**BioDockFlow**

**Índice**

[Objetivo 1](#_pic5ul1xy9b0)

[Monitoramento do processo 1](#_45vd67808q7x)

[Fases do processo 2](#_lz6qb0yli1d0)

[Fluxo de Trabalho 4](#_4d34og8)

[Atividades 5](#_7rj3rnirxd1r)

[Descrição das Atividades para Documentação 6](#_64eksm43f0zg)

[Descrição das Atividades para Refatoração, Correção de Bugs, Novos Requisitos e Mudança de Requisitos 9](#_6l6xysslyezt)

[Descrição das Atividades para Dockerização 13](#_ktujcgm3n7at)

[Melhorias 19](#_s2sfh0tjcf3e)

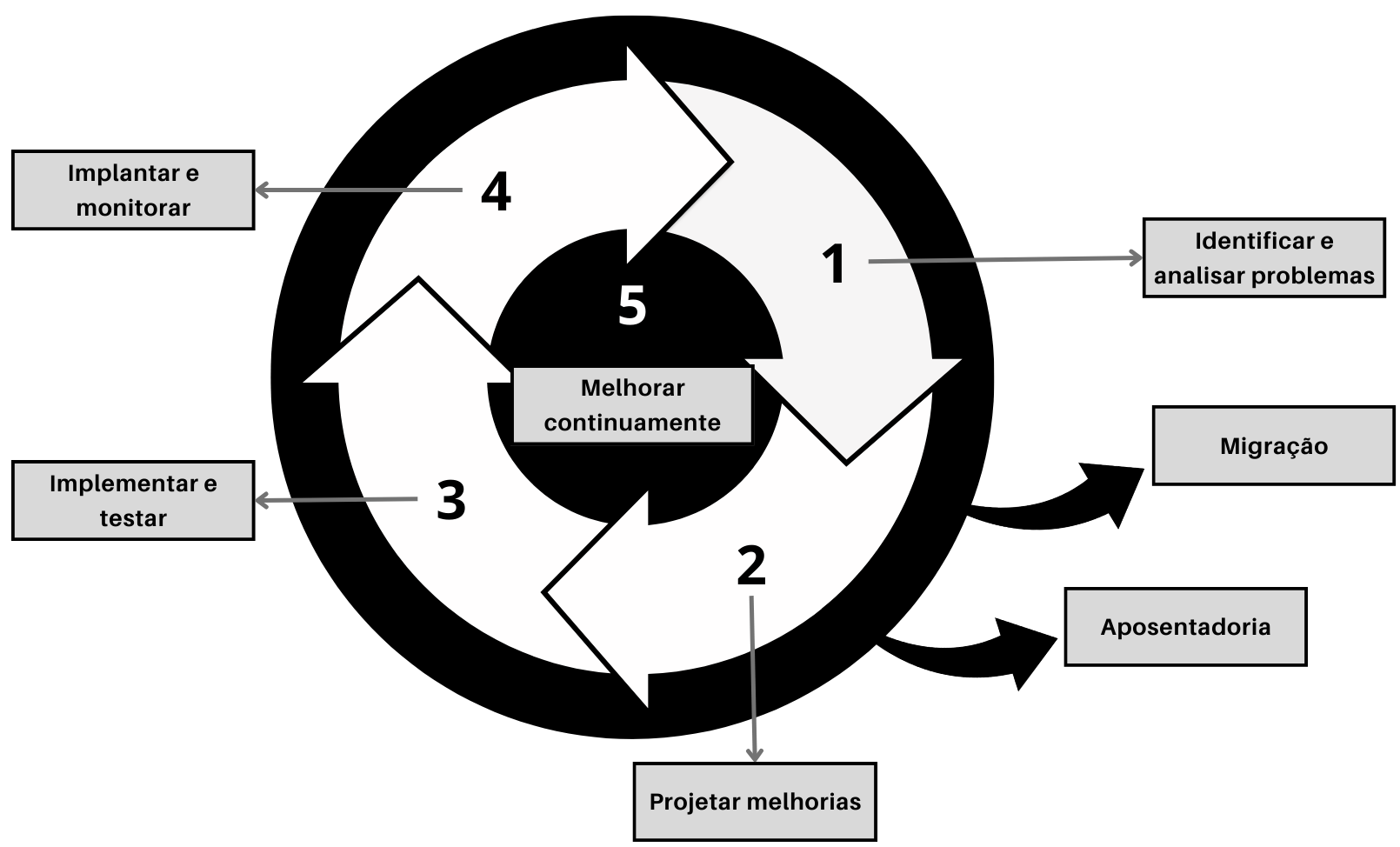
# Objetivo

Este processo tem por objetivo gerenciar todos os eventos relacionados aos requisitos do produto a ser desenvolvido pelo projeto. Dentro do escopo de gerenciar destacam-se a definição das fases, especificação das atividades que compõem essas fases e definição da ferramenta de apoio.

# Monitoramento do processo

É fundamental o uso de uma ferramenta de apoio para monitoramento das questões de manutenção (QM), validação dos atributos das QMs desde a primeira fase e acompanhamento da sua passagem correta até a conclusão. Para esse processo será utilizada a ferramenta [Jira](https://www.atlassian.com/br/software/jira). Deve ser configurado o fluxo de trabalho de acordo com as fases do processo.

# Fases do processo



A fase 1, identificar e analisar problemas, é acionada por diversas fontes, como a análise do código-fonte, a revisão da documentação, a identificação de defeitos provenientes de testes ou validações em homologação, solicitações de mudanças pelos usuários e mudanças emergentes devido a alterações no ambiente de produção. As questões levantadas serão categorizadas: problemas identificados em testes e validações podem ser classificados como correção de bugs, novas solicitações de usuários como novos requisitos ou mudança de requisitos, a análise do código-fonte pode indicar a necessidade de refatoração, a revisão da documentação pode apontar a necessidade de atualização, e questões relacionadas ao ambiente de produção podem sugerir a necessidade de conteinerização com Docker para tornar o software mais independente. No entanto, a definição da categoria só poderá ser feita com precisão após uma análise detalhada, que identificará o risco e o impacto em outros módulos do software.

A segunda fase, projetar melhorias, envolve o planejamento da solução com base na categoria definida na fase anterior. Cada categoria possui atividades de planejamento e elaboração específicas, porém, o foco principal aqui é identificar o que será feito e como será realizado. Nessa fase, são realizadas análises das condições em que a manutenção ocorrerá, bem como a avaliação de logs e mensagens que possam auxiliar na solução dos problemas. Dependendo das necessidades, a equipe de manutenção pode optar por trabalhar em questões de manutenção de forma isolada ou abordar um conjunto de questões simultaneamente, projetando-as juntas e desenvolvendo-as em paralelo, dentro de um mesmo período, para que esse conjunto forme uma entrega única.

Após a fase de projetar melhorias, a terceira fase envolve a implementação da solução, seguindo o escopo definido nas fases anteriores. Neste estágio, a solução é desenvolvida, testada e integrada ao software conforme o planejamento estabelecido. É fundamental que cada responsável por uma questão de manutenção realize testes de funcionalidade ao final de sua atividade, a fim de garantir que a implementação foi correta, ou seja, que não gerou novos pontos de manutenção em potencial e não impactou negativamente outros módulos do sistema.

A quarta fase engloba as atividades de deploy, como a atualização da versão do software em seu repositório, a preparação do ambiente de produção e a disponibilização do sistema para os usuários. No contexto do G2BC, devido à infraestrutura oferecida pelo servidor da UNEB, essa fase envolve uma colaboração entre a equipe de manutenção e uma equipe especializada em infraestrutura e redes. Juntas, elas realizam a solicitação de URLs, a configuração de certificados digitais e a configuração do proxy no servidor para tornar as aplicações acessíveis através das URLs fornecidas. Além disso, o software deve ser monitorado regularmente para garantir a otimização de desempenho e seu pleno funcionamento no ambiente de produção.

A melhoria contínua representa a última fase do processo e envolve a revisão regular e a atualização do sistema, com o objetivo de manter a otimização de desempenho. Essa fase é aplicada tanto ao software quanto ao processo em si, considerando que estratégias específicas de manutenção podem ser desenvolvidas, revisadas e ajustadas conforme as necessidades ao final de cada ciclo de manutenção. Caso um conjunto de questões de manutenção seja abordado, ao término do período estipulado para a implementação de todas as atividades, será realizada uma reunião com a equipe de manutenção para identificar os pontos de melhoria e adaptar o processo ao contexto da equipe.

# [Fluxo de Trabalho](https://drive.google.com/file/d/11AXqmqVhIrYpy7nK9dfBWzXU-WPT7NNq/view?usp=sharing)

# Atividades

| Fase:   * **1-Identificar e analisar problemas** | |
| --- | --- |
| Atividade: **Cadastrar questão de manutenção** | |
| Descrição:  Registro da questão de manutenção engatilhada por alguma das fontes. | |
| Pré-condições:   * Necessidade de manutenção evidenciada por alguma fonte/motivo | |
| Entradas:   * N/A | |
|
| Tarefas:   * Cadastrar questão de manutenção na ferramenta de apoio do processo informando uma descrição da razão/motivo, breve resumo e o tipo de manutenção a ser realizado. | |
| Ferramenta:   * Criar item no quadro do projeto na guia da primeira fase | |
| Saídas:   * QM - Questão de Manutenção registrada | |
| Observações:   * N/A | |

| Fase:   * **1-Identificar e analisar problemas** | |
| --- | --- |
| Atividade: **Refinar questão de manutenção** | |
| Descrição:  Registro da questão de manutenção engatilhada por alguma das fontes. | |
| Pré-condições:   * N/A | |
| Entradas:   * QM registrada no Jira | |
|
| Tarefas:   * Complementar a descrição da QM para incluir exatamente o que precisa ser modificado e quais módulos ou arquivos são afetados. | |
| Ferramenta:   * Completar registro da QM no Jira com detalhamento da descrição | |
| Saídas:   * Questão de Manutenção com descrição detalhada | |
| Observações:   * N/A | |

| Fase:   * **1-Identificar e analisar problemas** | |
| --- | --- |
| Atividade: **Analisar questão de manutenção** | |
| Descrição:  Analisar o impacto da QM, identificar as possíveis ações e requisitos necessários para completude na realização da manutenção. | |
| Pré-condições:   * N/A | |
| Entradas:   * QM registrada no Jira | |
|
| Tarefas:   * Avaliar impacto da mudança solicitada, analisando impacto nos requisitos, artefatos técnicos e no projeto (esforço, prazo, escopo, equipe, riscos). * Avaliar se solicitação de mudança interna impacta os requisitos do Cliente visando envolvê-lo no desenvolvimento da manutenção. * Complementar o registro da QM antes do desenvolvimento com as informações de risco, impacto, categoria (documentação, correção de bugs, dockerização, refatoração ou change request) e responsável * Estimar tempo de realização da manutenção indicando na QM a data limite para conclusão | |
| Ferramenta:   * Completar Registro da QM no Jira com os atributos de contexto * Mover QM para a guia da próxima fase | |
| Saídas:   * Questão de Manutenção com registro completo | |
| Observações:   * N/A | |

### Descrição das Atividades para Documentação

| Fase:   * 2-Projetar melhorias | |
| --- | --- |
| Atividade: **Identificar funcionalidades da aplicação** | |
| Descrição:  Mapear funcionalidades da aplicação | |
| Pré-condições:   * Manutenção categorizada como Documentação | |
| Entradas:   * Código do servidor da aplicação | |
|
| Tarefas:   * Analisar o código da API identificando as funcionalidades/métodos presentes | |
| Ferramenta:   * N/A | |
| Saídas:   * Relação das funcionalidades da aplicação | |
| Observações:   * N/A | |

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Identificar funcionalidades ausentes na documentação** | | | | |
| Descrição:  Mapear funcionalidades que precisam ser documentadas | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Relação das funcionalidades da aplicação * Documentação da aplicação | | | | |
|
| Tarefas:   * Comparar a documentação atual com as funcionalidades identificadas anteriormente | | | | |
| Ferramenta:   * N/A | | | | |
| Saídas:   * Relação das funcionalidades ausentes na documentação | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Identificar instruções de usabilidade ausentes na documentação** | | | | |
| Descrição:  Identificação de instruções de usabilidade e configuração do ambiente de desenvolvimento que estão ausentes na documentação | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Documentação da aplicação | | | | |
| Tarefas:   * Seguir as instruções presentes na documentação atual para identificar passos ausentes | | | | |
| Ferramenta:   * Mover QM no Jira para próxima fase | | | | |
| Saídas:   * Lista de instruções de usabilidade e configuração a serem incluídas | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Atualizar documentação** | | | | |
| Descrição:  Implementação da atualização da documentação com adição e atualização das funcionalidades necessárias bem como instruções de usabilidade e configuração do ambiente de desenvolvimento | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Documentação da aplicação * Lista de instruções de usabilidade e configuração a serem incluídas * Relação das funcionalidades ausentes na documentação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Documentação atualizada | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Validar documentação** | | | | |
| Descrição:  Validação da documentação seguindo as informações presentes na mesma após atualização | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Documentação atualizada | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * Mover QM no Jira para próxima fase | | | | |
| Saídas:   * Documentação validada | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 4-Implantar e monitorar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Atualizar documentação** | | | | |
| Descrição:  Atualização da documentação no repositório da aplicação na branch principal | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Documentação validada | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * Mover QM no Jira para guia de Pronto * Finalizar QM | | | | |
| Saídas:   * Repositório atualizado com nova documentação | | | | |
| Observações:   * N/A | | | | |

### Descrição das Atividades para Refatoração, Correção de Bugs, Novos Requisitos e Mudança de Requisitos

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Análisar Logs** | | | | |
| Descrição:  Rastreio da origem e identificação do bug através da análise de logs da aplicação e mensagens de erro ao executar a funcionalidade | | | | |
| Pré-condições:   * Manutenção categorizada como Correção de Bugs | | | | |
| Entradas:   * Logs da aplicação * Código da aplicação | | | | |
| Tarefas:   * Identificar origem do bug para correção | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Arquivo/Módulo que precisa de correção identificado | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Identificar módulos afetados** | | | | |
| Descrição:  Identificação dos módulos do sistema que são/serão afetados | | | | |
| Pré-condições:   * Manutenção categorizada como Refatoração, Correção de Bugs, Novos Requisitos ou Mudança de Requisitos | | | | |
| Entradas:   * Código da aplicação | | | | |
| Tarefas:   * Mapear os arquivos que precisam ser alterados | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Relação dos arquivos que precisam ser alterados | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Identificar nível de dependência entre os módulos** | | | | |
| Descrição:  Identificação das dependências do alvo da alteração com os demais módulos para maior clareza do risco e impacto da manutenção | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Relação dos arquivos que precisam ser alterados * Código da aplicação | | | | |
| Tarefas:   * Identificar o grau de dependência do(s) arquivo(s) alvo da manutenção com demais módulos | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Relação dos arquivos que precisam ser alterados com níveis de impacto | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Priorizar Implementação** | | | | |
| Descrição:  Priorização das tarefas com base no nível de dependência entre o arquivo alvo e os demais módulos partindo dos mais acoplados para os mais simples | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Relação dos arquivos que precisam ser alterados com níveis de impacto | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * Mover QM no Jira para próxima fase | | | | |
| Saídas:   * Relação dos arquivos que precisam ser alterados priorizada por nível | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Iniciar branch** | | | | |
| Descrição:  Iniciar branch no repositório da aplicação para implementação da solução | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Repositório da aplicação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Branch criada no repositório | | | | |
| Observações:   * Seguir padrão de nomenclatura de branch: "natureza da implementação/objetivo da atividade" Ex: feature/autocomplete; fix/userRegister; refact/getFungis | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Desenvolver solução** | | | | |
| Descrição:  Implementar solução da manutenção na branch criada anteriormente | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Branch criada no repositório * Código da aplicação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Código da aplicação atualizado com as mudanças | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Realizar testes** | | | | |
| Descrição:  Testar a funcionalidade alvo da manutenção bem como os módulos/arquivos que sofreram modificações para garantir a suficiência da implementação | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * N/A | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * Mover QM para próxima fase no Jira | | | | |
| Saídas:   * Relação dos testes de funcionalidades executados | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Iniciar Pull Request** | | | | |
| Descrição:  Testar a funcionalidade alvo da manutenção bem como os módulos/arquivos que sofreram modificações para garantir a suficiência da implementação | | | | |
| Pré-condições:   * Equipe de manutenção adotou Code Review (revisão de código) | | | | |
| Entradas:   * Branch criada para execução da QM | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * Completar registro da QM com atributo do link/url do Pull Request criado | | | | |
| Saídas:   * Pull Request criado no repositório do projeto | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Realizar Code Review** | | | | |
| Descrição:  Realizar a revisão de código no pull request da QM | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Pull Request criado no repositório do projeto * Link do Pull Request criado | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Pull Request aprovado | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Atualizar aplicação em desenvolvimento** | | | | |
| Descrição:  Realizar o merge das alterações realizadas e aprovadas no pull request à branch de desenvolvimento | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Pull Request aprovado | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Código da aplicação atualizado na branch de desenvolvimento | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 4-Implantar e monitorar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Atualizar aplicação** | | | | |
| Descrição:  Realizar o merge da branch de desenvolvimento ao final da fase de implementação e testes com a versão de produção da aplicação no repositório | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Relação dos testes de funcionalidades executados * Código da aplicação atualizado com as mudanças | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Código da aplicação atualizado na branch de produção | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 4-Implantar e monitorar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Publicar alterações** | | | | |
| Descrição:  Publicar aplicação atualizada em ambiente de produção | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Código da aplicação atualizado na branch de produção | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Nova versão do sistema em execução com as correções implementadas | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 4-Implantar e monitorar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Monitorar** | | | | |
| Descrição:  Monitorar aplicação atualizada após publicação para observar e garantir funcionamento correto da solução implementada no cenário de produção | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Nova versão do sistema em execução com as correções implementadas | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * Mover QM no Jira para guia de Pronto * Finalizar QM | | | | |
| Saídas:   * Nova versão do sistema em execução com as correções implementadas | | | | |
| Observações:   * N/A | | | | |

### Descrição das Atividades para Dockerização

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Análise de dependências** | | | | |
| Descrição:  Analisar e entender como a tecnologia de implementação da aplicação lida com as dependências e como as mesmas podem ser instaladas no ambiente de contêiner | | | | |
| Pré-condições:   * Manutenção categorizada como Docker | | | | |
| Entradas:   * Código da aplicação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Instruções para obter as dependências | | | | |
| Observações:   * N/A | | | | |

# 

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Análise de serviços** | | | | |
| Descrição:  Analisar os serviços que precisam ser acoplados e/ou isolados em seus próprios contêineres | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Código da aplicação * Arquitetura da aplicação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Relação de serviços presentes na aplicação | | | | |
| Observações:   * N/A | | | | |

# 

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Análise de volumes** | | | | |
| Descrição:  Analisar e entender quais volumes precisam ser criados para garantir persistência dos dados e acesso da aplicação a arquivos gerados por algum dos serviços | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Código da aplicação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Relação dos diretórios e volumes a serem mapeados | | | | |
| Observações:   * N/A | | | | |

# 

| Fase:   * 2-Projetar melhorias | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Definição do cenário** | | | | |
| Descrição:  Definir o cenário da conteinerização com base na análise dos serviços   * Cenário 1:Interação entre diversos contêineres; Cada serviço da aplicação isolado em um contêiner. * Cenário 2: Aplicação completa pode ser replicada em um único contêiner. | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Código da aplicação * Relação de serviços presentes na aplicação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * Mover QM para próxima fase no Jira | | | | |
| Saídas:   * Arquitetura da aplicação para contêineres | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Escrita Dockerfile** | | | | |
| Descrição:  Escrita do arquivo Dockerfile, se necessário, com base na análise das dependências realizada | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Arquitetura da aplicação para contêineres * Instruções para obter as dependências | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Dockerfile | | | | |
| Observações:   * N/A | | | | |

# 

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Escrita compose** | | | | |
| Descrição:  Escrita do arquivo docker-compose, se necessário, com base na análise de serviços e de volumes | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Arquitetura da aplicação para contêineres * Relação dos diretórios e volumes a serem mapeados * Relação de serviços presentes na aplicação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * docker-compose.yml | | | | |
| Observações:   * N/A | | | | |

# 

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Build do contêiner** | | | | |
| Descrição:  Realizar build do contêiner para atestar a execução correta da aplicação e identificar possíveis erros como problemas na versão da imagem ou falhas no gerenciamento das dependências | | | | |
| Pré-condições:   * N/A | | | | |
| Entradas:   * Dockerfile * docker-compose | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Contêiner(es) da aplicação *buildados* | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 3-Implementar e testar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Testes** | | | | |
| Descrição:  Realizar testes das funcionalidades da aplicação em contêiner para garantir o funcionamento e interação corretos entre os serviços | | | | |
| Pré-condições:   * Contêiner em execução | | | | |
| Entradas:   * Contêiner(es) da aplicação | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * Relação dos testes de funcionalidades executados | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 4-Implantar e monitorar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Configurar rede** | | | | |
| Descrição:  Adicionar configuração ao contêiner para execução em rede docker criada no ambiente de produção | | | | |
| Pré-condições:   * Aplicação disponível no ambiente de produção | | | | |
| Entradas:   * docker-compose.yml | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * N/A | | | | |
| Saídas:   * docker-compose.yml atualizado com rede docker do ambiente | | | | |
| Observações:   * N/A | | | | |

| Fase:   * 4-Implantar e monitorar | | | | |
| --- | --- | --- | --- | --- |
| Atividade: **Configurar proxy** | | | | |
| Descrição:  Configurar acesso a aplicação por proxy no ambiente de produção | | | | |
| Pré-condições:   * Aplicação configurada na rede docker adequada | | | | |
| Entradas:   * docker-compose.yml ajustado ao ambiente | | | | |
| Tarefas:   * N/A | | | | |
| Ferramentas:   * Mover QM no Jira para guia de Pronto * Finalizar QM | | | | |
| Saídas:   * Arquivo de configuração do proxy * Aplicação acessível externamente via url | | | | |
| Observações:   * N/A | | | | |

# 

# 

# Melhorias

A última fase do processo compreende a melhoria contínua, ao final do processo, ou seja, a finalização de uma questão de manutenção deve-se realizar uma revisão das estratégias e etapas para que o processo seja adaptado e melhorado para as próximas iterações de manutenção. Em um contexto de equipe de manutenção podem ser trabalhados conjuntos, agrupamentos de QMs dentro de um período definido, dessa forma a revisão da adoção e implementação do processo ao longo do período pode ser trabalhada em uma reunião a fim de adequar o processo ao contexto e condições de trabalho da equipe.