**Entrega 4:**

**Introdução**

Este documento apresenta a proposta de gerenciamento de projeto, abordando a estimativa de tempo, custo, e análise de riscos, além de discutir as práticas de qualidade de software, incluindo o uso de métricas e revisões de código. O objetivo é assegurar uma gestão eficiente e a entrega de um sistema robusto e confiável.

**1. Proposta de Gerenciamento de Projeto**

**1.1. Estimativa de Tempo**

O projeto é dividido em fases, cada uma com marcos importantes para facilitar o acompanhamento do progresso. Estimamos um período total de dois meses, conforme a seguinte divisão:

1. **Planejamento e Levantamento de Requisitos** (2 semanas)
   * Definição de requisitos funcionais, não funcionais e de domínio.
   * Esboço inicial da API e dashboards.
2. **Desenvolvimento Inicial** (3 semanas)
   * Implementação da API e integração com banco de dados.
   * Criação dos primeiros dashboards interativos para análise de interrupções e reclamações.
3. **Teste e Validação** (2 semanas)
   * Execução de testes unitários e de integração.
   * Correção de erros identificados durante os testes.
4. **Otimização e Implementação de Práticas de Qualidade** (1 semana)
   * Revisão de código e otimização de desempenho.
   * Documentação do sistema e preparação para entrega.
5. **Entrega Final e Manutenção Inicial** (1 semana)
   * Entrega do sistema completo.
   * Suporte inicial e planejamento de manutenção contínua.

**1.2. Estimativa de Custo**

O custo do projeto é calculado com base no tempo alocado para cada fase e na remuneração das equipes envolvidas. Considerando uma equipe de 3 desenvolvedores e 1 analista de qualidade, com uma taxa média de $40/hora, a estimativa de custo é a seguinte:

1. **Equipe de Desenvolvimento**: 3 desenvolvedores x $40/h x 8h/dia x 40 dias = $38,400
2. **Equipe de Análise de Qualidade**: 1 analista x $40/h x 8h/dia x 40 dias = $12,800
3. **Infraestrutura e Ferramentas**: Licenciamento de software e serviços de nuvem (estimativa): $5,000

**Custo Total Estimado: $56,200**

**1.3. Análise de Risco**

1. **Risco de Escopo**: Requisitos não claramente definidos ou mudanças frequentes podem afetar o prazo. *Mitigação*: Revisões regulares com o cliente.
2. **Risco Técnico**: Integração da API com a base de dados pode ter problemas de desempenho. *Mitigação*: Prototipagem e testes iniciais.
3. **Risco de Recursos**: A indisponibilidade da equipe pode atrasar o projeto. *Mitigação*: Treinamento cruzado e planejamento de contingência.
4. **Risco de Qualidade**: Bugs não identificados em tempo hábil podem afetar a entrega. *Mitigação*: Implementação de testes automatizados e revisões de código contínuas.

**2. Práticas de Qualidade de Software**

**2.1. Métricas de Software**

Para garantir a qualidade do software desenvolvido, as seguintes métricas serão monitoradas:

1. **Cobertura de Testes**: Medição da porcentagem de código coberto por testes automatizados. Meta: 80% de cobertura.
2. **Tempo de Resposta da API**: Tempo médio de resposta da API. Meta: < 300 ms.

**2.2. Revisões de Código**

A revisão de código será realizada regularmente para garantir a qualidade e a manutenção do código. O processo incluirá:

1. **Pull Requests (PRs)**: Antes de qualquer código ser mesclado na branch principal, será aberto um PR.
2. **Checklist de Revisão**: Um checklist padrão será usado para verificar:
   * Legibilidade e clareza do código.
   * Conformidade com os padrões de codificação definidos.
   * Implementação de boas práticas de segurança.
3. **Feedback Estrutural**: Revisores fornecerão feedback construtivo e sugestões de melhorias.

**2.3. Automação de Testes**

Para manter a qualidade, testes automatizados serão escritos para verificar funcionalidades críticas. Utilizando frameworks como pytest, garantiremos que cada atualização não introduza novos erros.

**3. Aplicação de Práticas DevOps**

Para manter o ciclo de desenvolvimento ágil e eficiente, aplicaremos práticas DevOps, mesmo em uma abordagem simplificada:

* **Integração Contínua (CI)**: Configuração de pipelines com GitHub Actions para execução automática de testes.
* **Entrega Contínua (CD)**: Preparação de scripts para facilitar a implantação no ambiente de produção.
* **Monitoramento e Logging**: Configuração de logs e alertas básicos para identificar e corrigir problemas rapidamente.

**Conclusão**

Este documento fornece um plano abrangente de gerenciamento de projeto, estimativa de tempo, custo e risco, juntamente com práticas de qualidade de software. A aplicação de métricas, revisões de código e automação de testes garante que o projeto atenda aos padrões desejados. A integração de práticas DevOps, mesmo que de forma básica, aumenta a eficiência e reduz a possibilidade de falhas na entrega do sistema.