UNIVERSIDADE SÃO JUDAS TADEU

Bruno Godoy Dias - 82311358 Matheus Henrique Oliveira Chuang - 823165173

Raí Joia Miquilino Valencio - 82318841

# Planejamento de Testes

Projeto A3

São Paulo 2025

Bruno Godoy Dias - 82311358 Matheus Henrique Oliveira Chuang - 823165173

Raí Joia Miquilino Valencio - 82318841

# Planejamento de Testes

Projeto A3 apresentado ao Gestão e qualidade de software, como parte dos requisitos necessários à obtenção do título de planejamento de testes.

**Professor:** Robson Calvetti **Disciplina:** Gestão e qualidade de software   
**Turma:** CCP1AN-MCD3-25070036

São Paulo 2025

O presente trabalho apresenta o planejamento de testes do sistema Easy Queue, desenvolvido para automatizar o atendimento da pizzaria Mooca’s Pizzas, respeitando sua proposta tradicional e humanizada. O objetivo do plano é garantir que os requisitos funcionais e não-funcionais do sistema sejam validados de forma sistemática e eficaz, assegurando qualidade e confiabilidade na entrega final. Como método, foi utilizada uma abordagem baseada na norma ISO/IEC 9126 e práticas de testes funcionais, de integração e usabilidade, aliada ao uso de ferramentas como Jest, Postman e Cypress, com apoio da infraestrutura em nuvem da Railway e integração contínua via GitHub Actions. Os testes foram organizados em cronograma específico, com definição de recursos e marcos do projeto, permitindo a rastreabilidade e a correção eficiente de falhas. Como resultado, espera-se que o sistema atenda satisfatoriamente aos fluxos definidos, oferecendo uma experiência fluida para clientes e funcionários. Conclui-se que o planejamento de testes é essencial para garantir a qualidade do software e contribuir para a excelência no atendimento digital da pizzaria.

**Palavras-chave:** Planejamento de Testes. Mooca’s Pizzas. Qualidade de Software.

ABNT. CI/CD.

[Planejamento de Testes 1](#_Toc200061653)

[Planejamento de Testes 2](#_Toc200061654)

[1. Planejamento de Testes de Software 4](#_Toc200061655)

[1.2. Alocação de Recursos 4](#_Toc200061656)

[1.3. Marcos do Projeto 4](#_Toc200061657)

[2.1.2. Escopo 5](#_Toc200061658)

[2.1.3. Recursos 5](#_Toc200061659)

[2.1.4. Estimativas de Projeto 5](#_Toc200061660)

[2.2. Documento de Requisitos 5](#_Toc200061661)

[2.3.1.2. Escopo 5](#_Toc200061662)

[2.3.1.3. Objetivos 6](#_Toc200061663)

[2.3.1.4. Requisitos a Serem Testados 6](#_Toc200061664)

[2.3.1.5. Estratégias, Tipos de Testes e Ferramentas 6](#_Toc200061665)

[2.3.1.6. Recursos a Serem Empregados 6](#_Toc200061666)

[2.3.1.7. Cronograma das Atividades 6](#_Toc200061667)

[2.3.1.8. Definição dos Marcos do Projeto 6](#_Toc200061668)

[2.3.2. Casos de Teste 6](#_Toc200061669)

[2.3.3. Roteiro de Testes 7](#_Toc200061670)

[• Passos: 7](#_Toc200061671)

[3. Gestão de Configuração de Software 7](#_Toc200061672)

[4. Repositório de Gestão de Configuração 7](#_Toc200061673)

# Planejamento de Testes de Software

* 1. **Cronograma de Atividades**

**Tabela 1 – Cronograma de Atividades**

|  |  |  |  |
| --- | --- | --- | --- |
| Atividade | Início | Término | Responsável |
| Elaboração do Plano de Testes | 10/06/2025 | 11/06/2025 | Equipe de QA |
| Criação de Casos de Testes | 12/06/2025 | 15/06/2025 | QA/Testadores |
| Execução dos Testes em Ambiente de Staging | 16/06/2025 | 20/06/2025 | QA + Devs |
| Correção de Bugs encontrados | 21/06/2025 | 24/06/2025 | Devs |
| Reteste e Validação Final | 25/06/2025 | 26/06/2025 | QA |
| Deploy para Produção | 27/06/2025 | 27/06/2025 | DevOps |

# Alocação de Recursos

**Tabela 2 – Alocação de Recursos**

|  |  |  |
| --- | --- | --- |
| Recurso | Papel | Quantidade |
| Desenvolvedores | Correção de bugs e suporte aos testes | 3 |
| Testadores (QA) | Elaboração, execução e registro de testes | 2 |
| Ambiente Staging (Railway) | Execução de testes | 1 |
| Banco de Dados PostgreSQL | Dados de teste simulados | 1 instância |
| Simuladores de Gateway e Notificações | Validação de integrações externas | 2 serviços mockados |
| CI/CD via GitHub Actions | Deploy automático após aprovação | 1 pipeline |

# Marcos do Projeto

* Finalização do Plano de Testes:**11/06/2025**
* Conclusão da Fase de Testes:**20/06/2025**
* Correções e Validações Finalizadas:**26/06/2025**
* Deploy para Produção:**27/06/2025**

1. **Documentos de Desenvolvimento de Software**
   1. **Plano de Projeto**
      1. **Planejamento do Projeto**

O projeto está sendo desenvolvido utilizando a metodologia ágil **SCRUM**, com ciclos semanais e refinamentos constantes com os stakeholders. A cada sprint, novas funcionalidades são implementadas e testadas em ambiente de staging. O pipeline de integração contínua (CI/CD) é mantido na plataforma **Railway**, com deploys automatizados a partir do **GitHub Actions**, permitindo entregas rápidas e validação contínua.

# Escopo

O sistema **Easy Queue** tem como objetivo modernizar o processo de atendimento da pizzaria **Mooca’s Pizzas**, oferecendo:

* Acesso à fila via QR Code;
* Gestão digital das mesas;
* Notificações automáticas para clientes e garçons;
* Controle e acompanhamento de pedidos;
* Painel da cozinha com pedidos em tempo real;
* Integração com meios de pagamento digitais.

# Recursos

* Node.js (sem frameworks)
* PostgreSQL
* Railway (deploy)
* CI/CD com GitHub Actions
* API REST
* Simuladores de gateway de pagamento e notificações

# Estimativas de Projeto

* Duração total estimada: 12 semanas
* Fase de testes: semanas 10 a 12
* Entrega final: Final de junho/2025

# Documento de Requisitos

**Requisitos Funcionais:** Gestão de mesas, pedidos, notificações, chamada de garçom, geração de conta, métodos de pagamento, relatórios, etc.

**Requisitos Não-Funcionais:**

* RNF01: Interface responsiva e simples
* RNF02: Paleta de cores tradicional italiana
* RNF03: Alta disponibilidade em horários de pico
* RNF04: Integração com notificações móveis
* RNF05: Segurança dos dados e das transações
  1. **Planejamento de Testes**
     1. **Plano de Testes**
        1. **Introdução**

Este plano de testes tem como objetivo garantir que o sistema **Easy Queue** esteja em conformidade com os requisitos estabelecidos no projeto da Mooca’s Pizzas, tanto funcionais quanto não-funcionais. Os testes serão conduzidos em ambiente de staging simulando o uso real do sistema.

# Escopo

Testes manuais e automatizados focando:

* Gestão da fila e mesas
* Integração cozinha-garçom-cliente
* Painel da cozinha
* Sistema de pedidos
* Integrações (pagamento e notificações)

# Objetivos

* Verificar o correto funcionamento dos fluxos
* Garantir usabilidade e performance
* Simular operações reais em ambiente staging

# Requisitos a Serem Testados

* Todos os RFs listados (RF01 a RF11)
* RNFs críticos: RNF01, RNF03, RNF05

# Estratégias, Tipos de Testes e Ferramentas

* **Estratégia:** Testes baseados em casos de uso
* **Tipos de Teste:** Funcional, Regressão, Usabilidade, Integração
* **Ferramentas:** Jest (unit), Postman (API), Cypress (E2E)

# Recursos a Serem Empregados

* Equipe de QA
* Railway (ambientes de staging)
* Mock de APIs de terceiros
* Banco de dados com dados simulados

# Casos de Teste

**Tabela 3 – Casos de Teste**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Caso de Teste | Descrição | Requisitos Cobertos |
| CT01 | Acesso via QR Code | Cliente acessa fila por QR | RF03 |
| CT02 | Notificação de mesa | Cliente recebe mesa disponível | RF02 |
| CT03 | Pedido por garçom | Pedido é registrado via tablet | RF06 |
| CT04 | Chamada de garçom | Cliente chama via botão | RF08 |
| CT05 | Pagamento via QR | Cliente finaliza conta via QR | RF10 |

# Roteiro de Testes

**Exemplo: Roteiro do Caso de Teste CT01 – Acesso via QR Code**

* **Pré-condição:** Cliente possui celular com câmera e acesso à internet.

# Passos:

* 1. Cliente aponta o celular para QR Code da entrada.
  2. Sistema exibe interface de cadastro.
  3. Cliente preenche nome e número de pessoas.
  4. Confirma o cadastro.
* **Resultado Esperado:** Cliente é adicionado à fila com sucesso.

# Gestão de Configuração de Software

* Utilização de Git (GitHub) para controle de versões.
* Branchmain protegida, com deploy automático.
* Ambientes distintos para staging e produção.
* Controle de mudanças documentado via pull requests e GitHub Issues.

# Repositório de Gestão de Configuração

* **Repositório:** <https://github.com/Raijoia/Moocas-Pizzas>
* **Pasta /tests/:** Casos de teste automatizados.
* **Pipeline GitHub Actions:** Linting, testes unitários, testes de integração e deploy.
* **Documentação:** README com instruções de setup e execuções de testes.