Olá, estudante.

A seguir, você dará continuidade ao desenvolvimento da sua solução através dos campos específicos para a resolução dos 3 desafios propostos, lembrando que eles se complementam.

Nome: Larissa Anielly Freitas Moura
RGM: 32091541

Documentação:

Revisite a documentação do projeto desenvolvido na PIT I, faça as atualizações e melhorias necessárias de acordo com os pontos explicitados no material teórico. Ao terminar os ajustes, suba a documentação em um repositório GIT.

***Link* do repositório:**

- https://github.com/Larissa-Anielly/Projeto-Integrador-Transdisciplinar-de-Engenharia-de-Software-II (link do repositorio no github)
- https://pitengenharia-de-softwar.bubbleapps.io/version-test/ (link do projeto)
- https://bubble.io/page?id=pitengenharia-de-softwar&tab=Design&name=index&type=page (link do editor)
- https://bubble.io/page?id=pitengenharia-de-softwar&tab=Data&name=index&type=page&type_id=pedidos (link do banco de dados)

Codificação:

Na Tabela a seguir insira as informações referentes ao desenvolvimento do código do *front-end* e *back-end*.

Linguagem do *Back-end*	**Backend Nativo do Bubble**
Banco de Dados	**Banco de Dados Nativo do Bubble**
Hospedagem	**Hospedagem Gratuita e Nativa do Bubble**
Plataforma	**Bubble**
Modo de Codificação	( ) Tradicional ( ) *Low-code* (x) *No-code*

***Link* do repositório no GitHub com os códigos abertos**	• https://github.com/Larissa-Anielly/Projeto-Integrador-Transdisciplinar-de-Engenharia-de-Software-II (link do repositorio no github) • https://bubble.io/page?id=pitengenharia-de-softwar&tab=Design&name=index&type=page (link do editor) • https://bubble.io/page?id=pitengenharia-de-softwar&tab=Data&name=index&type=page&type_id=pedidos (link do banco de dados)
***Link* da solução em funcionamento**	• https://pitengenharia-de-softwar.bubbleapps.io/version-test/ (link do projeto)

Solução do problema 1:

Diante da situação problema 1, a falta de tempo para o desenvolvimento tradicional, eu sugeriria o uso de uma plataforma no-code, como o Bubble, para criar rapidamente o módulo solicitado.

Essas plataformas permitem construir aplicações web funcionais sem necessidade de programação, utilizando recursos visuais e lógicos que facilitam o desenvolvimento. Assim, a equipe (mesmo sem experiência técnica) poderia montar uma interface para visualizar, editar e validar pedidos dos vendedores antes que se tornem ordens de serviço.

Além disso, esse tipo de ferramenta oferece integração com bancos de dados e APIs, permitindo conectar o módulo ao sistema já existente. Dessa forma, o problema seria resolvido de forma ágil, colaborativa e com baixo custo de implementação, sem depender da fila de demandas da equipe de TI.

Para iniciar o desenvolvimento desse módulo na plataforma no-code, bubble, precisamos gerar um plano de desenvolvimento:

Passo 1: Definir o escopo mínimo

- Objetivo:

    - Visualizar pedidos dos vendedores;
    - permitir editar / corrigir antes de virar ordem de serviço;
    - botão para “confirmar / transformar em OS”.
- Decidir quais campos são obrigatórios
    - (ex.: id_pedido, vendedor, cliente, itens[], quantidade, preço_unitário, total, status, observações, data_pedido).
- Resultado esperado:
    - lista dos pedidos
    - tela para edição
    - ação “Enviar para OS”.

Passo 2:Escolher ferramenta

- Bubble. Essa ferramenta full-stack no-code, permite integrações de diversas API’s como banco de dados externos, entre outros

Passo 3: Mapear dados e integração

- Liste as tabelas/endpoint existentes que contêm pedidos.
- https://bubble.io/page?id=pitengenharia-de-softwar&tab=Data&name=index&type=page&type_id=pedidos (link do banco de dados)

Passo 5: Construção do módulo no Bubble

- Criar página “Lista de Pedidos”: tabela com filtros (por vendedor, data, status) + botão “Editar / Validar”.
- Criar página “Edição de Pedido”: campos editáveis, validações inline e área de logs/alterações.
- Botão “Salvar rascunho” (não altera OS), botão “Confirmar > Transformar em OS” que chama endpoint.
- Registrar um log de auditoria (quem alterou, quando, antes/depois)
- https://bubble.io/page?id=pitengenharia-de-softwar&tab=Design&name=index&type=page (link do editor)

Passo 6: Regras de negócio e validações

- Validar soma de itens = total; validar estoque disponível (se aplicável), campos obrigatórios, limites de preço/desconto.
- Implementar permissões: só vendedores/operadores autorizados podem editar; aprovadores podem converter em OS.

Passo 7: Testes rápidos

- Teste fluxo: buscar pedido > editar > salvar > confirmar > verificar criação da OS no sistema destino.

- Teste casos inválidos: edição inválida, conflito de edição concorrente, perda de conexão.
- Teste revert: implementar “desfazer” ou procedimento de correção se algo virar OS indevidamente.

Passo 8: Documentação mínima & treinamento

- Documento com: como acessar, papéis/perm., checklist de validação e passo a passo de transformar em OS.
- https://github.com/Larissa-Anielly/Projeto-Integrador-Transdisciplinar-de-Engenharia-de-Software-II (link do repositorio no github)
- Rápido treinamento de 30–60 min para os responsáveis (demonstração prática do fluxo).

Passo 9: Deploy e monitoramento

- Lançar em ambiente de produção.
- Monitorar erros e comportamento nas primeiras horas/dias; coletar feedback e iterar.

Seguindo esse cronograma, a implementação do módulo ocorrerá de forma rápida e eficiente.

- https://pitengenharia-de-softwar.bubbleapps.io/version-test/ (link do projeto)