Caso de Uso 1: Autenticar Usuário

Perspectiva do Usuário: O objetivo é acessar o sistema de forma segura. O usuário espera uma tela de login simples (email/senha), com feedback claro em caso de sucesso ou falha (ex: "credenciais inválidas"). Uma funcionalidade de "esqueci minha senha" seria um recurso esperado.

Perspectiva do Projetista: Esta funcionalidade envolve uma tela de login (UI), um serviço de autenticação no backend e a consulta à tabela de usuários. As regras de negócio incluem a validação das credenciais, o armazenamento seguro de senhas com hash (ex: bcrypt), o gerenciamento da sessão do usuário (ex: JWT) e a implementação de um controle de acesso baseado em papéis (RBAC) para diferenciar usuários comuns de administradores. O desafio principal é garantir a segurança em todo o processo.

Perspectiva do Testador: Os testes devem cobrir o login bem-sucedido de usuários e administradores. Cenários de falha incluem tentativas com senha incorreta, email não existente e campos vazios. As validações devem verificar o redirecionamento correto pós-login, a exibição de mensagens de erro adequadas e o correto ciclo de vida da sessão (criação e destruição).

Caso de Uso 2: Gerenciar Usuários

Perspectiva do Usuário (Administrador): O objetivo é controlar quem pode acessar o sistema. O administrador espera uma tela com a lista de usuários, com funcionalidades para buscar, editar e excluir contas, além de um formulário para adicionar novos usuários. Uma confirmação antes da exclusão é essencial para evitar erros.

Perspectiva do Projetista: Envolve um painel de administração (UI) e uma API para operações CRUD (Criar, Ler, Atualizar, Deletar) na entidade de usuário. As regras de negócio determinam que apenas administradores podem acessar a funcionalidade, o email deve ser único no sistema e a exclusão pode ser lógica (soft delete) para preservar a integridade dos dados históricos. O principal desafio é a correta implementação da autorização.

Perspectiva do Testador: Os testes devem validar todas as operações CRUD. Cenários de falha incluem a tentativa de criar um usuário com email duplicado, submeter formulários com dados inválidos e o acesso à funcionalidade por um usuário não autorizado. As validações devem confirmar a atualização da lista de usuários, a persistência correta dos dados no banco e a aplicação rigorosa das regras de autorização.

Caso de Uso 3: Definir Regras Globais de Consumo

Perspectiva do Usuário (Administrador): O objetivo é padronizar os cálculos de consumo para todos os churrascos. O administrador espera uma tela de configurações para inserir valores padrão (ex: gramas de carne por pessoa, volume de bebidas) e salvá-los facilmente.

Perspectiva do Projetista: Envolve uma tela de configurações, um serviço de backend para gerenciar essas regras e uma tabela específica. As regras devem ser carregadas e aplicadas nos cálculos do carrinho de compras. Um desafio é definir o comportamento do sistema quando uma regra global é alterada: eventos já criados devem manter as regras vigentes no momento de sua criação.

Perspectiva do Testador: Os testes devem validar o salvamento e a atualização das regras. Cenários de falha incluem a tentativa de salvar dados inválidos (não numéricos, negativos). A validação principal consiste em criar um novo churrasco e verificar se os cálculos do carrinho utilizam as regras recém-definidas.

Caso de Uso 4: Criar e Editar Churrasco

Perspectiva do Usuário (Criador): O objetivo é registrar um novo evento de churrasco ou corrigir informações de um existente. O usuário espera um formulário intuitivo, com seletores de data e hora, e ser direcionado para a página de detalhes do evento após salvá-lo.

Perspectiva do Projetista: Envolve um formulário de evento na UI e uma API para operações CRUD na entidade "Churrasco". Regras de negócio importantes são: o usuário logado é definido como criador, campos principais são obrigatórios e a data do evento não pode estar no passado. O desafio é garantir que apenas o criador do evento possa editá-lo.

Perspectiva do Testador: Os testes devem cobrir a criação e a edição bem-sucedidas. Cenários de falha incluem salvar com campos obrigatórios em branco, usar uma data passada ou a tentativa de edição por um usuário não autorizado. As validações confirmam se o evento aparece na lista de churrascos e se os dados foram persistidos corretamente no banco.

Caso de Uso 5: Visualizar Meus Churrascos

Perspectiva do Usuário: O objetivo é ter uma visão rápida de todos os churrascos que criei ou para os quais fui convidado. A expectativa é uma lista clara com as informações principais (título, data) e a possibilidade de clicar para ver os detalhes completos.

Perspectiva do Projetista: Requer uma tela na UI e um endpoint na API que liste os eventos. A regra de negócio principal é a consulta ao banco, que deve filtrar os churrascos onde o usuário logado é o criador OU está na lista de convidados. O desafio técnico é otimizar essa consulta, possivelmente com paginação, para garantir a performance.

Perspectiva do Testador: Os cenários de teste devem validar se a lista exibida está correta para diferentes perfis: um usuário que só cria eventos, um que só é convidado e um que faz ambos. Um cenário de falha a ser tratado é quando um novo usuário, sem eventos, acessa a tela, devendo receber uma mensagem apropriada.

Caso de Uso 6: Convidar Participantes

Perspectiva do Usuário (Criador): O objetivo é convidar amigos para o churrasco. O usuário espera uma interface que permita selecionar usuários já cadastrados ou inserir novos e-mails, disparando um convite por e-mail para eles.

Perspectiva do Projetista: Envolve uma UI de convite, uma API e um serviço de notificação para envio de e-mails. A regra de negócio é associar o convite a um usuário existente, se o e-mail já for cadastrado, ou apenas enviar o convite. O desafio principal é a integração com um serviço de e-mail externo e o tratamento de possíveis falhas no envio.

Perspectiva do Testador: Os testes devem cobrir o convite para usuários já cadastrados e para e-mails novos. Cenários de falha incluem e-mails com formato inválido ou a tentativa de convidar alguém que já está na lista. A validação crucial é confirmar o recebimento do e-mail de convite.

Caso de Uso 7: Acompanhar Pagamentos dos Participantes

Perspectiva do Usuário (Criador): O objetivo é controlar quem já pagou. O usuário espera uma lista de convidados com o status de pagamento (pendente/confirmado) e a capacidade de alterar esse status manualmente de forma simples.

Perspectiva do Projetista: A implementação envolve o painel do churrasco na UI e uma API para atualizar o status de pagamento. A regra de negócio principal é que apenas o criador do evento pode fazer essa alteração. Por ser um processo manual, o foco do desafio técnico é na usabilidade.

Perspectiva do Testador: Os testes devem validar a alteração do status de "pendente" para "confirmado" e vice-versa. O cenário de falha principal é a tentativa de um participante alterar o próprio status. As validações confirmam a atualização visual na tela e a persistência da mudança no banco.

Caso de Uso 8: Montar Carrinho com Alertas de Consumo

Perspectiva do Usuário (Criador): O objetivo é criar a lista de compras com o auxílio do sistema para evitar comprar quantidades erradas. A expectativa é uma interface para adicionar itens e receber alertas em tempo real sobre as quantidades por pessoa, baseando-se nas regras globais.

Perspectiva do Projetista: Requer uma UI com lógica de frontend robusta (ex: JavaScript) para realizar os cálculos em tempo real. O sistema precisa contar os participantes confirmados, buscar as regras de consumo e comparar os resultados para exibir os alertas. O desafio é garantir que essa interação seja fluida e sem atrasos.

Perspectiva do Testador: Os testes devem simular a adição e remoção de itens e participantes para validar a exibição e o desaparecimento dos alertas nos momentos corretos. A precisão dos cálculos deve ser rigorosamente verificada, incluindo o tratamento de casos como a divisão por zero (quando não há participantes).

Caso de Uso 9: Registrar Compras e Anexar Comprovantes

Perspectiva do Usuário (Criador): O objetivo é registrar os gastos do evento e guardar as notas fiscais para a prestação de contas. A expectativa é um formulário simples para lançar despesas, com a opção de anexar um arquivo de comprovante.

Perspectiva do Projetista: Envolve um formulário na UI, uma API para as despesas e um serviço para upload de arquivos, integrado a um sistema de armazenamento (como Amazon S3). As entidades são "Compra" e "Comprovante". O desafio técnico reside na implementação segura e eficiente do upload e armazenamento de arquivos.

Perspectiva do Testador: Os testes devem validar o registro de uma compra, com e sem anexo. Também devem verificar a visualização e o download do comprovante. Cenários de falha incluem o upload de arquivos muito grandes ou de formatos não permitidos. As validações verificam a persistência dos dados da compra e o correto armazenamento do arquivo.

Caso de Uso 10: Consolidar e Visualizar Prestação de Contas

Perspectiva do Usuário: Como criador, o objetivo é fechar o balanço financeiro. Como participante, é visualizar como o dinheiro foi gasto. A expectativa é um resumo claro com "Total Arrecadado", "Total Gasto" e "Saldo", além do acesso aos comprovantes.

Perspectiva do Projetista: A funcionalidade requer uma tela de resumo e um endpoint na API que realize os cálculos. A regra de negócio é: Saldo = (soma dos pagamentos confirmados) - (soma das compras registradas). O desafio é garantir a precisão dos cálculos, que dependem da integridade dos dados de outros casos de uso.

Perspectiva do Testador: Os testes devem ser realizados após um fluxo completo de confirmação de pagamentos e registro de compras. O resultado do balanço deve ser comparado com um cálculo manual para garantir a precisão. As permissões de acesso (visualização para participantes, gestão para o criador) também devem ser validadas.

Caso de Uso 11: Realizar Check-in de Participante

Perspectiva do Usuário (Criador): O objetivo é registrar a presença dos participantes no dia do evento. A expectativa é uma função simples que permita marcar o check-in de quem compareceu.

Perspectiva do Projetista: A funcionalidade deve estar disponível no painel do criador e ter uma regra de negócio para permitir o check-in apenas de participantes com pagamento confirmado.

Perspectiva do Testador: O principal cenário de falha a ser testado é a tentativa de fazer check-in de um participante com pagamento pendente. O sucesso é o registro correto do check-in para um participante pago.