Plano de Teste para o Aplicativo ProdManage

1. Introdução

Este plano de teste descreve as estratégias, objetivos e abordagens que serão utilizados para garantir a qualidade do aplicativo **ProdManage**, que é voltado para a gestão de cortes de produção e monitoramento de desempenho dos funcionários. O objetivo dos testes é verificar se todas as funcionalidades principais estão funcionando corretamente e atendem aos requisitos especificados.

2. Objetivos do Teste

- Verificar se todas as funcionalidades principais do ProdManage estão funcionando conforme especificado.
- Garantir que as APIs backend do aplicativo estejam funcionando corretamente e retornando os dados esperados.
- Validar a interface e a experiência do usuário no aplicativo Flutter.
- Identificar e corrigir falhas ou problemas de desempenho nas funcionalidades críticas do aplicativo.
- Garantir a compatibilidade entre diferentes dispositivos móveis.

3. Escopo do Teste

Os testes incluirão:

- **Funcionalidades principais**: Cadastro de cortes, gestão de status de cortes, monitoramento de produção, e geração de relatórios de desempenho.
- **APIs**: Autenticação, cortes de produção, funcionários, performance, cargos e registros de operação.
- **Interface de usuário (UI/UX)**: Testes no front-end (Flutter) em diferentes dispositivos móveis.
- **Testes funcionais**: Verificação de cada funcionalidade conforme os requisitos.
- **Testes de API**: Verificação das respostas dos endpoints da API usando Ruby Cucumber e HTTParty.

4. Estratégia de Teste

4.1 Testes Funcionais

Esses testes garantem que o sistema funcione de acordo com os requisitos. Serão realizadas verificações de:

- **Cadastro de Cortes de Produção**: Criar, visualizar, editar e excluir cortes.
- **Gestão de Status de Cortes**: Atualizar status, deletar cortes.

- **Monitoramento de Produção de Funcionários**: Registro e visualização de desempenho, cálculo de metas e eficiência.
- **Relatórios de Desempenho**: Geração e visualização de relatórios semanais, quinzenais e mensais.

4.2 Testes de API

Os testes de API verificarão se:

- As requisições para criar, atualizar, e deletar recursos estão funcionando como esperado.
- Os dados retornados pelas APIs estão corretos.
- Os endpoints estão lidando com erros de forma adequada (validação, permissões, etc.).

4.3 Testes de Interface de Usuário (UI/UX)

Testes para garantir:

- Experiência consistente em diferentes dispositivos e resoluções.
- Layout correto e responsivo de todos os componentes.
- Funcionalidade adequada de todos os botões e ícones.
- Acessibilidade e usabilidade.

4.4 Testes de Desempenho

Avaliar o desempenho em termos de tempo de resposta do servidor, tempo de carregamento da página, e eficiência na renderização de dados em tempo real, especialmente nas áreas de monitoramento e relatórios.

4.5 Testes de Segurança

Testes para validar:

- Segurança de autenticação (ex.: falhas na validação de token, acesso não autorizado).
- Proteção de dados sensíveis, como dados de funcionários e produção.

5. Ambiente de Teste

- **Frontend**: Testes no Flutter (Android e iOS).
- **Backend**: APIs no NestJS.
- **Banco de Dados**: Prisma (MySQL).
- **Ferramentas**: Ruby Cucumber, HTTParty para testes de API, e testes manuais no front-end.

6. Casos de Teste

6.1 Cadastro de Cortes de Produção

- **Caso de Teste 1**: Criar um novo corte de produção com todos os campos obrigatórios preenchidos.
- **Caso de Teste 2**: Criar um corte com campos opcionais (ex.: linha2, comentário, imagem).
- **Caso de Teste 3**: Atualizar o status de um corte para "Finalizado".
- **Caso de Teste 4**: Excluir um corte de produção.

6.2 Monitoramento de Produção de Funcionários

- **Caso de Teste 1**: Registrar a produção de um funcionário com meta 100% e verificar se o cálculo de eficiência está correto.
- **Caso de Teste 2**: Registrar a produção de um funcionário que está abaixo da meta (ex.: 70% ou menos) e verificar a exibição em vermelho.
- **Caso de Teste 3**: Visualizar o relatório semanal de produção de um funcionário.

6.3 Relatórios de Desempenho

- **Caso de Teste 1**: Gerar um relatório semanal para todos os funcionários e verificar os dados apresentados.
- **Caso de Teste 2**: Gerar um relatório mensal e verificar se os dados agregados estão corretos.

6.4 APIs

- **Caso de Teste 1**: Realizar uma requisição POST para /cut-records com dados válidos e verificar a resposta.
- **Caso de Teste 2**: Realizar uma requisição GET para /performances e verificar os dados retornados.
- **Caso de Teste 3**: Realizar uma requisição DELETE para /employees/:id e verificar a exclusão do funcionário.

7. Critérios de Aceitação

- Todos os casos de teste devem passar sem falhas críticas.
- As funcionalidades devem estar funcionando corretamente em todos os dispositivos e navegadores suportados.
- O desempenho deve ser aceitável (ex.: tempo de resposta da API < 2 segundos).

8. Riscos e Mitigações

- **Risco**: Possíveis inconsistências no banco de dados durante as operações de CRUD.

    - **Mitigação**: Testar todas as operações de forma independente para garantir consistência.
- **Risco**: Falhas de compatibilidade em dispositivos com diferentes versões de sistema operacional.
    - **Mitigação**: Testar o aplicativo em diferentes versões do Android e iOS.

9. Conclusão

Esse plano de teste busca garantir que todas as funcionalidades críticas do ProdManage estejam funcionando conforme o esperado, promovendo uma experiência de usuário satisfatória e uma operação eficiente para o monitoramento e gestão da produção.