UNIVERSIDADE PAULISTA

ICET - INSTITUTO DE CIÊNCIAS EXATAS E TECNOLOGIA

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

PROJETO INTEGRADO MULTIDISCIPLINAR

PIM IV

Desenvolvimento de um sistema integrado para o controle de operações em uma fazenda urbana, visando apoiar uma startup inovadora na área de segurança alimentar

Nome	R.A
Carlos Henrique Duarte da Silva	G81BHC5
João Gabriel Alves dos Santos	R0167J3
Lucas Navimar de Toledo	R009JD0
Luis Henrique Vieira Barros	R024768
Matheus Guilherme Pacheco Courbassier	R026272
Ruan Pablo Galdino Dias Bento	G86JCF-1

SÃO JOSÉ DOS CAMPOS – SP

NOVEMBRO /2024

Carlos Henrique Duarte da Silva	G81BHC5
João Gabriel Alves dos Santos	R0167J3
Lucas Navimar de Toledo	R009JD0
Luis Henrique Vieira Barros	R024768
Matheus Guilherme Pacheco Courbassier	R026272
Ruan Pablo Galdino Dias Bento	G86JCF-1

Desenvolvimento de um sistema integrado para o controle de operações em uma fazenda urbana, visando apoiar uma startup inovadora na área de segurança alimentar

Projeto Integrado Multidisciplinar (PIM) desenvolvido como exigência parcial dos requisitos obrigatórios à aprovação semestral no Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da UNIP (Universidade Paulista), orientado pelo corpo docente do curso.

São José dos Campos – SP

NOVEMBRO 2024

RESUMO

O projeto foi continuado com base em um sistema de gestão de uma startup de fazenda urbana movida pela necessidade mundial da crescente demanda por produtos alimentícios, aumentado ainda mais nos últimos anos devido ao aumento da população. A startup se responsabilizou pelo compromisso de produzir alimentos orgânicos e saudáveis. Foi implantado três novos sistemas que foram web, mobile e desktop e o objetivo inicial foi levantar as regras e processos de negócios, fazer uma estimativa de viabilidade, criar valores profissionais da empresa e desenvolver um sistema de gerenciamento para a fazenda urbana. Foi feito o levantamento de requisitos, divisão entre requisitos funcionais e não funcionais, elaboração dos diagramas UML e diagrama Pert cpm, implementação SQL, planilha de testes. Foi adotado a metodologia Scrum para o desenvolvimento do projeto, tendo dividido a equipe em um scrum master, um product owner e quatro integrantes da equipe de desenvolvimento. Seguido pela organização do projeto, foi feito o desenvolvimento das três aplicações mencionadas, aplicações cujo foram utilizadas técnicas diferentes entre web e desktop, assim como reaproveitamento de código entre desktop e mobile. As linguagens utilizadas foram python para desktop e mobile e html, css e Javascript para a parte web, as linguagens foram escolhidas tendo como requisito a orientação a objetos.

SUMÁRIO

1. INTRODUÇÃO

Com o crescimento populacional e a urbanização acelerada, são colocados grandes desafios à segurança alimentar, que exigem soluções inovadoras para garantir a produção alimentar saudável e sustentável. Neste contexto, a agricultura urbana é uma alternativa viável para satisfazer a procura de alimentos frescos e reduzir o impacto ambiental do transporte e armazenamento de produtos agrícolas. As startups focadas na segurança alimentar desempenham um papel vital neste contexto, impulsionando a adoção de tecnologias avançadas e práticas sustentáveis.

De acordo com o relatório da ONU, A população mundial deve crescer em 2 bilhões de pessoas nos próximos 30 anos, passando dos atuais 7,7 bilhões de indivíduos para 9,7 bilhões em 2050. Isso traz uma grande responsabilidade para suprir a demanda alimentar da população, com qualidade e segurança. Porém, com a redução das áreas agricultáveis e a urgência em revermos velhas práticas de exploração de recursos, o desafio está em produzirmos alimentos em grande escala, mas de forma sustentável.

Segundo a empresa privada BRK (2023), A resposta para esse problema está no uso da tecnologia e da automação de processos extraordinariamente bem empregados em fazendas urbanas. A principal ideia é utilizar a tecnologia em diversos países, como o Brasil, para diminuir os gastos com transporta e uso de agrotóxicos, além de facilitar o manejo dos funcionários e logística de uma Fazenda Urbana.

Seguindo essas informações esse trabalho tem como objetivo o desenvolvimento de um sistema integrado, multiplataforma (mobile, web e desktop), para o controle das operações de uma fazenda urbana. O sistema será projetado para gerenciar de forma eficiente os processos da fazenda, incluindo o controle de clientes, fornecedores, tarefas operacionais e a produção agrícola. A plataforma visa otimizar o uso de recursos, melhorar a gestão de processos e garantir a rastreabilidade dos alimentos, fatores essenciais para atender à crescente demanda por alimentos de qualidade e sustentáveis.

A proposta desse sistema tem como foco aumentar a produtividade e a transparência das operações na fazenda urbana, além de apoiar a startup na melhoria da eficiência operacional e no monitoramento em tempo real. O estudo se delimita ao desenvolvimento dessa solução tecnológica, com ênfase na integração de funcionalidades para gerenciar de forma eficiente as atividades da fazenda e apoiar a tomada de decisões estratégicas. A

acessibilidade em diferentes plataformas torna a solução flexível e adaptável às necessidades dos usuários, proporcionando um ambiente de gestão ágil e eficaz

1.1. OBJETIVO GERAL

O objetivo geral deste projeto foi definido como a criação de mais sistema como mobile, desktop e web e banco de dados SQL para armazenamento e gerenciamento dos dados da fazenda. Sistemas que foram criados para auxiliar o usuário para uma melhor experiência e armazenar seus dados.

1.1.1. Objetivos Específicos

Com o propósito de atingir o objetivo geral proposto, serão considerados os seguintes objetivos específicos:

- Identificar as Regras de Negócio da fazenda urbana;
- Criar uma lista de requisitos funcionais e não funcionais;
- Criar um sistema mobile para os clientes usarem de seus próprios celulares;
- Criação de um sistema desktop;
- Criar um sistema web para que os usuários acessem diretamente da internet;
- Implementar o Banco de Dados com os outros sistemas;
- Utilizar a metodologia ágeis para as sprints durante a semana para analisar o andamento do projeto da fazenda.
- Utilizar UML para aprimorar os diagramas anteriores.
- Utilizamos um Diagrama Pert para gerenciamento de projetos para planejar, programar e controlar tarefas.
- Desenvolver CRUD (Create, Read, Update, Delete).

- Criar o Diagrama de implantação.
- Elaborar planilha de testes.
- Desenvolver o Diagrama de Sequência baseando-se no Caso de Uso;Com o propósito de atingir o objetivo geral proposto, serão considerados os seguintes objetivos específicos:
- Identificar as Regras de Negócio da fazenda urbana;
- Criar uma lista de requisitos funcionais e não funcionais;
- Criar um sistema mobile para os clientes usarem de seus próprios celulares;
- Criação de um sistema desktop;
- Criar um sistema web para que os usuários acessem diretamente da internet;
- Implementar o Banco de Dados com os outros sistemas;
- Utilizar a metodologia ágeis para as sprints durante a semana para analisar o andamento do projeto da fazenda.
- Utilizar UML para aprimorar os diagramas anteriores.
- Utilizamos um Diagrama Pert para gerenciamento de projetos para planejar, programar e controlar tarefas.
- Desenvolver CRUD (Create, Read, Uptade, Delete).
- Criar o Diagrama de implantação.
- Elaborar planilha de testes.
- Desenvolver o Diagrama de Sequência baseando-se no Caso de Uso.

2. Gerenciamento do Projeto

A fim de concluir a atividade atribuída ao dev team, é preciso ter uma boa gerência de equipe para sucesso da conclusão do projeto em tempo hábil. Por isso foi atribuída ao processo de desenvolvimento a metodologia agile.

De acordo com o site Project Builder (2021), o gerenciamento de projetos de software baseado na metodologia do PMI (Project Management Institute) segue as melhores práticas descritas no PMBOK (Project Management Body of Knowledge). Ele organiza o ciclo de vida dos projetos em cinco grupos de processos interligados: **Iniciação, Planejamento, Execução, Monitoramento e Controle, e Encerramento**. Cada etapa é composta por práticas específicas para garantir o sucesso na entrega de projetos, mesmo em contextos complexos e dinâmicos.

2.1 Elementos principais do PMBOK:

Segundo João Santos (2024), O PMBOK não é uma metodologia em si, mas sim um guia que detalha dez áreas de conhecimento críticas para o gerenciamento de projetos, cada uma contendo processos, ferramentas e técnicas essenciais para a gestão eficaz de projetos.

“O PMBOK detalha claramente os processos necessários para a abertura do projeto, incluindo a criação do termo de abertura do projeto, que é fundamental para definir os objetivos e o escopo inicial.” (João Santos, 2024).

1. **Áreas de conhecimento**: O PMBOK organiza os processos em 10 áreas, como gestão de escopo, tempo, custos, qualidade, recursos e riscos. Essas áreas ajudam a estruturar o planejamento e execução do projeto para alcançar resultados previsíveis e de alta qualidade.

2. **Gestão de stakeholders**: Enfatiza a identificação e o engajamento das partes interessadas, crucial para evitar conflitos e alinhar expectativas.

3. **Controle e monitoramento contínuos**: A metodologia promove ajustes frequentes com base em métricas como indicadores de desempenho (KPIs), garantindo que o projeto atenda às metas de prazo, orçamento e qualidade.

Vale mencionar que o PMI também é conhecido por suas certificações, como a PMP (Project Management Professional), que é amplamente reconhecida e proporciona maior credibilidade ao profissional na área de gestão de projetos.

2.2. Iniciação e Planejamento

A fase de iniciação é onde tudo começou. Aqui, o projeto foi oficialmente idealizado e começou os primeiros passos para tomar forma. O primeiro passo foi identificar as partes interessadas, ou seja, as pessoas e grupos que têm interesse no sucesso do projeto. Para a fazenda urbana TerrasMil, isso inclui o cliente (que irá usar o produto), a equipe de desenvolvimento (responsável pela criação do software) e os avaliadores do produto final.

A importância dessa fase é garantir que todas as partes interessadas tivessem uma visão comum sobre o projeto e porque ele é necessário. A clareza nessa etapa ajuda a evitar problemas no futuro e garantir que todos estivessem alinhados com as expectativas do projeto.

O planejamento foi uma das etapas mais cruciais no gerenciamento do projeto. Aqui a equipe alinhou exatamente o que seria definido, como seria feito e quando seria feito. Entre os principais processos dessa fase, podemos destacar:

- **Definição do escopo**: O que o software deveria fazer e quais funcionalidades são essenciais quais são os requisitos e as expectativas dos usuários. No caso do TerrasMil o escopo foi centrado em criar um app Web, Mobile e para Desktop, seguindo os requisitos apresentados anteriormente.

- **Planejamento do cronograma:** O cronograma foi definido de acordo com quanto tempo os membros do grupo levariam para terminar certas tarefas.

- **Identificação das atividades**: As tarefas necessárias para concluir o projeto foi a equipe listar todas as etapas, desde o design até os testes finais.

    - **Estimativa de duração**: Foi levantada uma estimativa e chegamos a um resultado, isso ajudou a planejar o trabalho e a alocar os recursos de forma eficiente.

    - **Criação do cronograma**: Com base nas estimativas, o cronograma do projeto foi montado, incluindo datas de entrega e marcos importantes.

- **Estimativa de custos**: Além dos salários dos desenvolvedores, é necessário considerar licenças de software, infraestrutura, testes e outros gastos.

- **Gestão dos recursos humanos**: Nessa etapa, foi importante identificar as habilidades necessárias para o projeto e garantir que a equipe tinha as competências necessárias.

- **Planejamento da comunicação**: Como as informações foram compartilhadas entre todos os envolvidos. Foi essencial estabelecer um plano de comunicação claro, com frequência de reuniões, relatórios de progresso e canais de comunicação (e-mail, ferramentas como Slack, ou software de gestão de projetos). Neste caso foi usado principalmente Whatsapp, Discord e email para envio de arquivos e fichários.

- **Gestão de riscos**: Foi analisado os possíveis obstáculos que surgiriam ao longo do caminho. Foi antecipado problemas de desenvolvimento e estratégias de mitigação que pode fazer toda a diferença. Caso um risco aconteça, a equipe se mobilizará para cobrir a parte não finalizada de quem foi vítima do risco.

2.3. Execução, Monitoramento e Controle

Na fase de execução, as atividades programadas foram realizadas, e a comunicação entre as partes foi fundamental. Algumas ações típicas nessa etapa incluíram:

- **Sistema de comunicação:** A equipe garantiu que todos os envolvidos foram mantidos atualizados sobre o progresso do projeto. O aplicativo WhatsApp foi usado para facilitar a troca de informações rápidas, enquanto reuniões mensais também foram usadas para alinhar expectativas.

- **Trabalho colaborativo:** Os Projetos de software exigiram colaboração constante entre os membros da equipe. A ferramentas mais usada foi o GitHub, que ajudou a manter a organização das tarefas e acompanhamento progresso e ajudou a fazer ajustes quando necessário. O bom trabalho em equipe foi essencial para garantir que o projeto teve o caminho certo.

Enquanto o projeto foi sendo executado, precisamos monitorá-lo de perto para garantir que tudo estivesse no caminho certo. O monitoramento contínuo permitiu identificar possíveis desvios e corrigir rapidamente. As atividades chave dessa fase incluíram:

- **Controle do cronograma**: O cronograma foi constantemente revisado para garantir que o projeto não ultrapasse os prazos definidos. Se houvesse atrasos, a equipe precisaria agir rapidamente para ajustar as atividades e evitar que o projeto fugisse de controle.

- **Gestão da qualidade**: A qualidade do software foi monitorada de perto. Isso envolveu testes regulares, revisões do código e a utilização de boas práticas de desenvolvimento que garantiram que o produto atendesse aos padrões exigidos.

- **Controle das comunicações**: O fluxo de informações precisou ser mantido durante toda a execução do projeto. Isso incluiu não só as atualizações para as partes interessadas, mas também para garantir que a equipe tinha as informações necessárias para tomar decisões rápidas e assertivas.

A fase de encerramento marcou a conclusão formal do projeto. Aqui o projeto é entregue ao cliente, a documentação final foi organizada e o aprendizado compartilhado. As principais atividades dessa fase foram:

•**Entrega do software:** O produto será entregue ao cliente, que testará em condições reais.

•**Documentação e formalização do encerramento**: A documentação do projeto, incluindo códigos, manuais de usuário e relatórios, serão entregues organizadas.

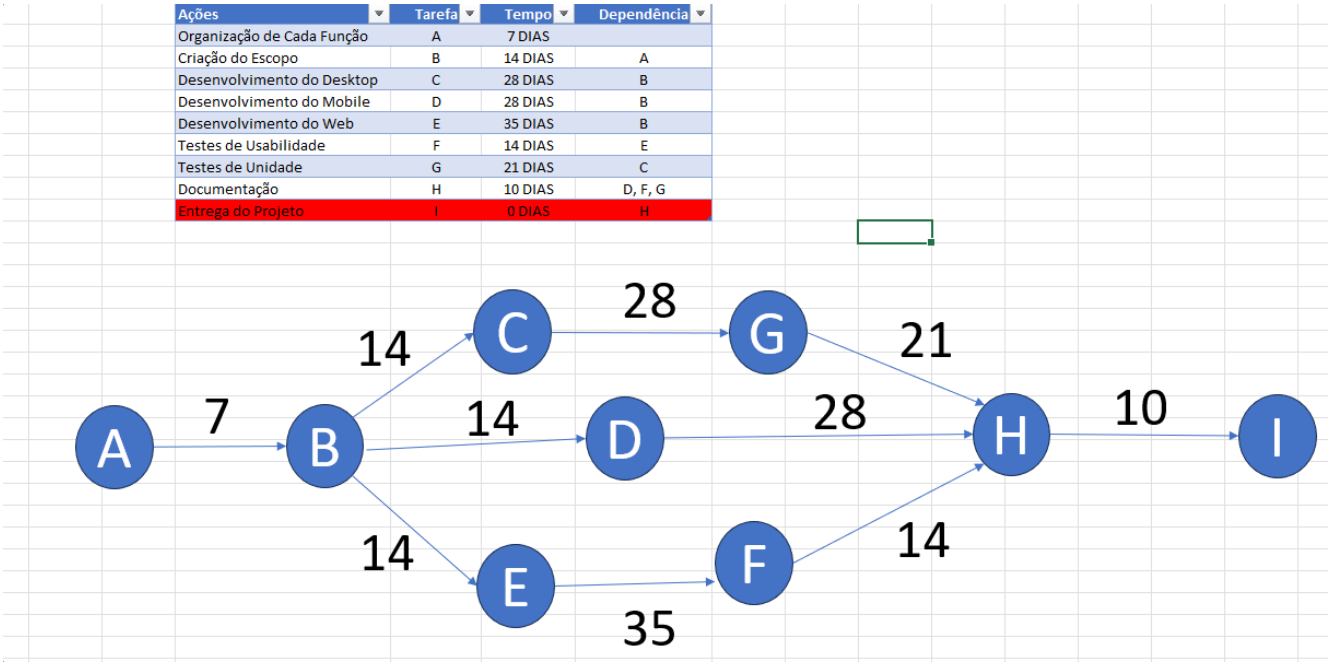
3. DESENVOLVIMENTO DO PROJETO

Após a coleta de requisitos funcionais, não funcionais, diagramas UML, banco de dados e CRUD anteriormente, foi possível dar início ao desenvolvimento total das ferramentas tecnológicas da fazenda urbana TerrasMil, visando a criação e implementação dos sistemas para Web, Mobile e Desktop.

3.1. PERT e Cronograma

Para realização do cronograma e diagrama PERT, foram levantados os seguintes requisitos: Aplicação Web, Aplicação Desktop, Aplicação Mobile, Teste de Unidade e Teste de Usabilidade.

Imagem 1 – PERT.



Fonte: Autor (2024).

A tabela acima mostra as tarefas e prazo estipulado para término das mesmas, sendo o caminho A-B-E-F-H-I o caminho crítico, sinalizando que o projeto levaria no mínimo 67 dias para ficar pronto, sendo seguido por uma milestone ao entregar o projeto realizado ao cliente.

3.2. Desktop

O desktop teve como base grande parte do código da aplicação mobile, devido a facilidade de adaptar com métodos adicionais.

Para o desenvolvimento da parte Desktop foi utilizado a linguagem Python com as bibliotecas Flet, Openpylx e Pandas como um método de integrar uma planilha de Excel como base de dados.

Imagem 2 – Identificação de Tabela.

```
file_path = 'base_dados.xlsx'
usuarios_df = pd.read_excel(file_path, sheet_name="USUARIOS")
```

Fonte: Autor (2024).

O uso de um arquivo Excel foi necessário por causa da confusão causada na integração do Banco de Dados, centraliza os dados dos usuários e possibilita flexibilidade na manipulação. A biblioteca Pandas é responsável por manejar os dados da planilha e integrá-los no app. O trecho acima carrega a planilha de Excel.

Imagem 3 – Autenticação de Usuário.

```
if float(usuario_cpf) in usuarios_df['CPF'].values:
    usuario_logado = usuarios_df[usuarios_df['CPF'] == float(usuario_cpf)].iloc[0]
    page.clean()
    mostrar_landing_page()
else:
    login_error.value = "CPF inválido. Tente novamente."
    page.update()
```

Fonte: Autor (2024).

Autêntica o usuário de acordo com os CPF da base de dados, exibe erro ao utilizar CPF não registrado. Se o CPF estiver correto a função joga o usuário para a landing page.

Imagem 4 – Carregar dados do Excel

```
def carregar_dados(sheet_name):
    df = pd.read_excel(file_path, sheet_name=sheet_name)
    return df
```

Fonte: (Autor 2024).

Lê uma aba específica do arquivo Excel e retorna os dados formatados como um DataFrame.

Imagem 5 – Exibir Tabelas.

```
def exibir_tabela(df):
    columns = [ft.DataColumn(ft.Text(col)) for col in df.columns]
    rows = [
        ft.DataRow(cells=[ft.DataCell(ft.Text(str(cell))) for cell in row])
        for row in df.values
    ]
    return ft.DataTable(columns=columns, rows=rows, width=page.width)
```

Fonte: Autor (2024).

Cria linhas com base no conteúdo do DataFrame e as converte em uma tabela exibível na interface.

Imagem 6 – Pesquisa de Dados.

```
def pesquisar_dados(sheet_name, termo_pesquisa):
    df = carregar_dados(sheet_name)
    df_filtrado = df[df.apply(lambda row: row.astype(str).str.contains(termo_pesquisa,
    return df_filtrado
```

Fonte: Autor (2024).

Filtra os dados de acordo com o termo de pesquisa e retorna linhas que contenham os termos pesquisados.

Imagem 7 – Mostra os métodos de Incluir, Aruivar e Editar dados.

```python
# Funções para incluir, arquivar e editar dados
def incluir_dados(sheet_name, dados):
    df = carregar_dados(sheet_name)
    df = df.append(dados, ignore_index=True)
    df.to_excel(file_path, sheet_name=sheet_name, index=False)
    mostrar_dados(sheet_name)

def arquivar_dados(sheet_name, index):
    df = carregar_dados(sheet_name)
    df = df.drop(index)
    df.to_excel(file_path, sheet_name=sheet_name, index=False)
    mostrar_dados(sheet_name)

def editar_dados(sheet_name, index, dados):
    df = carregar_dados(sheet_name)
    for col, value in dados.items():
        df.at[index, col] = value
    df.to_excel(file_path, sheet_name=sheet_name, index=False)
    mostrar_dados(sheet_name)
```

Fonte: Autor (2024).

Inicializa as funções que manipulam os dados da planilha Excel com base em Incluir, Arquivar e Editar.

Imagem 8 – Exibir Dados.

```python
def mostrar_dados(sheet_name):
    def atualizar_tabela(e=None):
        termo_pesquisa = search_input.value
        df = carregar_dados(sheet_name) if not termo_pesquisa else pesquisar_dados(shee
        data_table.controls.clear()
        data_table.controls.append(exibir_tabela(df))
        page.update()

    search_button.on_click = atualizar_tabela
    atualizar_tabela()
```

Fonte: Autor (2024).

Atualiza a exibição da tabela com base na aba e/ou termo de pesquisa atual, também limpa e carrega os novos dados da interface.

Imagem 9 – Trecho da tela principal.

```
def mostrar_tela_principal():
    page.clean()

    global tabs
    tabs = ft.Tabs(
        selected_index=0,
        tabs=[
            ft.Tab(text="Clientes"),
            ft.Tab(text="Fornecedores"),
            ft.Tab(text="Tarefas"),
            ft.Tab(text="Produção"),
        ],
        on_change=lambda e: [
            mostrar_dados("CLIENTE") if e.control.selected_index == 0 else None,
            mostrar_dados("FORNECEDOR") if e.control.selected_index == 1 else None,
            mostrar_dados("TAREFAS") if e.control.selected_index == 2 else None,
            mostrar_dados("PRODUÇÃO") if e.control.selected_index == 3 else None,
        ],
        width=page.width
    )

    # Texto de boas-vindas
    bem_vindo_text = ft.Text(f"Bem-vindo, {usuario_logado['NOME']}", size=20)
```

Fonte: Autor (2024).

Exibe a tela principal com abas para diferentes tipos de dados e define as abas e o comportamento para carregar os dados ao trocar de aba.

Imagem 10 – Trecho que exibe um formulário para inclusão.

```
def abrir_modal_incluir(sheet_name):
    form = ft.Column(spacing=10)

    # Define os campos
    campos = {
        "CLIENTE": ["CÓDIGO", "CLIENTE", "DATA_REGISTRO", "CONTATO_PRINCIPAL", "NOME_EMPRESA", "STATUS", "ESTADO", "CNPJ"],
        "FORNECEDOR": ["ID", "DATA_REGISTRO", "CONTATO_PRINCIPAL", "NOME_EMPRESA", "STATUS", "ESTADO", "CNPJ", "CEP"],
        "TAREFAS": ["NOME", "PRIORIDADE", "Prazo", "ETAPA", "N_ETAPAS"],
        "PRODUÇÃO": ["ID", "DATA_INICIO", "PRAZO", "QTD", "DATA_FIM", "PRIORIDADE", "NOMEPRODUTO", "ID_FORNECEDOR", "CODIGO_CLIENTE
    }

    for campo in campos[sheet_name]:
        form.controls.append(ft.TextField(label=campo, name=campo))

    # Função para salvar os dados preenchidos
    def salvar_dados(e):
        dados = {campo: input.value for input in form.controls}
        incluir_dados(sheet_name, dados)
        modal.close()  # Fecha o modal após salvar
        mostrar_dados(sheet_name)  # Atualiza a tabela exibida

    # Modal de entrada de dados
    modal = ft.AlertDialog(
        title=ft.Text(f"Incluir {sheet_name.capitalize()}"),
        content=form,
        actions=[ft.ElevatedButton("Salvar", on_click=salvar_dados)],
        modal=True,
    )
```

Fonte: Autor (2024).

Esse trecho de código cria uma interface de modal que permite ao usuário inserir novos dados em uma planilha específica ("Clientes", "Fornecedores", "Tarefas" ou "Produção").

Imagem 11 – Trecho que exibe um formulário para Editar.

```python
def abrir_modal_editar(sheet_name, index):
    # Carrega os dados existentes
    df = carregar_dados(sheet_name)
    linha = df.iloc[index]

    form = ft.Column(spacing=10)
    campos = {
        "CLIENTE": ["CÓDIGO", "CLIENTE", "DATA_REGISTRO", "CONTATO_PRINCIPAL", "NOME_EMPRESA", "STATUS", "ESTADO", "CNPJ"],
        "FORNECEDOR": ["ID", "DATA_REGISTRO", "CONTATO_PRINCIPAL", "NOME_EMPRESA", "STATUS", "ESTADO", "CNPJ", "CEP"],
        "TAREFAS": ["NOME", "PRIORIDADE", "Prazo", "ETAPA", "N_ETAPAS"],
        "PRODUÇÃO": ["ID", "DATA_INICIO", "PRAZO", "QTD", "DATA_FIM", "PRIORIDADE", "NOMEPRODUTO", "ID_FORNECEDOR", "CODIGO_CLIENTE
    }

    for campo in campos[sheet_name]:
        form.controls.append(ft.TextField(label=campo, name=campo, value=str(linha[campo])))

    # Função para salvar os dados editados
    def salvar_edicao(e):
        dados = {campo: input.value for input in form.controls}
        editar_dados(sheet_name, index, dados)
        modal.close()  # Fecha o modal após salvar
        mostrar_dados(sheet_name)  # Atualiza a tabela exibida

    # Modal de edição de dados
    modal = ft.AlertDialog(
        title=ft.Text(f"Editar {sheet_name.capitalize()}"),
        content=form,
        actions=[ft.ElevatedButton("Salvar", on_click=salvar_edicao)],
        modal=True,
```

Fonte: Autor (2024).

Esse trecho de código cria um modal para editar um registro específico na planilha.

Imagem 12 – Trecho que implementa a lógica para abrir modais.

```python
            modal=True,
        )
        page.dialog = modal
        modal.open = True
        page.update()

    incluir_button = ft.ElevatedButton("Incluir", on_click=lambda e: abrir_modal_incluir("CLIENTE"))

    arquivar_button = ft.ElevatedButton("Arquivar", on_click=lambda e: arquivar_dados("CLIENTE", 0))
    editar_button = ft.ElevatedButton("Editar", on_click=lambda e: abrir_modal_editar("CLIENTE", 0))
```

Fonte: Autor (2024).

Exibe o trecho de código que implementa a lógica para abrir modais e executar ações como incluir, arquivar, editar registros.

Imagem 13 – Trecho da tela de login.

```
def mostrar_tela_login():
    page.add(
        ft.Stack(
            controls=[
                ft.Image(src="fazendas.jpg", opacity=0.3, fit=ft.ImageFit.COVER, width=page.width, height=page.height),
                ft.Column(
                    controls=[
                        ft.Text("Login", size=24),
                        login_input,
                        login_button,
                        login_error,
                    ],
                    alignment=ft.MainAxisAlignment.CENTER,
                    horizontal_alignment=ft.CrossAxisAlignment.CENTER,
                    width=page.width,
                    height=page.height
                ),
            ]
        )
    )
```

Fonte: Autor (2024).

Renderiza a tela inicial de login com uma imagem de fundo e define uma imagem opaca de fundo para a tela.

Imagem 14 – Trecho da Landing Page.

```
def mostrar_landing_page():
    page.add(
        ft.Stack(
            controls=[
                ft.Image(src="fazendas.jpg", opacity=0.3, fit=ft.ImageFit.COVER, width=page.width, height=page.height),
                ft.Column(
                    controls=[
                        ft.Text("Bem-vindo à TerrasMil", size=30, weight="bold"),
                        ft.Text(
                            "TerrasMil é uma fazenda urbana comprometida com práticas sustentáveis e a produção de alimentos frescos para nossa comunidade.\n"
                            "Nossos valores incluem inovação, sustentabilidade e responsabilidade com o meio ambiente. Estamos orgulhosos de nossa equipe que "
                            "trabalha para oferecer o melhor todos os dias.",
                            size=18,
                            text_align="center"
                        ),
                        ft.ElevatedButton("Acessar dados", on_click=lambda e: page.clean() or mostrar_tela_principal())
                    ],
                    alignment=ft.MainAxisAlignment.CENTER,
                    horizontal_alignment=ft.CrossAxisAlignment.CENTER,
                    spacing=20,
                    width=page.width,
                    height=page.height
                ),
            ]
        )
    )
```

Fonte: Autor (2024).

Exibe a página de boas-vindas ao usuário autenticado junto com uma mensagem talvez inspiradora sobre a empresa.

3.3 Mobile

Para o desenvolvimento da parte mobile foi utilizado a linguagem Python com as bibliotecas Flet, Openpylx e Pandas como um método de integrar uma planilha de Excel como base de dados.

Imagem 15 – Identificação de Tabela.

```
file_path = 'base_dados.xlsx'
usuarios_df = pd.read_excel(file_path, sheet_name="USUARIOS")
```

Fonte: Autor (2024).

O uso de um arquivo Excel foi necessário por causa da confusão causada na integração do Banco de Dados, centraliza os dados dos usuários e possibilita flexibilidade na manipulação. A biblioteca Pandas é responsável por manejar os dados da planilha e integrá-los no app. O trecho acima carrega a planilha de Excel.

Imagem 16 – Autenticação de Usuário.

```
if float(usuario_cpf) in usuarios_df['CPF'].values:
    usuario_logado = usuarios_df[usuarios_df['CPF'] == float(usuario_cpf)].iloc[0]
    page.clean()
    mostrar_landing_page()
else:
    login_error.value = "CPF inválido. Tente novamente."
    page.update()
```

Fonte: Autor (2024).

Autêntica o usuário de acordo com os CPF da base de dados, exibe erro ao utilizar CPF não registrado. Se o CPF estiver correto a função joga o usuário para a landing page.

Imagem 17 – Carregar dados do Excel

```
def carregar_dados(sheet_name):
    df = pd.read_excel(file_path, sheet_name=sheet_name)
    return df
```

Fonte: (Autor 2024).

Lê uma aba específica do arquivo Excel e retorna os dados formatados como um DataFrame.

Imagem 18 – Exibir Tabelas.

```
def exibir_tabela(df):
    columns = [ft.DataColumn(ft.Text(col)) for col in df.columns]
    rows = [
        ft.DataRow(cells=[ft.DataCell(ft.Text(str(cell))) for cell in row])
        for row in df.values
    ]
    return ft.DataTable(columns=columns, rows=rows, width=page.width)
```

Fonte: Autor (2024).

Cria linhas com base no conteúdo do DataFrame e as converte em uma tabela exibível na interface.

Imagem 19 – Pesquisa de Dados.

```
def pesquisar_dados(sheet_name, termo_pesquisa):
    df = carregar_dados(sheet_name)
    df_filtrado = df[df.apply(lambda row: row.astype(str).str.contains(termo_pesquisa,
    return df_filtrado
```

Fonte: Autor (2024).

Filtra os dados de acordo com o termo de pesquisa e retorna linhas que contenham os termos pesquisados.

Imagem 20 – Exibir Dados.

```
def mostrar_dados(sheet_name):
    def atualizar_tabela(e=None):
        termo_pesquisa = search_input.value
        df = carregar_dados(sheet_name) if not termo_pesquisa else pesquisar_dados(shee
        data_table.controls.clear()
        data_table.controls.append(exibir_tabela(df))
        page.update()

    search_button.on_click = atualizar_tabela
    atualizar_tabela()
```

Fonte: Autor (2024).

Atualiza a exibição da tabela com base na aba e/ou termo de pesquisa atual, também limpa e carrega os novos dados da interface.

Imagem 21 – Trecho da tela principal.

```python
def mostrar_tela_principal():
    page.clean()

    global tabs
    tabs = ft.Tabs(
        selected_index=0,
        tabs=[
            ft.Tab(text="Clientes"),
            ft.Tab(text="Fornecedores"),
            ft.Tab(text="Tarefas"),
            ft.Tab(text="Produção"),
        ],
        on_change=lambda e: [
            mostrar_dados("CLIENTE") if e.control.selected_index == 0 else None,
            mostrar_dados("FORNECEDOR") if e.control.selected_index == 1 else None,
            mostrar_dados("TAREFAS") if e.control.selected_index == 2 else None,
            mostrar_dados("PRODUÇÃO") if e.control.selected_index == 3 else None,
        ],
        width=page.width
    )

    # Texto de boas-vindas
    bem_vindo_text = ft.Text(f"Bem-vindo, {usuario_logado['NOME']}", size=20)
```

Fonte: Autor (2024).

Exibe a tela principal com abas para diferentes tipos de dados e define as abas e o comportamento para carregar os dados ao trocar de aba.

Imagem 22 – Trecho da tela de login.

```
def mostrar_tela_login():
    page.add(
        ft.Stack(
            controls=[
                ft.Image(src="fazendas.jpg", opacity=0.3, fit=ft.ImageFit.COVER, width=page.width, height=page.height),
                ft.Column(
                    controls=[
                        ft.Text("Login", size=24),
                        login_input,
                        login_button,
                        login_error,
                    ],
                    alignment=ft.MainAxisAlignment.CENTER,
                    horizontal_alignment=ft.CrossAxisAlignment.CENTER,
                    width=page.width,
                    height=page.height
                ),
            ]
        )
    )
```

Fonte: Autor (2024).

Renderiza a tela inicial de login com uma imagem de fundo e define uma imagem opaca de fundo para a tela.

Imagem 23 – Trecho da Landing Page.

```
def mostrar_landing_page():
    page.add(
        ft.Stack(
            controls=[
                ft.Image(src="fazendas.jpg", opacity=0.3, fit=ft.ImageFit.COVER, width=page.width, height=page.height),
                ft.Column(
                    controls=[
                        ft.Text("Bem-vindo à TerrasMil", size=30, weight="bold"),
                        ft.Text(
                            "TerrasMil é uma fazenda urbana comprometida com práticas sustentáveis e a produção de alimentos frescos para nossa comunidade.\n"
                            "Nossos valores incluem inovação, sustentabilidade e responsabilidade com o meio ambiente. Estamos orgulhosos de nossa equipe que "
                            "trabalha para oferecer o melhor todos os dias.",
                            size=18,
                            text_align="center"
                        ),
                        ft.ElevatedButton("Acessar dados", on_click=lambda e: page.clean() or mostrar_tela_principal())
                    ],
                    alignment=ft.MainAxisAlignment.CENTER,
                    horizontal_alignment=ft.CrossAxisAlignment.CENTER,
                    spacing=20,
                    width=page.width,
                    height=page.height
                ),
            ]
        )
    )
```

Fonte: Autor (2024).

Exibe a página de boas-vindas ao usuário autenticado junto com uma mensagem talvez inspiradora sobre a empresa.

Foi utilizado o BeeWare para gerar o apk TerrasMil a partir do código fonte.

3.4. Web

Para o desenvolvimento da web, foram utilizadas as linguagens HTML, CSS e JavaScript.

Imagem 24 - Formulário de Login

```
14          <section class="container">
15              <h1> Bem vindo ao Terras Mil 🐤</h1>
16              <div class="social-login">
17              </div>
18              <div class="divider">
19                  <div class="line"></div>
20                  <div class="line"></div>
21              </div>
22
23              <form>
24                  <label for="email">Email:</label>
25                  <div class="custome-input">
26                      <input type="email" name="email" placeholder="Seu email" required>
27                      <i class='bx bx-at'></i>
28                  </div>
29                  <label for="password">Senha:</label>
30                  <div class="custome-input">
31                      <input type="password" name="password" placeholder="Sua senha" required>
32                      <i class='bx bx-lock-alt'></i>
33                  </div>
34                  <button class="login" type="button" onclick="window.location.href='dashboard.html'">Login</button>
35
36
37                  <div class="links">
38                      <a href="#" onclick="forgotPassword()">Esqueci minha senha</a>
39                  </div>
40              </form>
41
42              <section id="forgotPasswordSection" style="display: none;">
43                  <p>Verifique sua solicitação no RH</p>
44              </section>
45
46          </section>
```

Fonte: Autor(2024).

Trecho do código que contém todas as informações do login, com o placeholder para email e senha, botão de login e a opção para recuperar a senha.

Imagem 25 - Recuperar senha

```
<script>
function forgotPassword() {
    const forgotPasswordSection = document.getElementById("forgotPasswordSection");
    if (forgotPasswordSection.style.display === "none") {
        forgotPasswordSection.style.display = "block";
    } else {
        forgotPasswordSection.style.display = "none";
    }
}
</script>
```

Fonte: Autor(2024)

Trecho do código que contém a função executada ao clicar na opção de “Esqueci minha senha”, que irá mostrar a mensagem que está escondida e caso o usuário clique na opção novamente, ele irá esconder a mensagem outra vez.

Imagem 26 – Funções do login

```
1   // Seleciona o botão com a classe login
2   const loginButton = document.querySelector('.login');
3
4   // Adiciona o evento 'click' ao botão
5   loginButton.addEventListener('click', () => {
6     // Redireciona para dashboard.html
7     window.location.href = 'dashboard.html';
8   });
9
10  //Função para usuário que esqueceu senha
11  function forgotPassword() {
12    document.getElementById("forgotPasswordSection").style.display = "block";
13  }
```

Fonte: Autor(2024).

Este trecho do código apresenta a função de redirecionamento de link, que ao usuário inserir suas informações e clicar no botão de login, ele será redirecionado ao dashboard.

Imagem 27 - Gráfico de Produção da semana

```
48      new Chart(ctx2, {
49          type: 'line',
50          data: {
51              labels: ['Jan', 'Fev', 'Mar', 'Abr', 'Mai', 'Jun', 'Jul'],
52              datasets: [{
53                  label: 'Class GPA',
54                  data: [6, 10, 8, 14, 6, 7, 4],
55                  borderColor: '#0891b2',
56                  tension: 0.4
57              },
58              {
59                  label: 'Aver GPA',
60                  data: [8, 6, 7, 6, 11, 8, 10],
61                  borderColor: '#ca8a04',
62                  tension: 0.4
63              }
64              ]
65          },
```

Fonte: Autor(2024).

Trecho resumido do código que contém a função do gráfico de produção da semana.

Imagem 28 - Gráfico de Produção em toneladas

```
new Chart(ctx2, {
    type: 'line',
    data: {
        labels: ['Jan', 'Fev', 'Mar', 'Abr', 'Mai', 'Jun', 'Jul'],
        datasets: [{
            label: 'Class GPA',
            data: [6, 10, 8, 14, 6, 7, 4],
            borderColor: '#0891b2',
            tension: 0.4
        },
        {
            label: 'Aver GPA',
            data: [8, 6, 7, 6, 11, 8, 10],
            borderColor: '#ca8a04',
            tension: 0.4
        }
        ]
    },
```

Fonte: Autor(2024)

3.5. Tabela de Testes

Para garantir o melhor funcionamento do sistema de gerenciamento, foi criado uma planilha com testes que a equipe de desenvolvimento julgou necessários, tendo tido como base melhorar a usabilidade da interface e checar erros presentes no código de algumas classes.

A tabela 3 ilustra os testes escolhidos para o software da fazenda urbana Terras Mil.

Tabela 1 – Planilha de Testes.

Teste de Unidade					
Nível de Prioridade	**Tipo de teste**	**Responsável**	**Resultado Esperado**		**Comentários**
Alta	Unidade/Unitário	Ruan	Sim ou Não	Sim	Mudança de classe
Baixa	Unidade/Unitário	Ruan	Sim ou Não	Sim	Adição e troca de valores
Baixa	Unidade/Unitário	Matheus	Sim ou Não	Não	Execução de métodos
Alta	Unidade/Unitário	Matheus	Sim ou Não	Não	Implementação do banco de dados
Teste A/B					
Nível de Prioridade	**Tipo de teste**	**Responsável**	**Resultado Esperado**		**Comentários**
Alta	Teste A/B	Luís	A > B ou B > A	B>A	Botões podem estar melhor posicionados
Alta	Teste A/B	Lucas	A > B ou B > A	A>B	Os gráficos no A contém um melhor posicionamento do que o B
Baixa	Teste A/B	Luís	A > B ou B > A	A>B	Sistema sem vida(muitas cores neutras)
Média	Teste A/B	Lucas	A > B ou B > A	B>A	Alterar o posicionamento dos elementos na tela para melhor harmonia visual
	Numero participantes				
	3 a 5				

Fonte: Autor (2024).

Foi decidido utilizar o teste de unidade para checar incongruências e atividades suspeitas no código das classes, as classes escolhidas foram as julgadas mais complexas para o sistema.

O teste A/B foi escolhido por sua versatilidade em compara variáveis da interface, tendo como objetivo testar as diferentes alterações feitas no protótipo da interface.

O teste unitário obteve resultados mistos, enquanto o de usabilidade mostrou resultados esperados pelo responsáveis do teste, certos elementos gráficos eram realmente estranhos e não se encaixavam no modelo final do protótipo e projeto.

4. CONSIDERAÇÕES FINAIS

Seguindo logo após a colheita de requisitos e modelagem de sistema, entramos na parte do de facto desenvolvimento do sistema da fazenda urbana TerrasMil, desenvolvimento esse que seguiu a trancos e barrancos, com uma equipe caótica e apocalítica, com certos atrasos no desenvolvimento do projeto, tempos turbulentos que se tornaram apenas um inconveniência graças ao comprometimento do time para com a entrega no prazo correto.

Foi definido como objetivo do projeto a criação de tres aplicações, para web, desktop e mobile. No fim todas foram desenvolvidas, porém com certas diferenças do idealizado, principalmente no quesito do banco de dados e sua integração com o código integração esse que foi ardilosa e desafiadora. Por causa de implicações na implantação, o aplicativo mobile acabou por ter a biblioteca Pandas e Excel como substituto da integração com SQL Server.

Todavia, o dev team mostrou resiliência e persistência para com os empecilhos durante o progresso do projeto. Sempre seguindo em frente apesar das frustrações proporcionadas pela lógica de programação das aplicações da fazenda urbana TerrasMil. O próprio desenvolvimento do desktop foi pausado por isso, ou seja, estava totalmente inviável continuar com os métodos utilizados com a linguagem python na criação do desktop, elemento esse que seria a biblioteca Kivy, que foi iniciado com tudo ocorrendo bem, porém conforme o código foi ficando cada vez mais complexo, mais erros apareciam onde o Kivy estava inserido. Erros que a equipe atribuiu á complexidade da escrita dos comandos do Kivy, e como o Kivy estava deixando a desejar, foi substituído pela biblioteca Flet.

De um ponto de vista diferente, todos esses desafios não foram necessariamente ruins para a equipe, tendo em vista que com eles foram aprendidas várias técnicas de resolução de problemas, além de trabalhar o diálogo e comunicação da equipe, tendo até momentos de união entre a maioria dos membros para resolver pendencias atrasadas do cronograma e erros aparentemente sem resolução aparente.

No fim, a equipe superou os desafios apresentados e terminou o projeto, tornando-se talvez mais madura e experiente do que estava no começo.

REFERÊNCIAS

AWARI. **Gestão de Projetos PMI: O que é e como funciona**. Disponível em: https://awari.com.br/gestao-de-projetos-pmi-o-que-e-e-como-funciona/. Acesso em: 4 set. 2024.

NAÇÕES UNIDAS BRASIL. **População mundial deve chegar a 9,7 bilhões de pessoas em 2050, diz relatório da ONU**. Disponível em: https://brasil.un.org/pt-br/83427-população-mundial-deve-chegar-97-bilhões-de-pessoas-em-2050-diz-relatório-da-onu. Acesso em: 9 abr. 2024.

PM BLOG. **Guia PMBOK com as melhores práticas de Gerenciamento de Projetos em 2024**. Disponível em: https://pmp.com.br/metodologias/pmbok-guide/. Acesso em: 19 set. 2024.

PM BLOG. **PMBOK Guide ® e Gerenciamento de Projetos com Excelência**. Disponível em: https://pmp.com.br/ferramentas/pmbok-guide-gerenciamento-de-projetos/. Acesso em: 1 out. 2024.

PROJECT BUILDER. **O que é PMI?**. Disponível em: https://www.projectbuilder.com.br/blog/o-que-e-pmi/. Acesso em: 7 out. 2024.

SANEAMENTO EM PAUTA. **Fazendas urbanas: conheça o conceito e descubra seus benefícios**. Disponível em: https://blog.brkambiental.com.br/fazendas-urbanas/#:~:text=Fazendas%20urbanas%20são%20espaços%20concebidos,estufas%20agrícolas%20anexas%20às%20edificações.. Acesso em: 15 abr. 2024.

FICHA DE CONTROLE DO PIM

Ano: 2024 Período: 3°/4° Coordenador: Prof Roberto Cordeiro Waltz

Tema (Identificação da startup): Sistema de gerenciamento da fazenda urbana TerrasMil.

Alunos

RA	Nome	E-mail	Curso	Visto do aluno
G81BH5C	Carlos Henrique Duarte da Silva	carlos.silva818@aluno.unip.br	**CST em ADS**	
R0167J3	João Gabriel Alves dos Santos	joao.santos811@aluno.unip.br	**CST em ADS**	
R009JD0	Lucas Navimar de Toledo	lucas.toledo12@aluno.unip.br	**CST em ADS**	
R024768	Luis Henrique Vieira Barros	luis.barros15@aluno.unip.br	**CST em ADS**	
R026272	Matheus Guilherme Pacheco Courbassier	matheus.cour@aluno.unip.br	**CST em ADS**	
G86JCF-1	Ruan Pablo Galdino Dias Bento	ruan.bento@aluno.unip.br	**CST em ADS**	

Registros

Data do encontro	Observações
20/8/2024	Revisão do PIM III
24/8/2024	Especificação de requisitos
01/09/2024	Escolha de recursos a serem utilizados no projeto
11/09/2024	Começo dos diagramas UML
16/10/2024	Início das três aplicações
23/10/2024	Tentativa integração do banco de dados
25/10/2024	Início da Documentação
09/11/2024	Revisão dos Testes
14/11/2024	Revisão de todas as partes do projeto