LOCAL FARM | Fazenda Urbana

# Sobre

Com a crescente ideia de fazendas urbanas sendo mais abrangente no dia-a-dia do consumidor e do produtor local, com prédios e empresas sendo montados para abranger um sistema de produção de hortaliças, vegetais e legumes. Viu-se a necessidade de um sistema mais robusto para o gerenciamento do mesmo, para que o fornecedor e produtor local entre como uma nova vertente no sistema, para que se tenha um mapeamento mais detalhado e por fim, ter uma análise de dados e escolher melhores decisões para a fazenda urbana, sendo assim, gerando lucro e rotatividade de fornecedores e consumidores.

# Integrantes

1. Gabriel Pinelli Silva (PO)
2. Luana Martini (Scrum Master)
3. Arieli (Dev)
4. Beatriz (Dev)
5. Mateus (Dev)
6. Fábio (Dev)

# BACKLOG PRODUTO

1. Realizar, inicialmente, uma pesquisa do que são fazendas urbanas, seus tipos, e relatar um breve histórico no Brasil e no mundo. Descrever o que significa segurança alimentar e banco de alimentos. Explorar os princípios da ODS (Objetivos de Desenvolvimento Sustentável). Explicar o que é ESG (em português meio ambiente, social e governança corporativa) e seu relacionamento com os princípios da ODS. Descrever os objetivos da COP30 (30ª Conferência da ONU sobre Mudanças Climáticas) e como o Brasil está envolvido nesse tema.
2. Criar um cenário bem detalhado, com: regras do negócio, glossário do sistema, pesquisa de mercado, livros sobre o tema.
3. Definir e justificar o ciclo de vida de desenvolvimento de software
4. Descrever os requisitos dos usuários e os requisitos do sistema. Classificar cada requisito (tanto os de usuários quanto os de sistema) como requisito funcional ou não funcional. Especificar o tipo de cada requisito não funcional levantado (usabilidade, desempenho, capacidade etc.);
5. Elaborar protótipos de telas para aprovação dos gestores do sistema;
6. Elaborar os modelos de casos de uso para cada cenário e um geral resumido. Cada caso de uso deve ter uma descrição suscinta do seu comportamento, dos fluxos principais, alternativos e de exceção e pré e pós-condições. Identificar relacionamentos de include, extend e generalização;
7. Elaborar os diagramas de classes de análise (Boundary, Control, Entity). Demonstrar o comportamento dos casos de uso por meio do diagrama de sequência. Elaborar o diagrama de implantação (definir quantos servidores, banco de dados, estrutura utilizada para o sistema e como instalar o sistema);
8. Elaborar o Diagrama ER do banco de dados e o dicionário de dados.
9. Gerar o script de criação do banco (use uma ferramenta) e scripts de dados iniciais de testes (roteiros de teste) e homologação do sistema (testes de unidade, integração, performance, usabilidade etc);
10. Criar planilhas de testes para homologação do sistema identificando como produzir as evidências do teste e as querys no banco, para certificar que os dados estão corretos (um mínimo de testes que assegurem a funcionalidade básica do sistema);
11. Elaborar manual de uso do sistema para treinamento (eletrônico).
12. Definir relatórios de gestão para análise de evolução dos negócios, análise de mercado, desempenho dos funcionários (RH) etc;
13. PoC (Proof of Concept – prova de conceito) em modo console que faça o CRUD (Creat – Criar - , Read – Ler, Update – Atualizar e Delete – Apagar). \* \*Chamar classes em uma dll para o cadastro dos itens no banco. No segundo semestre, o sistema deverá ser implementado.
14. Economia e mercado ---> Estimativa e viabilidade.
15. Gestão estratégica de recursos humanos ---> Descrever quais recursos humanos serão utilizados.

Primeira Sprint:

# BACKLOG DAS SPRINTS

Descrever os requisitos dos usuários e os requisitos do sistema. Classificar cada requisito (tanto os de usuários quanto os de sistema) como requisito funcional ou não funcional. Especificar o tipo de cada requisito não funcional levantado (usabilidade, desempenho, capacidade etc.);

## Segunda Sprint:

Elaborar os diagramas de classes de análise (Boundary, Control, Entity).

Demonstrar o comportamento dos casos de uso por meio do diagrama de sequência. Elaborar o diagrama de implantação (definir quantos servidores, banco de dados, estrutura utilizada para o sistema e como instalar o sistema);

## Terceira Sprint:

Criar um cenário bem detalhado, com: regras do negócio, glossário do sistema, pesquisa de mercado, livros sobre o tema.

## Quarta Sprint:

Elaborar os modelos de casos de uso para cada cenário e um geral resumido. Cada caso de uso deve ter uma descrição suscinta do seu comportamento, dos fluxos principais, alternativos e de exceção e pré e pós-condições. Identificar relacionamentos de include, extend e generalização;

## Quinta Sprint:

Definir e justificar o ciclo de vida de desenvolvimento de software

## Sexta Sprint:

Elaborar o Diagrama ER do banco de dados e o dicionário de dados.

## Sétima Sprint:

Gerar o script de criação do banco (use uma ferramenta) e scripts de dados iniciais de testes (roteiros de teste) e homologação do sistema (testes de unidade, integração, performance, usabilidade etc);

## Oitava Sprint:

Criar planilhas de testes para homologação do sistema identificando como produzir as evidências do teste e as querys no banco, para certificar que os dados estão corretos (um mínimo de testes que assegurem a funcionalidade básica do sistema);

## Nona Sprint:

Elaborar protótipos de telas para aprovação dos gestores do sistema;

## Décima Sprint:

Elaborar o manual de uso do sistema para treinamento (eletrônico).

## Décima Primeira Sprint:

Definir relatórios de gestão para análise de evolução dos negócios, análise de mercado, desempenho dos funcionários (RH) etc;

## Décima Segunda Sprint:

PoC (Proof of Concept – prova de conceito) em modo console que faça o CRUD (Creat – Criar - , Read – Ler, Update – Atualizar e Delete – Apagar). \* \*Chamar classes em uma dll para o cadastro dos itens no banco. No segundo semestre, o sistema deverá ser implementado.

## Décima Terceira Sprint:

Economia e mercado ---> Estimativa e viabilidade.

## Décima Quarta Sprint:

Gestão estratégica de recursos humanos ---> Descrever quais recursos humanos serão utilizados.

# BACKLOG DO PIM

- **Introdução**: A introdução deve apresentar o tema de modo contextualizado, de forma que fique claro ao leitor qual o objetivo do projeto. É preciso explicitar claramente o objetivo do trabalho e a delimitação do estudo. Pensem que seus leitores precisam entender o foco de seu trabalho logo ao ler o resumo e a introdução. Como leitores, ao terminarmos de ler sua introdução, temos que estar motivados e curiosos para lermos o restante do PIM.

- **Considerações finais:**  Nas considerações finais deve haver um texto relacionando o problema que foi proposto e as soluções que foram sugeridas, de tal maneira a valorizar as opções adotadas e motivar o prosseguimento dos estudos, para que a empresa de locação receba mais propostas e contribuições da área de Análise e Desenvolvimento de Sistemas e suas aplicações.

- **Resumo:** O RESUMO deve apresentar, obrigatoriamente: • Panorama do tema, objetivo, metodologia, resultados, conclusões, palavras chave. • O texto do resumo deve ser escrito no passado, em um único parágrafo com uma extensão de 200 a 250 palavras ou 1400 a 1700 caracteres, fonte: Arial 12, espaço entrelinhas simples. • O resumo deve começar indicando qual é a natureza do trabalho, indicando o tema tratado, os objetivos a serem alcançados e as conclusões ou resultados a que se chegou com o projeto. • No texto deve haver a indicação clara dos seguintes itens: objetivo do trabalho; principais teóricos estudados e conceitos mais relevantes; procedimentos metodológicos – contexto pesquisado, dados coletados caso isso tenha sido feito, procedimentos de análise; síntese dos resultados obtidos / descobertos após a análise; e rápida conclusão do grupo.

- **Padronização do documento**:

- **Criar apresentação**: Apresentação em Powerpoint e submissão à banca.