UNIVERSIDADE DE SÃO PAULO (USP)

INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO (ICMC)

DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO (SCC)

Gabriel Costa - 14785489

Isabella Arão - 9265732

Marina Fagundes - 9265405

Agriware

São Carlos

2023

Gabriel Costa - 14785489

Isabella Arão - 9265732

Marina Fagundes - 9265405

Agriware - Metodologia de Desenvolvimento

Relatório apresentado à disciplina Metodologia de Desenvolvimento de Software, como parte dos requisitos para aprovação na matéria oferecida pela Universidade de São Paulo, na área de Computação.

Prof.: Ellen Francine Barbosa

Disciplina: Metodologia de Desenvolvimento de Software

Turma: Turma SSC 0532

São Carlos

2023

1. Agriware

A estrutura completa do agronegócio exige lidar com diferentes problemas ao longo da cadeia produtiva, como manipulação de dados descentralizados, o que acarreta dificuldade de comunicação e coordenação de projetos. Isso gera problemas na tomada de decisão e causa contratempos que podem gerar, no fim, prejuízos para as partes interessadas.

A proposta do *Agriware* vem justamente para suprir essas dificuldades, ao criar um sistema de informação que foque em gestão de estoque de insumos, ramos essenciais para o agronegócio e que exigem a manipulação cuidadosa. Essa solução visa automatizar processos, para que o produtor tenha acesso às informações necessárias para ter sucesso no seu negócio.

2. Declaração dos objetivos

O Sistema de Informações *Agriware* busca organizar os insumos agrícolas da fazenda em um único sistema. Seu objetivo principal é **centralizar e gerenciar as informações relacionadas aos insumos**, como sementes, fertilizantes, defensivos e equipamentos.

Os objetivos específicos são:

1) **Implementar uma base de dados centralizada**, em que todas as informações relevantes sobre os insumos serão armazenadas: isso incluirá dados como quantidade disponível, validade, histórico de compras e uso. Com todas essas informações reunidas em um só lugar, a equipe responsável pela gestão do estoque terá **acesso fácil e rápido aos dados necessários** para o controle e a reposição adequada dos insumos;
2) **Registrar entradas e saídas de cada item do estoque**, atualizando automaticamente a quantidade disponível em tempo real: dessa forma, é possível acompanhar o estoque de forma precisa e evitar problemas como a falta de insumos quando necessários ou o excesso deles, reduzindo desperdícios;
3) **Planejar a reposição de estoque com base em dados históricos e projeções de demanda**: com relatórios gerados pelo sistema, é possível identificar padrões de consumo, calcular as quantidades necessárias para

cada período e tomar decisões embasadas sobre quando e quanto repor os insumos agrícolas;

4) Permitir a **análise de desempenho do estoque ao longo do tempo**: assim, será possível fornecer dados e relatórios que poderão possibilitar a avaliação do desempenho do estoque, identificar áreas de melhoria e otimizar processos. Essas análises contribuirão para aprimorar continuamente a gestão do estoque, aumentando a eficiência e a produtividade da fazenda.

Outro objetivo específico extras estão relacionados ao gerenciamento de serviços terceirizados e à possibilidade de implementação do *Agriware* em dispositivos móveis.

3. Definição de limites do projeto

3.1. Partes interessadas

1) **Clientes:** são os proprietários de fazendas que adotarão o *Agriware* para melhorar a gestão de suas operações agrícolas. Eles se beneficiarão com melhorias na eficiência operacional, na tomada de decisões e no gerenciamento do estoque de insumos.
2) **Funcionários:** são os envolvidos diretamente nas operações diárias da fazenda. Para eles, o *Agriware* facilitará o seu trabalho, trazendo maior eficiência nas tarefas diárias e contribuindo para a melhoria na comunicação e colaboração entre os membros da equipe.
3) **Usuários:** são os gerentes das fazendas, que utilizarão o sistema para a geração de relatórios para tomar decisões. Com o *Agriware*, eles poderão tomar decisões estratégicas embasadas em dados confiáveis e em relatórios precisos gerados pelo sistema.

3.2. Limites

1) **Dentro do escopo:**
    a) Centralização e gerenciamento das informações coletadas;
    b) Base de dados centralizada;
    c) Registro das entradas e saídas de cada item do estoque;
    d) Geração de relatórios contendo diversas informações valiosas;
    e) Controle organizado e atualizado de serviços terceirizados.

2) **Fora do escopo:**
   - **a)** Integração com outro sistema já existente na fazenda;
   - **b)** Qualquer atividade da fazenda que não esteja relacionada com insumos agrícolas;
   - **c)** Previsão climática: o software não abrange informações meteorológicas;
   - **d)** Gestão de recursos humanos: assuntos relacionados a funcionários, folha de pagamento etc. não fazem parte do escopo.

4. Metodologia de desenvolvimento

A proposta do sistema *Agriware* leva em consideração a participação ativa do cliente durante o processo de desenvolvimento, para atingir os objetivos da melhor maneira possível. Diante desse contexto, a metodologia escolhida para desenvolvimento desse sistema de informação é o modelo evolutivo incremental.

Esse modelo combina elementos dos modelos tradicionais cascata e prototipação. Do primeiro, utiliza as definições de fases de projeto, mas com a possibilidade de voltar a fases anteriores se necessário. Do segundo, utiliza a filosofia iterativa. O modelo incremental propõe a união de forças entre equipe e cliente para que os requisitos sejam coletados da melhor forma, além de elencar as funcionalidades que deverão ter maior importância. Após isso, serão definidos estágios de entrega, em que cada estágio fornece um subconjunto das funcionalidades do sistema como um todo, para que seja entregue ao cliente uma versão executável, que poderá ser testada pelo usuário e então ter os feedbacks necessários.

O motivo da escolha dessa metodologia se deu pela possibilidade de requisitos serem adicionados ao longo do caminho, já que, a princípio, o cliente pode não ter todos muito bem definidos. Além disso, outras demandas podem surgir conforme o desenvolvimento avança e o sistema é testado, então algumas funcionalidades podem ser adicionadas conforme necessárias.

Em suma, o método escolhido visa o trabalho em conjunto entre as partes interessadas para que o sistema tenha a melhor performance e que, por fim, todas as funcionalidades possam ser abarcadas.

5. Identificação dos riscos e estratégias

O projeto de um software envolve muitas variáveis que podem apresentar riscos ao desenvolvimento do programa. Os riscos que mais se destacam estão listados a seguir, juntamente com a estratégia mais adequada para mitigar tais ameaças.

1) **Requisitos mal definidos/mudança de requisitos:** a fraca definição dos requisitos, ou sua mudança tardia, pode ter alto custo, pois gera grande perda de trabalho e, consequentemente, perda de esforços e tempo gastos. Para lidar com esse possível risco, a escolha da metodologia é essencial para que não haja nenhum problema. A metodologia escolhida, incremental, toma partido de incorporar requisitos e funcionalidade ao longo do projeto, ou seja, uma boa estratégia para lidar com requisitos que possam surgir mais tarde no desenvolvimento;
2) **Alterações no escopo do projeto:** para evitar riscos com alterações de escopo, é muito importante que este seja muito bem definido desde o início do projeto. Por isso, definir os limites do projeto e elencar os itens que estão dentro do escopo (requisitos, funcionalidades etc.) é muito importante para que não haja nenhuma confusão em relação ao que deve ou não ser abarcado dentro do software que está sendo construído;
3) **Prazos não cumpridos:** a perda de prazos pode causar, também, altos custos ao projeto, o que pode ter um impacto negativo sobre a equipe e sobre a relação com o cliente. Uma estratégia a ser colocada em prática para que isso não aconteça é a definição de um cronograma com folgas e que antever possíveis imprevistos antes do começo do projeto, elencando quanto tempo cada fase terá e o que será feito em cada uma delas. Além disso, deve-se ter atenção e monitoramento em todas as etapas, para se ter controle do andamento do projeto;
4) **Orçamento excedido:** exceder o orçamento original é algo que pode gerar muitos impactos negativos no projeto. Para que isso não ocorra, é necessário fazer um planejamento financeiro detalhado, com previsão do que será gasto em cada etapa e que inclua margens de erros, a fim de que o orçamento seja respeito do início ao fim;
5) **Qualidade do código:** a qualidade do código é de fundamental importância, pois é um dos pilares para se ter um *software* de qualidade. Para que não haja riscos da qualidade do código ser impactada, devem ser feitas constantes revisões em cada estágio de entrega, para que se tenha

conhecimento do que foi feito até então. Além disso, uma boa documentação é essencial para que erros não sejam carregados para etapas futuras;

6) **Dificuldades com o uso do novo sistema:** implantar um novo sistema é desafiador, pois as novas funcionalidades podem não ser tão intuitivas a princípio. Para que isso não se torne um grande problema, os usuários finais do software devem ser treinados para o uso do novo sistema. Em complemento à capacitação, é essencial uma documentação completa de todas as funcionalidades para que os usuários tenham acesso e possam consultar se ocorrer dúvidas. E, por fim, também é muito importante o suporte da equipe de desenvolvimento pós implantação, dando o apoio necessário;

Estes são os principais riscos elencados que podem vir a comprometer o desenvolvimento e implantação de um novo software. Além do que foi listado, é necessário prever uma boa gestão de riscos, para que quando um novo risco for identificado, a gerência do projeto ter condições, juntamente com a equipe, de tomar a melhor decisão para que o projeto não seja prejudicado.

6. Recursos necessários

6.1. Recursos humanos

1) **Gerente de projeto:** responsável pelo planejamento e gerenciamento do projeto. Ele é quem elaborará o plano de negócio, com cronograma, orçamento e recursos necessários, irá identificar e gerenciar os riscos que possam surgir ao longo do desenvolvimento, elaborar o plano de desenvolvimento, comunicar as partes interessadas para manter todos atualizados quanto o status do projeto e garantir que o projeto seja entregue dentro do escopo, prazo e orçamento previstos;
2) **Analista de Sistemas:** responsável pela especificação de requisitos. Ele coletará as necessidades das partes interessadas para documentar os requisitos e a partir disso definir a arquitetura do projeto que guiará todo o seu desenvolvimento para atender as expectativas dos usuários;
3) **Desenvolvedores:** responsáveis pela implementação do projeto. Eles escreverão o código-fonte com base nas especificações, irão desenvolver todas as funcionalidades do projeto, além de realizar testes unitários para garantia da qualidade e colaborar entre si para integrar as partes do sistema;

4) **Designers (UI/UX):** responsáveis pela interface de usuário. Eles criarão os layouts, fluxos de trabalho e elementos visuais do projeto, para garantir que o sistema seja intuitivo e agradável para o usuário;
5) **Testadores (QA):** responsáveis pela qualidade do sistema. Eles desenvolverão os planos de testes, irão implementá-los e executá-los, a fim de identificar *bugs* e problemas no código feito e garantir a aceitação do usuário por meio de testes de interface que analisam se o usuário consegue desempenhar as tarefas para as quais o sistema foi desenvolvido.

6.2. Hardware e *software*

6.2.1. Hardware

1) **Servidor ou armazenamento em nuvem:** é preciso ter um servidor dedicado para hospedar o sistema e o banco de dados, ou utilizar serviços de armazenamento em nuvem confiáveis.

6.2.2. Software

2) **Linguagens de programação e Frameworks:** para o desenvolvimento da solução, podendo ser: *Java*, *C#*, *JavaScript*, *Python*, *ReactJS*, *NextJS*, *NodeJS*, *Spring* e *React Native* ou *Flutter* no caso de desenvolvimento mobile;
3) **Ambiente de desenvolvimento:** editores de código ou IDEs como *Visual Studio*, *Visual Studio Code*, *IntelliJ*, dentre outros;
4) **Banco de dados:** Sistema de gerenciamento de banco de dados como *MySQL*, *PostgreSQL*, dentre outros;
5) **Ferramentas de Controle de Versão:** para gerenciar o código fonte, podendo ser o *GitHub*;
6) **Ferramentas de Gerenciamento de Projeto:** soluções para registrar e rastrear as tarefas e progresso, como o *Jira*, *Trello*, *Stack* e outros.

6.3. Outros

6.3.1. Serviços de comunicação interna: para facilitar a comunicação entre as equipes, podendo utilizar soluções como: *Teams*, *Zoom*, *Skype*;

6.3.2. Recursos financeiros: Orçamento para custos relacionados ao desenvolvimento, como licenciamento de software, hospedagem e demais custos operacionais.

7. Requisitos/funcionalidades

O *Agriware* abrange desde registros de notas fiscais e controle de estoques até a geração de relatórios, análises e avisos, visando a uma gestão eficiente e organizada dos insumos agrícolas.

As principais funcionalidades propostas pelo *Agriware* são:

1) **Registro de notas fiscais:** o produtor poderá inserir as notas fiscais dos produtos adquiridos, possibilitando o armazenamento de informações sobre os insumos que estarão disponíveis para quando o produtor precisar;

2) **Geração de *QR Code*:** o sistema tem a funcionalidade de geração de *QR Code* das informações sobre insumos, equipamentos, sementes etc. para que possam ser acessadas por dispositivos móveis e por pessoas que não têm acesso ao sistema;

3) **Análise de compras:** com os dados de notas fiscais armazenados no sistema, essas informações poderão ser utilizadas para analisar as compras realizadas pelo produtor, gerando assim mais dados sobre os produtos comprados e, consequentemente, informações relevantes sobre as compras;

4) **Relatórios de fluxo de entrada e saída:** com as informações de entrada e saída sobre os insumos, o *Agriware* irá gerar relatórios organizados sobre as quantidades de insumos estocados, permitindo assim um controle sobre a quantidade de produtos armazenados;

5) **Controle de validade dos insumos:** o sistema terá controle sobre a validade de todos os insumos, como sementes, fertilizantes, defensivos, etc. Assim, serão gerados avisos sobre a suas validades, evitando a perda de insumos por vencimento;

6) **Controle de estoque mínimo e máximo:** o produtor poderá ter o controle sobre as quantidades mínima e máxima de estoque, desta forma, o *Agriware* irá gerar avisos quando esses limites estiverem próximos;

7) **Relatórios de compras e vendas de insumos:** o sistema irá gerar relatórios sobre as compras e vendas dos insumos, fornecendo assim informações importantes sobre os registros financeiros como custos e receitas dos produtos adquiridos;

8) **Relatórios sobre insumos mais e menos utilizados:** o sistema analisa os dados de uso dos insumos e gera relatórios que mostram quais são os insumos mais e menos utilizados. Com essas informações, o produtor poderá avaliar a eficiência de seus processos, ajustar estratégias de aquisição e identificar possíveis oportunidades de otimização;

9) **Controle de serviços terceirizados:** o sistema permite o registro de serviços terceirizados, como manutenção de máquinas e equipamentos. Isso ajuda o produtor a manter um controle organizado e atualizado desses serviços, facilitando o planejamento e a gestão dos recursos envolvidos;

10) **Relatórios de plantio e colheita:** o sistema irá gerar relatórios sobre o plantio e a colheita, permitindo ao produtor o acompanhamento do ciclo produtivo e planejamento da rotação de culturas;

11) **Relatórios de fluxo de caixa:** o sistema gera relatórios de fluxo de caixa, que apresentam informações sobre receitas e despesas relacionadas aos insumos agrícolas. Isso ajuda o produtor a ter uma visão clara da situação financeira, facilitando o controle e a tomada de decisões;

12) **Avisos e notificações:** o sistema emite avisos e notificações sobre diferentes aspectos, como falta de insumos, final de contrato de aluguel de equipamentos e outros eventos relevantes para a gestão do negócio agrícola. Essas notificações ajudam o produtor a estar ciente de eventos importantes e a tomar ações adequadas.

Além dessas funcionalidades, devemos incluir a implementação de um banco de dados, que é o primeiro dos objetivos específicos de (2).

8. Fases do ciclo de vida

Uma vez que optamos pelo **Modelo Evolutivo Incremental**, sabemos que tais funcionalidades devem ser categorizadas quanto à prioridade. A divisão idealizada foi em quatro grande fases de projeto:

1) **Fase 1 ou início:** a fase inicial abrange as funcionalidades 1, 2 e 3, além da implementação do banco de dados único, que constituem as funções mais básicas do *Agriware*;
2) **Fase 2 ou controle:** as funcionalidades 5, 6, 9 e 12 apoiam as funções básicas, aprimorando o controle de estoque e de sua validade, de serviços terceirizados e as notificações relacionadas a esse controle. Sendo assim, a fase dois está focada, principalmente, na expansão das atividades que o *software* ajuda a controlar (inclusão de serviços terceirizados e de validade dos insumos) e nas ferramentas para melhor visualização e aprimoramento desse controle. No cronograma presente em (9), as notificações adentram a fase três, pois consideramos que elas podem ser melhoradas quando a equipe definir melhor as demais saídas do sistema;
3) **Fase 3 ou relatórios:** essa fase se refere às saídas do sistema. As funcionalidades 4, 7, 8, 10 e 11 geram os mais diversos relatórios que irão apoiar a tomada de decisão dos usuários do sistema;
4) **Fase 4 ou dispositivos móveis:** por fim, adicionamos uma quarta fase, para uma possível implementação de um aplicativo para dispositivos móveis. Consideramos, no cronograma apresentado a seguir, que a fase quatro pode começar em paralelo com o final da fase três: uma segunda equipe de desenvolvimento pode, durante o sétimo trimestre do projeto, avaliar se é possível adaptar o *Agriware* para dispositivos móveis. Se essa possibilidade existir, do oitavo ao décimo trimestre, esse sistema será implementado para sistemas *Android* e *IOS*.

Destacamos que o cronograma foi dividido em períodos trimestrais, totalizando dois anos e meio (caso a implementação de aplicativos em dispositivo móvel seja considerada possível e viável).

9. Cronograma

Cronograma												
Fase	**Descrição da fase**	**Funcionalidades a serem implementadas**	**Tempo**									
			Tr. 1	**Tr. 2**	**Tr. 3**	**Tr. 4**	**Tr. 5**	**Tr. 6**	**Tr. 7**	**Tr. 8**	**Tr. 9**	**Tr. 10**
1 (Início)	Implementação das funcionalidades básicas do SI	Implementação do banco de dados único	■	■	■							
		Registro de notas fiscais		■	■							
		Geração de QR Code		■	■							
		Análise de compras		■	■							
2 (Controle)	Implementação das funcionalidades relacionadas ao controle e à emissão de alertas que ajudam nesse controle	Controle de validade dos insumos				■	■					
		Controle de estoque mínimo e máximo				■	■					
		Controle de serviços terceirizados				■	■					
		Avisos e notificações				■	■	■	■			
3 (Relatórios)	Implementação das funcionalidades relacionadas às demais saídas do SI (os relatórios)	Relatórios de fluxo de entrada e saída						■	■			
		Relatórios de compras e vendas de insumos						■	■			
		Relatórios sobre insumos mais e menos utilizados						■	■			
		Relatórios de plantio e colheita						■	■			
		Relatórios de fluxo de caixa						■	■			
4 (Dispositivos móveis)	Possibilidade de expansão para dispositivos móveis e implementação	Expansão para aplicativos em dispositivos móveis							■	■	■	■

10. Referências bibliográficas

BARBOSA, E. F. Material didático apresentado na disciplina SCC0532 - Metodologia de Desenvolvimento de Software. São Carlos/SP. s.d.;

10 EXEMPLOS de riscos de um projeto de software: conheça e previna-se! Disponível em: <https://nerus.com.br/blog/processos/exemplos-riscos-projeto-sofware/>. Acesso em: 04 de out. de 2023;

6 RISCOS gerenciais do desenvolvimento de software. Disponível em: <https://www.gobacklog.com/blog/riscos-gerenciais-desenvolvimento-software/>. Acesso em: 04 de out. de 2023;

QUAIS são os principais riscos em um projeto de software? Disponível em: <https://www.teclogica.com.br/riscos-em-um-projeto-de-software/>. Acesso em: 04 de out. de 2023;

GERENCIAMENTO de risco no desenvolvimento de softwares. Disponível em: <https://www.locaweb.com.br/blog/temas/codigo-aberto/gerenciamento-de-risco-no-desenvolvimento-de-softwares-como-fazer/>. Acesso em: 04 de out. de 2023.