo_cliente, quantidade_hospedes, data_entrada, data_saida) Cria uma estadia para um cliente, buscando automaticamente um quarto disponível com capacidade adequada. Retorna o objeto Estadia criado ou None se não houver quarto livre.

5. buscar_cliente_por_codigo(self, codigo) Localiza um cliente pelo código. Retorna o objeto Cliente ou None se não encontrado.

6. buscar_quarto_por_numero(self, numero) Localiza um quarto pelo número informado. Retorna o objeto Quarto ou None.

7. buscar_estadia_por_codigo(self, codigo) Retorna a estadia correspondente ao código informado ou None caso não exista.

8. verificar_disponibilidade(self, numero_quarto, data_entrada, data_saida) Verifica se um quarto está livre para o período desejado. Retorna True se não houver conflitos com outras estadias, caso contrário retorna False.

9. fazer_checkin(self, codigo_estadia) Realiza o check-in de uma estadia, ocupando o quarto e atualizando seu status. Retorna True em caso de sucesso ou False em caso de erro.

10. fazer_checkout(self, codigo_estadia, data_checkout=None) Finaliza uma estadia, recalculando valores se necessário e liberando o quarto. Retorna uma tupla indicando sucesso ou erro e o valor da estadia.

11. listar_clientes(self) Retorna a lista de todos os clientes cadastrados no hotel.

12. listar_quartos_disponiveis(self) Retorna apenas os quartos com status “Disponível”.

13. listar_estadias_por_cliente(self, codigo_cliente) Lista todas as estadias associadas ao cliente informado.

14. cancelar_estadia(self, codigo) Cancela uma estadia e libera o quarto. Retorna True se cancelado ou False se não for possível.

15. relatorio_ocupacao(self) Gera um relatório contendo quantidade de quartos, quantos estão disponíveis, ocupados ou em manutenção, além da taxa de ocupação.

16. relatorio_receita(self) Retorna um relatório financeiro com receita total, concluída, pendente e número de estadias.
17. salvar_dados(self, arquivo='data/hotel_dados.bin') Salva todos os dados do hotel em um arquivo binário, criando diretórios se necessário. Retorna True se salvar com sucesso.

18. carregar_dados(self, arquivo='data/hotel_dados.bin') Carrega dados salvos anteriormente do arquivo binário. Retorna True se carregado ou False se não for possível.

19. calcular_pontos_fidelidade(self, estadias) Soma todas as diárias das estadias do cliente e retorna a pontuação total (10 pontos por diária).

20. marcar_ocupado(self) Marca o quarto como “Ocupado”.

21. marcar_desocupado(self) Libera o quarto, deixando-o com status “Disponível”.

22. marcar_manutencao(self) Coloca o quarto em status “Manutenção”, impedindo novos usos.

23. calcular_diarias(self) Retorna a quantidade de diárias da estadia.

24. calcular_valor_total(self) Retorna o valor total multiplicando diárias pelo preço da diária do quarto.

25. confirmar(self) Confirma a estadia se ainda estiver pendente.

26. cancelar(self) Cancela a estadia e libera o quarto automaticamente.

27. fazer_checkin(self) Realiza o check-in, marcando o quarto como ocupado.

28. fazer_checkout(self, data_checkout=None) Realiza o check-out, recalculando valores quando necessário e liberando o quarto.

29. validar_data(data_str, formato='%d/%m/%Y') Converte uma string em data válida. Retorna a data convertida ou None se inválida.

30. validar_numero(txt, minimo=None) Valida se o texto representa um número inteiro. Retorna o inteiro ou None se inválido.

31. validar_preco(valor_str) Valida valores monetários aceitando vírgula ou ponto. Retorna o valor em float.

32. validar_tipo_quarto(tipo) Padroniza e valida o tipo de quarto. Retorna "Simples", "Duplo" ou "Suíte".

33. limpar_tela() Limpa o console do sistema operacional.

34. pausar() Aguarda o usuário pressionar ENTER para continuar.

TESTS:

Casos de teste do software:

	Entradas	Classes Válidas	Resultado Esperado	Classes Inválidas	Resultado Esperado2
2	Cliente: Nome (string), Endereço (string), Telefone (string)	Nome válido não vazio, endereço válido, telefone válido. Ex: 'Maria Silva', 'Rua A 123', '31999999999'	Cliente cadastrado com código automático sequencial (1, 2, 3...), retorna objeto Cliente	Nome vazio ou None, endereço vazio, telefone vazio ou inválido	Sistema rejeira cadastro de cliente
3	Funcionário: Nome, Telefone, Cargo, Salário	Nome válido, telefone válido, cargo válido (Recepcionista, Gerente, etc), salário numérico positivo. Ex: 'Ana Costa', '31988888888', 'Recepcionista', 2500.00	Funcionário cadastrado com código automático (1, 2, 3...), todos campos armazenados corretamente	Nome vazio, cargo inválido, salário negativo ou zero, telefone vazio	Retorno de funcionario não cadastrado Rejeitado pelo sistema
4	Quarto: Número, Tipo, Quantidade Hóspedes, Preço Diária	Número único não duplicado (101, 102, 201), tipo válido (Simples, Duplo, Suite), quantidade_hospedes inteiro positivo (1, 2, 4), preco_diaria float positivo (150.00, 250.00)	Quarto adicionado com sucesso, status inicial 'Disponível' (maiúscula), retorna True	Número duplicado (tentar adicionar 101 novamente), quantidade_hospedes <=0, preco_diaria <=0	Sistema rejeita cadastro duplicado (retorna False)
5	Estadia : Código Cliente, Quantidade Hóspedes, Datas	Cliente existente (código 1, 2, 3), quantidade_hospedes válida (1, 2, 4), data_entrada < data_saida, exemplo: cliente=1, hospedes=2, entrada=10/12/2025, saida=12/12/2025	Sistema busca automaticamente quarto disponível com capacidade suficiente, cria estadia com código único, calcula diárias e valor_total, status='Confirmada'	Cliente inexistente (código 999), data_saida <= data_entrada, quantidade_hospedes maior que qualquer quarto disponível, sem quartos disponíveis	Sistema retorna None, estadia não criada
6	Busca Cliente	Código numérico válido (1, 2, 3) ou nome parcial/completo ('Maria', 'Silva', 'maria silva')	Retorna lista com cliente(s) encontrado(s), busca case-insensitive por nome	Código inexistente (82362), nome inexistente ('LX'), string vazia	Retorna lista vazia []
7	Busca Funcionário	Código numérico válido ou nome parcial/completo existente	Retorna lista com funcionário(s) encontrado(s), busca case-insensitive	Código inexistente, nome inexistente	Retorna lista vazia []
8	Busca Quarto: Número do quarto	Número válido de quarto existente (101, 102, 201)	Retorna objeto Quarto com todos os dados (numero, tipo, quantidade_hospedes, preco_diaria, status)	Número inexistente (999, 500)	Retorna None
9	Check-in: Código da estadia	Estadia existente com status='Confirmada'	Marca quarto como 'Ocupado' (maiúscula), retorna True	Estadia inexistente, status diferente de 'Confirmada' (Pendente, Cancelada, Concluida)	Retorna False, quarto não é marcado como ocupado
10	Checkout: Código da estadia	Estadia existente com status='Confirmada', data_checkout >= data_entrada	Marca quarto como 'Disponível' (maiúscula), status='Concluida', recalcula diárias e valor se data_checkout diferente da prevista, retorna (True,	Estadia inexistente, status != 'Confirmada', data_checkout < data_entrada	Retorna (False, mensagem_erro)
11	Pontos Fidelidade: Código do cliente	Cliente existente com ou sem estadias	Calcula e retorna total de pontos: soma de (quantidade_diarias × 10) de todas estadias do cliente	Cliente inexistente	Erro ou retorna 0 dependendo da implementação
12	Cancelamento Estadia: Código da estadia	Estadia existente com status='Pendente' ou 'Confirmada'	Status alterado para 'Cancelada', quarto marcado como 'Disponível' (maiúscula), retorna True	Estadia inexistente, status='Concluida' ou 'Cancelada'	Retorna False, nenhuma alteração
13	Listar Estadias Cliente: Código do cliente	Cliente existente, código válido (1, 2, 3)	Retorna lista com todas estadias do cliente (qualquer	Cliente inexistente	Retorna lista vazia []
14	Relatório Ocupação: Sistema com quartos	Sistema possui quartos cadastrados, pelo menos um quarto com status diferente de 'Disponível'	Retorna dict com total_quartos, quartos_disponiveis (status='Disponível'), quartos_ocupados (status='Ocupado'), quartos_manutencao (status='Manutenção'), taxa_ocupacao (porcentagem)	Sistema sem quartos cadastrados	Retorna total_quartos=0, taxa_ocupacao=0
15	Relatório Receita: Sistema com estadias	Sistema possui estadias em diferentes status (Confirmada, Concluida, Cancelada)	Retorna dict com receita_total, receita_concluida, receita_pendente, total_estadias	Sistema sem estadias	Retorna todos valores zerados
16					
17					

Testes executados:

	ID Teste	Módulo	Descrição do Teste	Status	Resultado	Observações
2	TC-CLI-001	Clientes	Cadastrar cliente com dados válidos	Executado	Aprovado	Código automático gerado (1-2-3). Campos: nome-endereco-telefone
3	TC-CLI-002	Clientes	Buscar cliente por código existente	Executado	Aprovado	buscar_cliente_por_codigo(1) retorna objeto Cliente correto
4	TC-CLI-003	Clientes	Buscar cliente por código inexistente	Executado	Aprovado	buscar_cliente_por_codigo(999) retorna None
5	TC-CLI-004	Clientes	Pesquisar cliente por nome parcial	Executado	Aprovado	pesquisar_cliente('Maria') encontra 'Maria Silva' (case-insensitive)
6	TC-CLI-005	Clientes	Pesquisar cliente por código string	Executado	Aprovado	pesquisar_cliente('1') converte e retorna cliente código 1
7	TC-CLI-006	Clientes	Listar todos os clientes	Executado	Aprovado	listar_clientes() retorna lista completa
8	TC-FUNC-00	Funcionários	Cadastrar funcionário com dados válidos	Executado	Aprovado	Código automático. Campos: nome-telefone-cargo-salario
9	TC-FUNC-00	Funcionários	Buscar funcionário por código	Executado	Aprovado	buscar_funcionario_por_codigo retorna funcionário com cargo e salário
10	TC-FUNC-00	Funcionários	Pesquisar funcionário por nome	Executado	Aprovado	pesquisar_funcionario busca por nome parcial case-insensitive
11	TC-FUNC-00	Funcionários	Pesquisar funcionário por código string	Executado	Aprovado	Conversão automática de string para int na busca
12	TC-QTO-00	Quartos	Adicionar quarto com dados válidos	Executado	Aprovado	adicionar_quarto retorna True. Status inicial 'Disponível'
13	TC-QTO-00	Quartos	Impedir cadastro de número duplicado	Executado	Aprovado	adicionar_quarto(101) duplicado retorna False
14	TC-QTO-00	Quartos	Buscar quarto por número	Executado	Aprovado	buscar_quarto_por_numero(101) retorna objeto Quarto
15	TC-QTO-00	Quartos	Validar estrutura do quarto conforme código	Executado	Aprovado	Campos: numero-quantidade_hospedes-preco_diaria-status
16	TC-QTO-00	Quartos	Validar status inicial Disponível	Executado	Aprovado	Status inicial = 'Disponível' (maiúscula)
17	TC-QTO-00	Quartos	Testar mudança de status ocupado/desocupado	Executado	Aprovado	marcar_ocupado() -> 'Ocupado'; marcar_desocupado() -> 'Disponível' (maiúsculas)
18	TC-QTO-00	Quartos	Listar quartos disponíveis	Executado	Aprovado	listar_quartos_disponiveis() filtra status='Disponível'
19	TC-QTO-00	Quartos	Validar campo quantidade_hospedes tipo int	Executado	Aprovado	quantidade_hospedes é int (não float ou capacidade)
20	TC-EST-001	Estadias	Cadastrar estadia com busca automática	Executado	Aprovado	cadastrar_estadia busca quarto por quantidade_hospedes automaticamente
21	TC-EST-002	Estadias	Rejeitar estadia com cliente inexistente	Executado	Aprovado	cadastrar_estadia com codigo_cliente=999 retorna None
22	TC-EST-003	Estadias	Rejeitar estadia com data_saida <= data_entrada	Executado	Aprovado	Validação de datas funciona. Retorna None
23	TC-EST-004	Estadias	Rejeitar quando sem quarto com capacidade	Executado	Aprovado	Busca automática retorna None se quantidade_hospedes > todos quartos
24	TC-EST-005	Estadias	Calcular diárias corretamente	Executado	Aprovado	quantidade_diarias = (data_saida - data_entrada).days
25	TC-EST-006	Estadias	Calcular valor_total corretamente	Executado	Aprovado	valor_total = quantidade_diarias × preco_diaria
26	TC-EST-007	Estadias	Estadia criada com status Confirmada	Executado	Aprovado	Status inicial após cadastrar_estadia é 'Confirmada'
27	TC-EST-008	Estadias	Verificar disponibilidade por período	Executado	Aprovado	verificar_disponibilidade impede sobreposição de datas
28	TC-EST-009	Estadias	Fazer estadia manual com número de quarto	Executado	Aprovado	fazer_estadia permite escolher quarto manualmente
29	TC-EST-010	Estadias	Buscar estadia por código	Executado	Aprovado	buscar_estadia_por_codigo retorna objeto Estadia
30	TC-EST-011	Estadias	Listar estadias de cliente específico	Executado	Aprovado	listar_estadias_por_cliente retorna todas estadias do cliente
31	TC-CHK-00	Check-in/out	Fazer check-in em estadia confirmada	Executado	Aprovado	fazer_checkin marca quarto como 'Ocupado' (maiúscula) e retorna True
32	TC-CHK-00	Check-in/out	Rejeitar check-in se status diferente	Executado	Aprovado	fazer_checkin retorna False se status != 'Confirmada'
33	TC-CHK-00	Check-in/out	Fazer checkout com data padrão	Executado	Aprovado	fazer_checkout sem data usa date.today(). Retorna (True-valor_total)
34	TC-CHK-00	Check-in/out	Checkout recalcula valor se data diferente	Executado	Aprovado	Se data_checkout != data_saida prevista recalcula diárias e valor
35	TC-CHK-00	Check-in/out	Checkout marca quarto disponível e estadia concluí	Executado	Aprovado	Status='Concluida' e quarto.status='Disponível' (maiúscula) (maiúscula)
36	TC-CHK-00	Check-in/out	Rejeitar checkout se data < entrada	Executado	Aprovado	Validação impede checkout antes de check-in
37	TC-CANC-0	Cancelament	Cancelar estadia confirmada	Executado	Aprovado	cancelar_estadia altera status='Cancelada' e quarto='Disponível'
38	TC-CANC-0	Cancelament	Rejeitar cancelamento se já concluída	Executado	Aprovado	Só cancela se status='Pendente' ou 'Confirmada'
39	TC-PONT-0	Pontos	Calcular pontos com múltiplas estadias	Executado	Aprovado	calcular_pontos_fidelidade soma (diárias × 10) de todas estadias
40	TC-PONT-0	Pontos	Cliente sem estadias retorna 0 pontos	Executado	Aprovado	Cliente novo sem histórico = 0 pontos
41	TC-REL-001	Relatórios	Relatório de ocupação com cálculo correto	Executado	Aprovado	taxa_ocupacao = (quartos_ocupados/total) × 100
42	TC-REL-002	Relatórios	Relatório de receita por status	Executado	Aprovado	Separa receita_concluida e receita_pendente corretamente
43	TC-PERS-0	Persistência	Salvar dados em arquivo pickle	Executado	Aprovado	salvar_dados() cria arquivo .bin (NÃO JSON) usando pickle
44	TC-PERS-0	Persistência	Carregar dados de arquivo pickle	Executado	Aprovado	carregar_dados() restaura tudo de arquivo .bin pickle
45	TC-PERS-0	Persistência	Verificar integridade após reload	Executado	Aprovado	Campos preservados: quantidade_hospedes e status='Disponível'
46	TC-VAL-001	Validações	Cliente deve existir para criar estadia	Executado	Aprovado	cadastrar_estadia com codigo_cliente=999 retorna None
47	TC-VAL-002	Validações	Quarto deve existir para estadia manual	Executado	Aprovado	fazer_estadia com numero_quarto=999 retorna None
48	TC-VAL-003	Validações	Sistema usa 3 status de quarto	Executado	Aprovado	Status válidos: 'Disponível'-'Ocupado'-'Manutenção' (maiúsculas)
49						