**ENGENHARIA DE SOFTWARE**
2° Série – Noturno

André Luiz Souza Volpato – RA: 21011458-2

José Ricardo Miessi Gomes – RA: 21105683-2

Leonardo Losso Szychta – RA: 21088003-2

Renan dos Santos Dias – RA: 210009712-2

# DESENVOLVIMENTO DE SOFTWARE PARA AUTOMATIZAÇÃO DE CLIMATIZAÇÃO E DE ALERTA

ATIVIDADE DE ESTUDO PROGRAMADA DO 4º BIMESTRE

MARINGÁ
2021

# DESENVOLVIMENTO DO SOFTWARE

## INTRODUÇÃO

A carne de frango é considerada importante alimento para o consumo interno e para as exportações. O brasil atingiu a quarta colocação no ranking mundial de produção de frango, com 5.6% do total em 2020. Tal número é tão importante que reflete no PIB (Produto Interno Bruto) brasileiro, em 2020 o PIB do frango teve um crescimento de quase 10% em relação a 2019.

No entanto, não é somente números que são procurados, valoriza-se também à alta tecnologia envolvida nas granjas brasileiras e consequentemente o aumento na qualidade de vida e o bem-estar do animal de corte.

Com o constante crescimento desse mercado, o custo de produção aumenta, juntamente com o custo das tecnologias envolvidas, tendo isso em mente, nosso software visa ser um custo-benefício aos pequenos e médios produtores, entregando funcionalidade semelhantes de outros softwares, porém, com um custo mais acessível.

Após pesquisas e entrevistas que fizemos na região com produtores da área de avicultura de corte, já houve percas das aves por altas temperaturas, no caso, o equipamento de resfriamento não foi acionado e por falta de atenção dos funcionários (que não checaram os equipamentos), houve essas percas.

Buscando solucionar esse problema, surge o nosso software ‘’Gallanis’’ para automatizar esses equipamentos e alertar o proprietário sobre qualquer desequilíbrio no ambiente da granja e qualquer problema com os equipamentos.

## OBJETIVO

Melhorar e automatizar as granjas, visando reduzir perdas em relação as temperaturas excessivas, consumo excessivo de energia em momentos que poderiam ser reduzidos sem afetar a produtividade da granja.

Facilitar o monitoramento remoto da granja através de um aplicativo próprio, com sistemas de monitoramento da exaustão, aquecedores, controle de

umidade e o controle das luzes. Tudo isso feito automaticamente previamente programado de acordo com as necessidades do avicultor.

## JUSTIFICATIVA

Facilitar e automatizar serviços pertinentes a uma granja, reduzir custos, aumentar produtividade e consequentemente o lucro final do avicultor. Com uma rápida pesquisa pode-se perceber que as granjas que não tem serviços de automatizações estão mais suscetíveis a terem os mais variados problemas, baixa umidade e temperaturas acima de 40°C, com prejuízos na casa do milhões de reais.

## DESENVOLVIMENTO

Diante dos problemas apresentados anteriormente, consideramos que é de suma importância a solução deste, é um problema que não é visto somente em Maringá, mas em todas as granjas do Brasil.

Tendo em vista os benefícios da automatização e monitoramento remoto de uma granja, nosso grupo trabalho em uma ferramenta que auxilia e facilita ao máximo a vida do avicultor.

## 1) PROCESSOS DE SOFTWARE

Os modelos mais comuns de processos de softwares que se tem em uso hoje em dia são: cascata, incremental, espiral e reuso.

Modelo em cascata: Este modelo sugere uma abordagem sequencial e sistemática para o desenvolvimento de software. Dessa forma, começamos com o levantamento de requisitos ou necessidades junto ao cliente, depois vamos para a fase de planejamento onde definiremos as estimativas, cronograma e acompanhamento, após isso partiremos para a modelagem onde fazemos a análise e projeto, seguido da construção onde codificamos e testamos, passamos para a implantação ou emprego onde efetuamos a entrega, suporte e feedback do software concluído. Contudo este modelo não é flexível o que acaba

causando alguns problemas ao projeto, após o término de uma fase o resultado obtido não poderá ser alterado, tendo isso em mente é preciso ter uma experiência considerável com o levantamento de requisitos, tendo em vista que provavelmente o cliente terá dificuldade em fazer esse levantamento sozinho. Raramente um projeto real sempre seguirá o fluxo que o modelo propõe.[1][2]

Modelo Incremental: Combina elementos do modelo em cascata (aplicado repetidamente) com a filosofia iterativa da prototipação, o objetivo deste modelo é trabalhar junto ao usuário para descobrir seus requisitos, de maneira incremental até que o produto final seja obtido. Neste modelo, o desenvolvimento é repartido em várias pequenas partes chamada de incremento e tende a priorizar os requisitos dos usuários.

O primeiro incremento pode ser chamado de "núcleo do produto" e contém a implementação dos requisitos básicos para que o sistema possa funcionar e atender as minimamente as necessidades do cliente.

> Quando se utiliza um modelo incremental, frequentemente o primeiro incremento é um produto essencial. Ou seja, os requisitos básicos são atendidos; porém, muitos recursos complementares (alguns conhecidos, outros não) ainda não são entregues. Esse produto essencial é utilizado pelo cliente (ou passa por uma avaliação detalhada). (PRESSMAN, 2016, p. 72).[3]

Cada aprimoramento, nesse modelo, é lançado como uma versão. Novas versões são criadas até que o sistema esteja completo e adequado para que seja lançada a versão final, que atendera os requisitos dos clientes.

No Modelo Incremental, as atividades da Especificação, Projeto, Implementação e Validação ocorrem intercaladamente, acontecendo em cada nova versão, isso permite um rápido feedback em todas as atividades.

Contudo essa divisão em pequenos incrementos traz consigo dificuldades em definir as necessidades dos usuários e é de suma importância que o projeto inteiro esteja definido, para quando ocorrer a junção desses pequenos incrementos não resulte em um sistema com uma arquitetura pobre.

---

[1] Disponível em: <https://www.portaleducacao.com.br/conteudo/artigos/biologia/modelos-de-processo-de-softwares/53061#>. Acesso em 16 set. 2021.
[2] Disponível em: <https://edisciplinas.usp.br/pluginfile.php/839466/mod_resource/content/1/Aula02_ModelosProcessos.pdf>. Acesso em 16 set. 2021.
[3] PRESSMAN, Roger S. Engenharia de Software: Uma abordagem profissional. 2016. 8° Edição. Acesso em 16 set. 2021.

Os clientes podem estabelecer as prioridades das partes do sistema a serem desenvolvidas, especificando as mais úteis primeiro. Após o cliente estabelecer as funcionalidades necessárias, são criados os estágios (incrementos) de entrega, onde cada estágio fornece um conjunto de funcionalidades do sistema. Recebendo a cada versão lançada um feedback do cliente, para analisar se está de acordo com o que foi pedido.[4]

Modelo Espiral: Consiste em realizar as fases de desenvolvimento (especificação, análise, projeto, etc..) repetindo sistematicamente 4 atividades para cada uma das fases, sendo elas:

A) Definição dos objetivos;

B) Avaliação de redução de riscos;

C) Desenvolvimento e validação;

D) Planejamento;

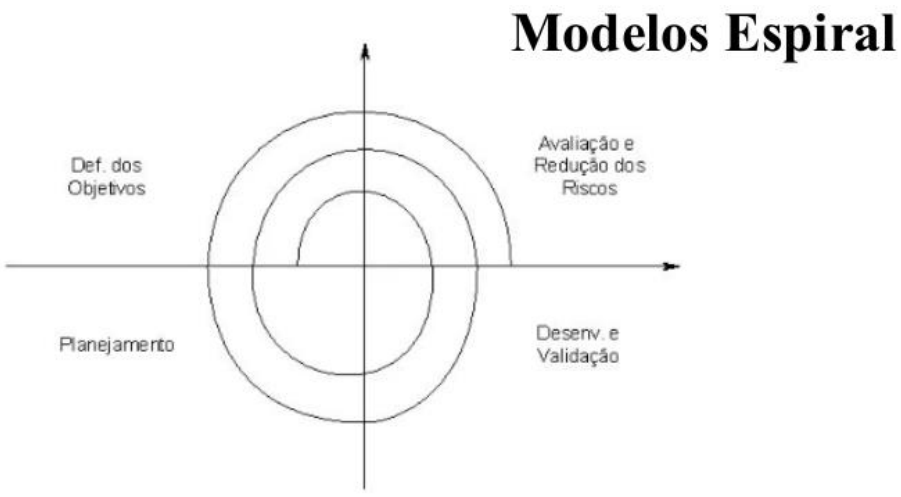
Cada loop da espiral representa uma fase do processo de desenvolvimento:

1° loop – Viabilidade do projeto.

2° loop - Referência dos requisitos / análise.

3° loop – Projeto lógico

4° loop – Projeto físico




Neste modelo, define-se os objetivos específicos para a fase, identificam-se as restrições para o processo de desenvolvimento da fase e para os produtos,

[4] Disponível em: <https://medium.com/contexto-delimitado/o-modelo-incremental-b41fc06cac04>. Acesso em 16 set. 2021.

prepara-se um plano para o controle e gerenciamento da fase e identificam-se os riscos do projeto. É um ótimo modelo para conter os riscos pois faz-se uma análise dos riscos identificados e providenciam-se ações para a redução desses riscos.[5]

Modelo Reuso: Este modelo reutiliza códigos, funções e classes pré-existentes de softwares, sistemas já existentes para fazer um novo sistema/software. Mas como eles reutilizam esses códigos? Existe algumas "bibliotecas" de códigos com códigos existentes que foram criados para resolver algum tipo de problema e que um desenvolvedor pode utilizar no próprio sistema, só precisa modificar para ficar do modo que necessita. Geralmente essa reutilização pode ocorrer informalmente, no qual o desenvolvedor que está trabalhando em um projeto novo já utilizou ou conhecem programas e/ou códigos semelhantes ao que é requisitado e os modificam conforme o sistema necessite.[6]

Contudo, um ponto negativo deste modelo: O custo de manutenção pode ser maior do que os outros modelos, pois se o código-fonte de um sistema ou os componentes do software reusáveis não estiverem disponíveis, os custos da manutenção irão ser maiores, pois os elementos reusados podem ser tornar cada vez mais incompatíveis com as alterações feitas no sistema.

Para o presente projeto escolhemos o modelo em cascata como processo de software pois, como é um projeto pequeno e não será um sistema desenvolvido completamente podemos ficar em uma etapa por um tempo grande sem gerar problemas.

## 2) ARQUITETURA DE COMPUTADORES

Antes de começarmos a falar sobre a arquitetura de computadores, precisamos colocar em contexto como foi a evolução do computador, desde as primeiras máquinas de calcular até o que chamamos hoje de computador. Comecemos pelo básico que é a criação do ábaco em 2500 a.C, que servia para resolver operações matemáticas. Passados alguns anos, tivemos a invenção da

[5] Disponível em: <https://medium.com/contexto-delimitado/o-modelo-em-espiral-de-boehm-ed1d85b7df>. Acesso em: 16 set. 2021.
[6] Disponível em: https://medium.com/@mikiasoliveira/re%C3%BAso-de-software-8406766d9eb8. Acesso em 16 set. 2021.

régua de cálculo em 1633 que foi o primeiro computador analógico e poucos anos mais tarde em 1642, tivemos a invenção da primeira máquina de somar, a Pascalina, que executava operações aritméticas quando se giravam os discos interligados, é a percursora das calculadoras mecânicas. Passados 160 anos, mais especificamente no ano de 1802 foi inventado a máquina de cartões metálicos perfurados que servia para controlar e automatizar máquinas de tear, inventada por Joseph Marie Jacquard.

Anos mais tarde, em 1944 no quase fim da Segunda Guerra Mundial, foi inventado o COLOSSUS, um computador que usava 2.000 válvulas eletrônicas, inventado por Alan Turing, que é considerado hoje em dia o pai da computação. As coisas começaram a andar melhor a partir de 1981, com o primeiro IBM PC, a partir daí a evolução não parou mais e continua até hoje evoluindo.

Para um computador, tudo são números e é utilizado o sistema binário - (bit que vem do Binary digIT). Eles representam informações utilizando dois estados possíveis, 1 – ligado e 0 – desligado. O número binário é uma unidade de informação, uma quantidade computacional que pode tomar um de dois valores tais como verdadeiro e falso ou 1 e 0, e assim por diante. Os sistemas são formados por vários circuitos que passam informações em forma de correntes, ligadas ou desligadas (1 ou 0), respectivamente. Os bits tem uma capacidade de representação e há a necessidade de representar símbolos utilizadas nas linguagens escritas, como por exemplo na tabela abaixo:

Capacidade de representação:

| Bits | Símbolos |
|---|---|
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |
| **8** | **256** |
| 9 | 512 |
| 10 | 1024 |

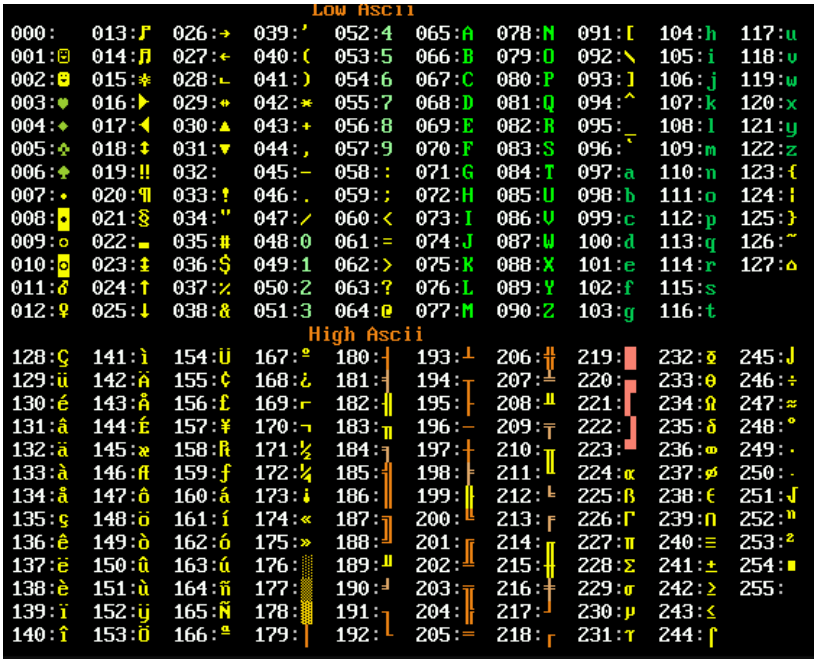
A junção de 8 bits é chamada de 1 BYTE (BInary TErm), é tratado de forma individual como uma unidade de armazenamento. Um dos sistemas mais

importantes que temos para realizar representações de símbolos com números binários é a tabela ASCII (American Standart Code for Information – Código padrão Americano para o Intercâmbio de Informações), representado pela tabela abaixo:

| BINARIOS | DECIMAL | PALAVRA |
|---|---|---|
| 00000 | 0 | |
| 00001 | 1 | A |
| 00010 | 2 | B |
| 00011 | 3 | C |
| 00100 | 4 | D |
| 00101 | 5 | E |
| 00110 | 6 | F |
| 00111 | 7 | G |
| 01000 | 8 | H |
| 01001 | 9 | I |
| 01010 | 10 | J |
| 01011 | 11 | K |
| 01100 | 12 | L |
| 01101 | 13 | M |
| 01110 | 14 | N |
| 01111 | 15 | O |

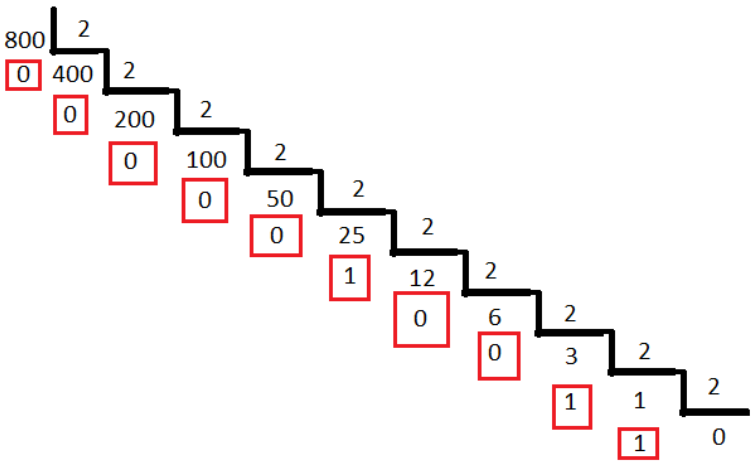
| BINARIOS | DECIMAL | PALAVRA |
|---|---|---|
| 10000 | 16 | P |
| 10001 | 17 | Q |
| 10010 | 18 | R |
| 10011 | 19 | S |
| 10100 | 20 | T |
| 10101 | 21 | U |
| 10110 | 22 | V |
| 10111 | 23 | W |
| 11000 | 24 | X |
| 11001 | 25 | Y |
| 11010 | 26 | Z |
| 11011 | 27 | ? |
| 11100 | 28 | ! |
| 11101 | 29 | . |
| 11110 | 30 | , |
| 11111 | 31 | ; |

A tabela acima mostra os valores na forma binária e a tabela abaixo mostra os valores na forma decimal, já com a devida conversão de binário para decimal, conversão essa que veremos logo a seguir:




Para fazermos a conversão de números decimais para binário é possível fazer utilizando o método matemático que consiste em divisões sucessivas,

deve-se utilizar a divisão com resto. Podemos utilizar de exemplo o número $800_{10}$:




O resultado em binário lê-se da direita pra esquerda, de baixo para cima, ficando assim em binário: 1100100000. Para fazer a conversão inversa fica da maneira que mostraremos na figura a seguir:

| 2048 | 1024 | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $2^{11}$ | $2^{10}$ | $2^{9}$ | $2^{8}$ | $2^{7}$ | $2^{6}$ | $2^{5}$ | $2^{4}$ | $2^{3}$ | $2^{2}$ | $2^{1}$ | $2^{0}$ |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

Colocando os números em binário nas suas respectivas casas, soma-se o valor de cada casa com as que estiverem preenchidas com o valor 1, resultando em: 512 + 256 + 32 = 800, tem-se o valor em decimal novamente.

O Mapa de Karnaugh, é um método de simplificação da expressão booleana, sendo que cada célula do mapa representa uma linha da coluna de resultado de uma tabela-verdade, com isso, reduz a expressão booleana em sua forma mais simplificada, podendo ser aplicada para mais de uma variável.

Para cada tipo de tabela-verdade o mapa é feito de uma forma diferente, ou seja, quando na tabela-verdade há duas entradas, o mapa é feito de uma forma, quando há três entradas na tabela, o mapa é feito de outra forma, conforme vai aumentando o número de entradas, vai mudando junto a forma que o mapa é feito. Para cada tipo de tabela-verdade há uma forma específica para se fazer o Mapa de Karnaugh.

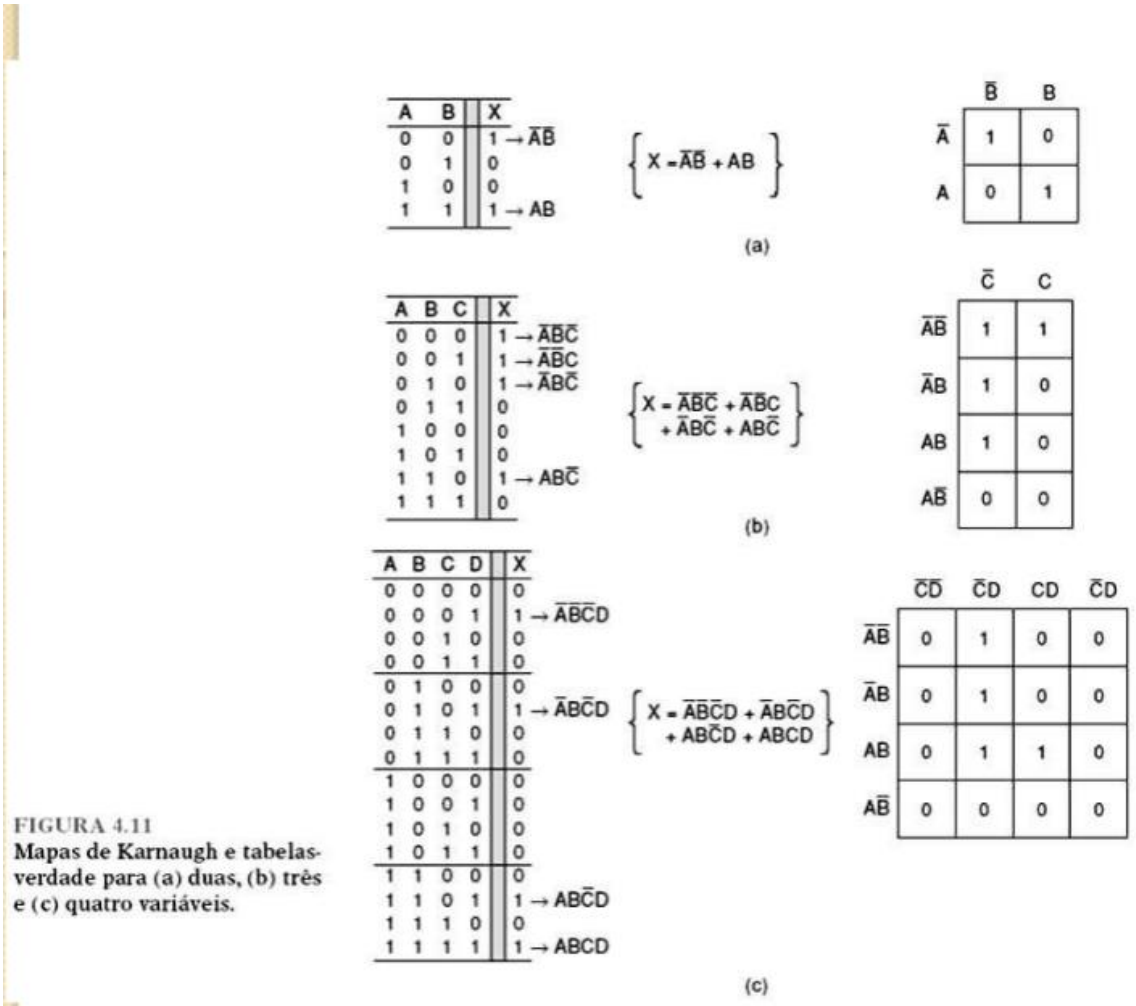



FIGURA 4.11
**Mapas de Karnaugh e tabelas-verdade para (a) duas, (b) três e (c) quatro variáveis.**

## 3) ALGORITMOS E LÓGICA DE PROGRAMAÇÃO

A base de todo software seja de computador, smartphone, tecnologias embarcadas, o que for, a base por trás de todas as linhas de código é a lógica de programação. O algoritmo é uma sequência de processos de cálculos, um sistema complexo de etapas que ao final delas resultará na solução de um problema.[7]

Em quanto a lógica de programação é conhecer o significado das palavras da linguagem de programação, é como se fosse uma sintaxe, sendo assim, ambas estão profundamente ligadas. É importante aprender lógica de

[7] Disponível em: <https://acadtec.com.br/blog/desenvolvimento-backend/qual-a-importancia-de-algoritmos-na-programacao>. Acesso em: 16 set. 2021.

programação pois ela ensina a pessoa a pensar sobre como resolver problemas, nos ensinando a “quebrar” o problema em partes até chegar ao seu núcleo, para assim, podermos começar a elaborar um algoritmo que resolva esse problema. O raciocínio lógico é a ferramenta fundamental do desenvolvedor, já que uma máquina é incapaz de compreender ordens não lógicas.[8]

Dentro de algoritmos e lógica de programação temos que nos familiarizar com alguns aspectos, sendo alguns deles: variáveis e constantes, tipos de dados (inteiro, real, lógico, etc.), estrutura de repetição e de seleção e vários outros. É fundamental que o desenvolvedor saiba esses conceitos para que ele possa criar softwares e resolver problemas. Para um bom desenvolvedor não basta somente saber os conceitos ao pé da letra, é preciso também que o código esteja limpo e entendível e não somente fazer um código para si mesmo, onde a única pessoa que conseguirá dar a manutenção do código é você mesmo, é preciso com que outros desenvolvedores consigam fazer a manutenção do mesmo.[9]

Temos que entender também o conceito de procedimentos que diferem do conceito de funções apenas por não retornarem nenhum resultado. Já as funções, não precisa copiar um código inteiro para se fazer uma operação, basta chamar a função que ela retornará à operação solicitada. As funções reduzem o tamanho de um algoritmo, facilitam a compreensão e visualização do algoritmo, são declarados no início do algoritmo e podem ser chamadas em qualquer ponto após sua declaração.

## 4) MATEMÁTICA PARA COMPUTAÇÃO

Como bem sabemos a matemática para computação é uma base fundamental para diversas disciplinas e ciências atuais, podemos citar algumas como: Sistemas operacionais, banco de dados, compiladores, estrutura de dados, técnicas digitais, algoritmos e diversas outras. Graças a matemática, desenvolvedores podem criar softwares que não apenas sejam fiéis à realidade

[8] Disponível em: <https://kenzie.com.br/blog/logica-de-programacao/>. Acesso em: 16 set. 2021.

[9] Disponível em: <https://www.devmedia.com.br/logica-uma-ferramenta-indispensavel-na-programacao-de-computadores/28386>. Acesso em: 16 set. 2021.

como também funcionam eficientemente. É também graças a ela que temos a lógica, a fundamentação mais básica da programação, sem ela não teríamos sequer um computador.

Um dos ensinamentos da matemática para computação são as fórmulas *WFF (Well-formed Formulas) of Propositional Logic - Fórmulas bem formuladas.* Utiliza-se da lógica proposicional e faz o uso de símbolos como linguagem para representar a lógica estrutural ou a fórmula de uma proposição composta. Como em qualquer outra língua, essa linguagem tem regras e sintaxe gramaticais para colocar os símbolos juntos e nos lugares corretos.[10]

Já estabelecido a importância da matemática e das WFF e com base no problema proposto pela AEP, verificamos em pesquisas por fatos ocorridos onde a falta de automatização trouxe prejuízo para o avicultor. Uma dessas pesquisas nos levou a achar uma reportagem que dizia que mais de mil aves morreram em granjas por conta do calor na cidade de Pratápolis em Minas Gerais. O prejuízo é de mais de R$5.000,00 reais e foram provocadas após um pico de energia que comprometeu o controle da temperatura, mesmo tendo um gerador de energia não foi o suficiente para manter todos os exaustores e nebulizadores ligados. Tal problema se tivesse redundância suficiente poderia ter alertado o avicultor ou os operadores da granja, evitando prejuízos financeiros.[11]

Com base nisso, estabelecemos as seguintes fórmulas para efetuar a resolução do problema:

Para toda granja de frango, se as condições estão ideais então os frangos ficam vivos e saudáveis

x → Granja de frango

C→ Condições ideais

F→ frangos

V→VIVOS

S→ Saudáveis

[10] Disponível em: <http://www.skillfulreasoning.com/propositional_logic/well-formed_formulas.html>. Acesso em: 16 set. 2021.

[11] Disponível em: https://dl.acm.org/doi/10.5555/1074100.1074917. Acesso em: 16 set. 2021.

(∀x) C(x)→ F(x)V(x)^S(x)

Se a temperatura aumenta muito então os frangos morrem. Se a temperatura despenca então os frangos morrem. Se estiver nas condições ideais, então os frangos vivem.

T→ Temperatura aumenta muito

G→Temperatura despenca

M→Morre

F→Frango

V→VIVEM

C→Condições Ideais

(T→FM)(G→FM)(C→FV)

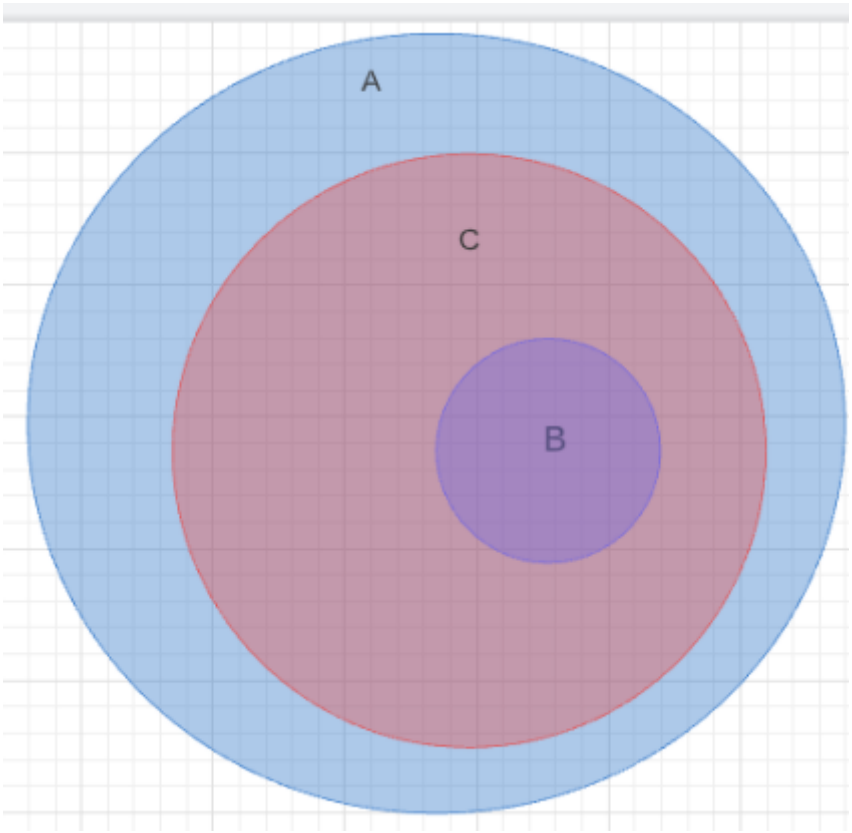
O diagrama de Venn é uma representação visual das similaridades e diferenças entre dois ou mais itens. Ele consiste na sobreposição de uma série de formas, normalmente círculos.

Também conhecidos como diagramas de conjuntos ou diagramas lógicos, são amplamente utilizados na matemática, estatística, lógica, ensino, linguística, ciência da computação e nos negócios.

Os diagramas são usados para estabelecer a validade de determinados argumentos e conclusões. Em um raciocínio dedutivo, se as premissas são reais e a forma de argumentação está correta, então a conclusão deve ser verdadeira.[12] Por exemplo, se todos os cães são animais, e nosso mascote Mojo é um cão, então Mojo tem que ser um animal. Se atribuirmos variáveis, digamos que os cães são C, os animais são A e Mojo é B. Em forma de argumento, dizemos: todos os C são A. B é C. Portanto, B é A.

Representando isso no diagrama de Venn fica da seguinte forma:

[12] Disponível em: https://www.voitto.com.br/blog/artigo/diagrama-de-venn. Acesso em 14 de nov. 2021.




**5) ENGENHARIA DE SOFTWARE**

Com base no que vimos em aulas, o engenheiro de software tem que ser uma pessoa ética e com muitas responsabilidades a serem carregadas. Em uma entrega de projeto, fazer somente o requisitado pois, temos de apresentar ao cliente o que ele deseja, se tivermos alguma ideia para inserir no projeto, devemos apresentá-la ao cliente para somente depois, se ele quiser, inserir no projeto.

É importante para o engenheiro de software ter uma comunicação, é simplesmente inviável um engenheiro de software não ter uma boa comunicação, uma soft skill liga na outra, uma não consegue sozinha ser boa suficiente para compensar a falta das outras. No geral é importante desenvolver todas as soft skills possíveis, não somente para se tornar um bom profissional, mas para também desenvolver habilidade interpessoais e para que o projeto que esteja sendo desenvolvido fique sempre alinhado com as expectativas da equipe que trabalha no mesmo.

Para ser um bom engenheiro de software é preciso também ter bons conhecimentos na elicitação de requisitos, essa é a fase onde o engenheiro vai

até o cliente e por meio de uma série de perguntas ele busca informações sobre o que o cliente quer que seja construído em seu sistema. É nessa fase que o engenheiro de software deve entender e compreender o que o cliente quer, quais as suas regras de negócio, funcionalidades do sistema, usabilidade do software, acessibilidade entre muitos outros requisitos.

Apesar de parecer fácil fazer o levantamento dos requisitos, essa etapa é extremamente complicada, pois o cliente tem um problema que é passado ao engenheiro e ele receba a função de corrigir esse problema. Também é nessa mesma fase que acontece desentendimentos com o cliente, onde ele sabe bem o que quer para o seu sistema, mas não sabe a forma correta de passar isso ao engenheiro de uma forma compreensível ou ele não sabe bem o que o sistema dele precisa, é aqui que um bom engenheiro saberá resolver esse conflito e explicar exatamente o que pode ser feito ao cliente.[13]

A fase de levantamento de requisitos pode ainda ser dividida entre duas outras fases: Requisitos funcionais e não funcionais.

Os requisitos funcionais de um sistema descrevem o que ele deve fazer. Dependem do tipo de software a ser desenvolvido, de quem são seus possíveis usuários e da abordagem geral adotada pelo engenheiro ao escrever os seus requisitos. Exemplo, um usuário deve ser capaz de pesquisar as listas de agendamentos para todas as clínicas, ou ainda, esse mesmo sistema das clínicas deve gerar um relatório dos clientes daquele dia.

Já nos requisitos não funcionais, são requisitos que não estão diretamente relacionais com os serviços específicos oferecidos pelo sistema a seus usuários. Eles podem estar relacionados as propriedades do sistema, como a confiabilidade, hardware a ser utilizado, banco de dados, e outros.

De acordo com Ian Sommerville:

> Os requisitos não funcionais, como desempenho, proteção ou disponibilidade, normalmente especificam ou restringem as características do sistema como um todo. Requisitos não funcionais são frequentemente mais críticos que requisitos funcionais individuais. Os usuários do sistema podem, geralmente, encontrar maneiras de contornar uma função do sistema que realmente não atenda as suas necessidades. No entanto, deixar de atender a um requisito não funcional pode significar inutilizar todo o sistema. Um exemplo crítico desses, pode ser citado o sistema de uma aeronave, se ele não cumprir

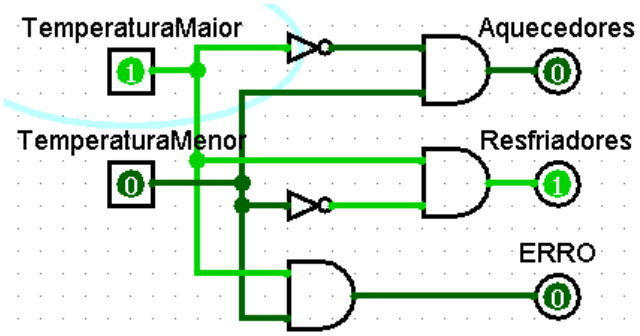
[13] Disponível em: https://www.devmedia.com.br/elicitacao-de-requisitos-levantamento-de-requisitos-e-tecnicas-de-elicitacao/31872. Acesso em: 05 de nov. de 2021.

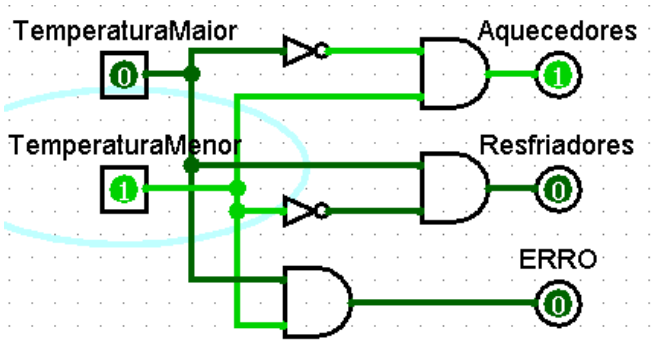
> seus requisitos de confiabilidade, não será certificado como um sistema seguro para operar.[14]

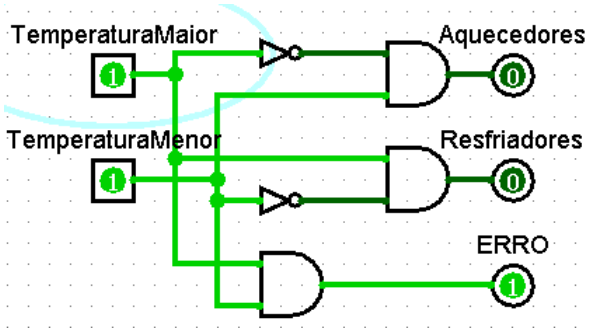
**SOLUÇÃO TÉCNICA:**

**1) ARQUITETURA DE COMPUTADORES**

Na parte de arquitetura de computadores, elaboramos um circuito lógico para as luzes, umidade e temperatura para tudo ficar de forma automatizada. No circuito lógico da temperatura, o resultado ficou da seguinte forma:

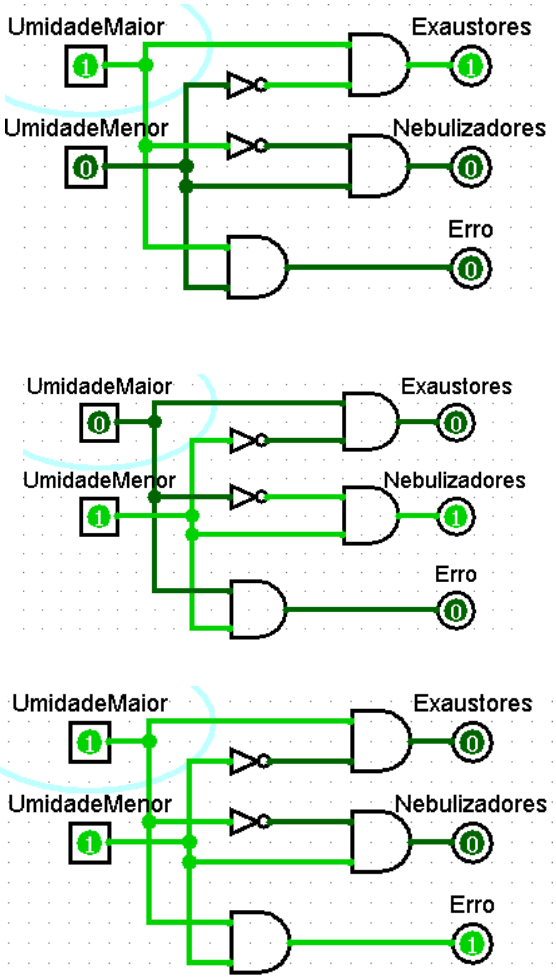


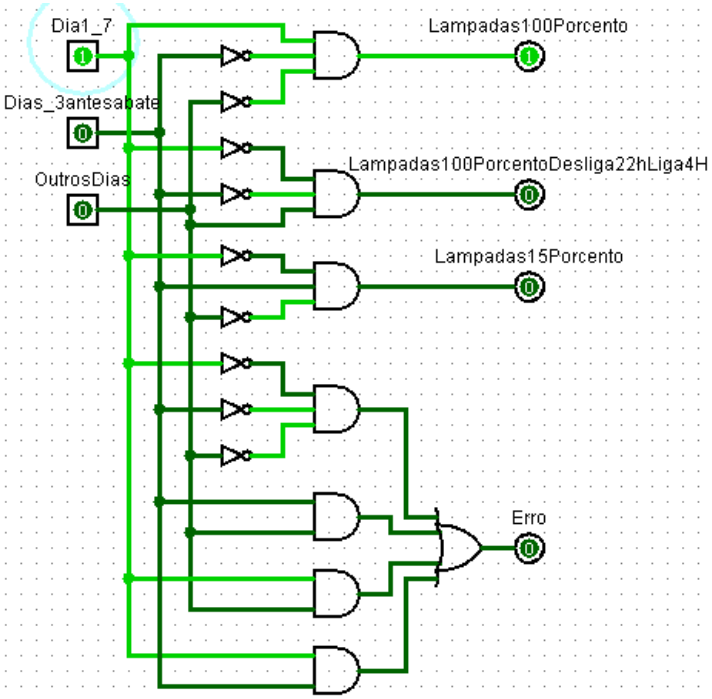



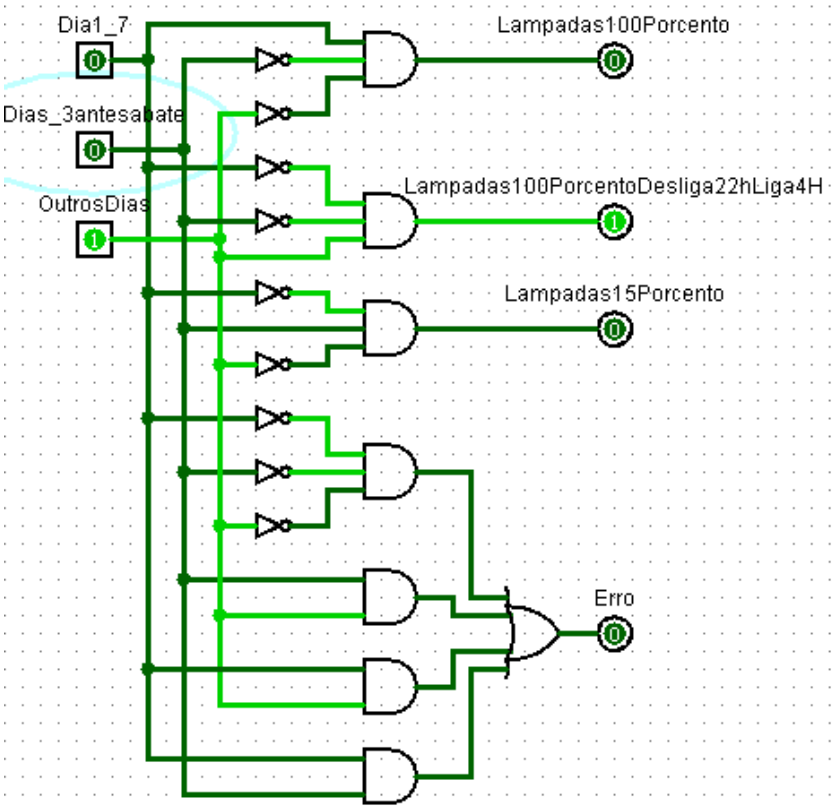

[14] Disponível em: SOMMERVILLE, Ian. Engenharia de Software. 2011. 9° Edição. Acesso em 05 nov. 2021.

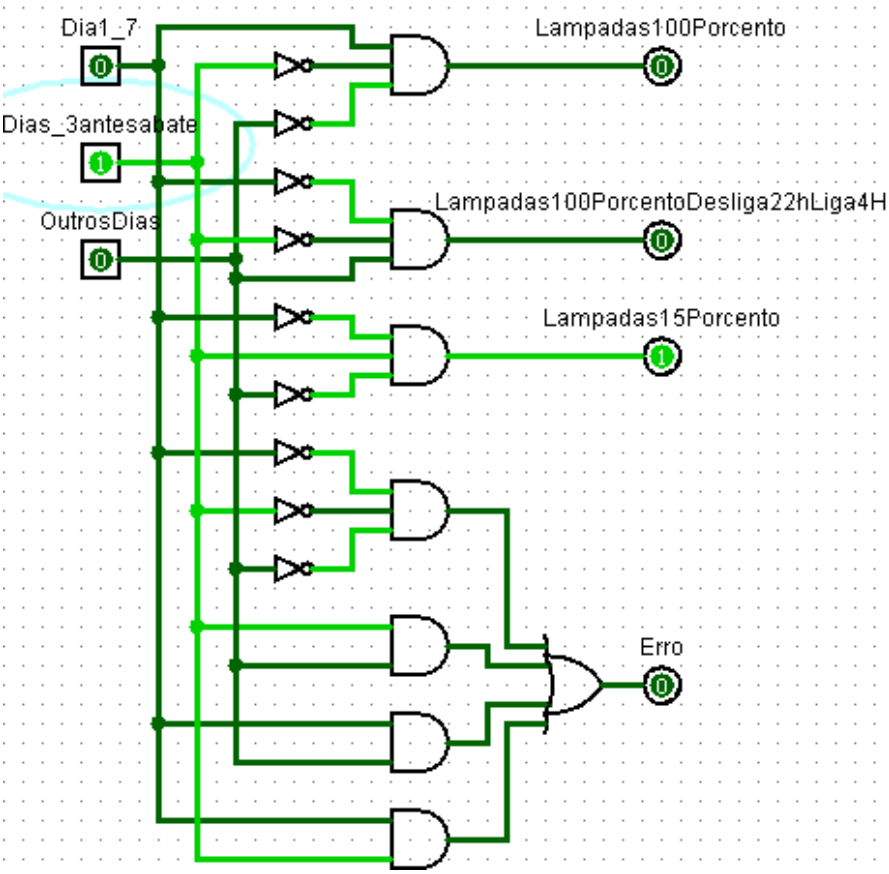
No circuito lógico da umidade da granja ficou da seguinte forma:

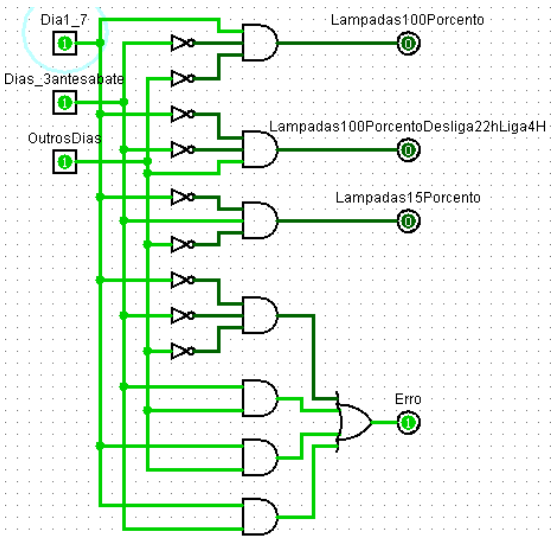



No circuito lógico das luzes, o resultado é o seguinte:




Dia1_7
Dias_3antesabate
OutrosDias
Lampadas100Porcento
Lampadas100PorcentoDesliga22hLiga4H
Lampadas15Porcento
Erro


Dia1_7
Dias_3antesabate
OutrosDias
Lampadas100Porcento
Lampadas100PorcentoDesliga22hLiga4H
Lampadas15Porcento
Erro




Usamos o Mapa de Karnaugh para fazer a simplificação do sistema de controle de luz, temperatura e umidade, para equilibrar o meio que as aves estão e quando precisar ligar, o acionamento será automático. O de luz tem três variáveis e o de temperatura e umidade são duas variáveis, ficando da seguinte forma:

**TEMPERATURA**

| Aquecedor | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 1 |
| A | 0 | 0 |
| R: A~B | | |

| Resfriador | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 0 |
| A | 1 | 0 |
| R:AB~ | | |

| Erro | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 0 |
| A | 1 | 1 |
| R: A | | |

**UMIDADE**

| Exaustores | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 0 |
| A | 1 | 0 |
| R: AB~ | | |

| Nebulizadores | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 1 |
| A | 0 | 0 |
| R: A~B | | |

| Erro | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 0 |
| A | 0 | 1 |
| R: AB | | |

**LUZ**

| 100% | | | | |
|---|---|---|---|---|
| | B~ | | B | |
| A~ | 0 | 0 | 0 | 0 |
| A | 1 | 0 | 0 | 0 |
| | C~ | C | | C~ |
| R:AB~C~ | | | | |

| 100% 24h | | | | |
|---|---|---|---|---|
| | B~ | | B | |
| A~ | 0 | 1 | 0 | 0 |
| A | 0 | 0 | 0 | 0 |
| | C~ | C | | C~ |
| R:A~B~C | | | | |

| 15% | | | | |
|---|---|---|---|---|
| | B~ | | B | |
| A~ | 0 | 0 | 0 | 1 |
| A | 0 | 0 | 0 | 0 |
| | C~ | C | | C~ |
| R:A~BC~ | | | | |

| Erro | | | | |
|---|---|---|---|---|
| | B~ | | B | |
| A~ | 1 | 0 | 1 | 0 |
| A | 0 | 1 | 1 | 1 |
| | C~ | C | | C~ |
| R:AC;AB; BC; A~B~C~ | | | | |

## 2) ALGORITMOS E LÓGICA DE PROGRAMAÇÃO

algoritmo "Automatização e alerta de aviário de corte"

var

diasfrango: inteiro
// Essa variável irá guardar quantos dias o frango ira ficar na granja
tempd0_d2,tempd3_d5,tempd6_d8,tempd9_d11,tempd12_d14: real
//essas variáveis irão guardar a temperatura (em graus celsius) que o usuário desejar.
tempd15_d17, tempd18_d20, tempd21_d27, tempd28_d34,tempd35_d41, tempd42_d48, tempd49_d50: real
//essas variáveis irão guardar a temperatura (em graus celsius) que o usuário desejar.
Umidade, umidadepainel, termometro: real
//Essas variáveis irão receber os valores que os sensores passarem para elas, e estarão passando constantemente para o sistema.

inicio
Repita
Escreva ("Informe quantos dias (em números) os frangos irão ficar na granja: ")
Leia(diasfrango)

Ate (diasfrango="0") e (diasfrango="1") e (diasfrango="2") e (diasfrango="3") e (diasfrango="4")
e (diasfrango="5") e (diasfrango="6") e (diasfrango="7") e (diasfrango="8") e (diasfrango="9")

Escreva ("Informe a temperatura desejada do dia 00 ate dia 02: ")
Leia(tempd0_d2)
Escreva ("Informe a temperatura desejada do dia 03 ate dia 05: ")
Leia(tempd3_d5)
Escreva ("Informe a temperatura desejada do dia 06 ate dia 08: ")
Leia(tempd6_d8)
Escreva ("Informe a temperatura desejada do dia 09 ate dia 11: ")
Leia(tempd9_d11)
Escreva ("Informe a temperatura desejada do dia 12 ate dia 14: ")
Leia(tempd12_d14)
Escreva ("Informe a temperatura desejada do dia 15 ate dia 17: ")
Leia(tempd15_d17)
Escreva ("Informe a temperatura desejada do dia 18 ate dia 20: ")
Leia(tempd18_d20)
Escreva ("Informe a temperatura desejada do dia 21 ate dia 27: ")
Leia(tempd21_d27)
Escreva ("Informe a temperatura desejada do dia 28 ate dia 34: ")
Leia(tempd28_d34)
Escreva ("Informe a temperatura desejada do dia 35 ate dia 41: ")
Leia(tempd35_d41)
Escreva ("Informe a temperatura desejada do dia 42 ate dia 48: ")
Leia(tempd42_d48)
Escreva ("Informe a temperatura desejada do dia 49 ate dia 50: ")
Leia(tempd49_d50)

//a variável "termômetro" está recebendo valores de um sensor que mede a temperatura do ambiente e passa para o sistema.

```
//aqui com o "randi(51)" estamos simulando um sensor passando um valor para a variável termômetro, que mede a temperatura do ambiente de 0 a 50 Graus Celsius.

Termometro<-RANDI(51)
se (termometro>tempd0_d2) entao
  Escreval ("ALERTA!! ALGO ESTÁ ERRADO! A TEMPERATURA ESTÁ ACIMA DO RECOMENDADO")
senao
  se (termometro<tempd0_d2) entao
    Escreval ("ALERTA!! ALGO ESTÁ ERRADO!A TEMPERATURA ESTÁ ABAIXO DO RECOMENDADO")
  fimse
fimse

//a variável "umidade" recebe um valor a partir de um sensor que está medindo a umidade do ambiente e passa o valor para o sistema.
//aqui com o "randi(101)" estamos simulando um sensor passando um valor para a variável umidade, onde o sensor mede a umidade do ambiente de 0 a 100%.

umidade<-RANDI(101)
Se (Umidade>50) entao
  Escreval("Confirmar o acionamento dos exaustores, o nivel de umidade dentro da granja está em: ", umidade, "%")
  Senao
   Se (umidade<40) entao
   Escreval("Confirmar o acionamento dos nebulizadores, o nivel de umidade dentro da granja está em: ", umidade, "%")
   Senao
    Se (Umidade=40) ou (Umidade=41) ou (Umidade=42) ou (Umidade=43) ou (Umidade=44) ou (Umidade=45) ou (Umidade=46) ou (Umidade=47) ou (Umidade=48) ou (Umidade=49)  entao
     Escreval ("Umidade está em um nível aceitável")
```

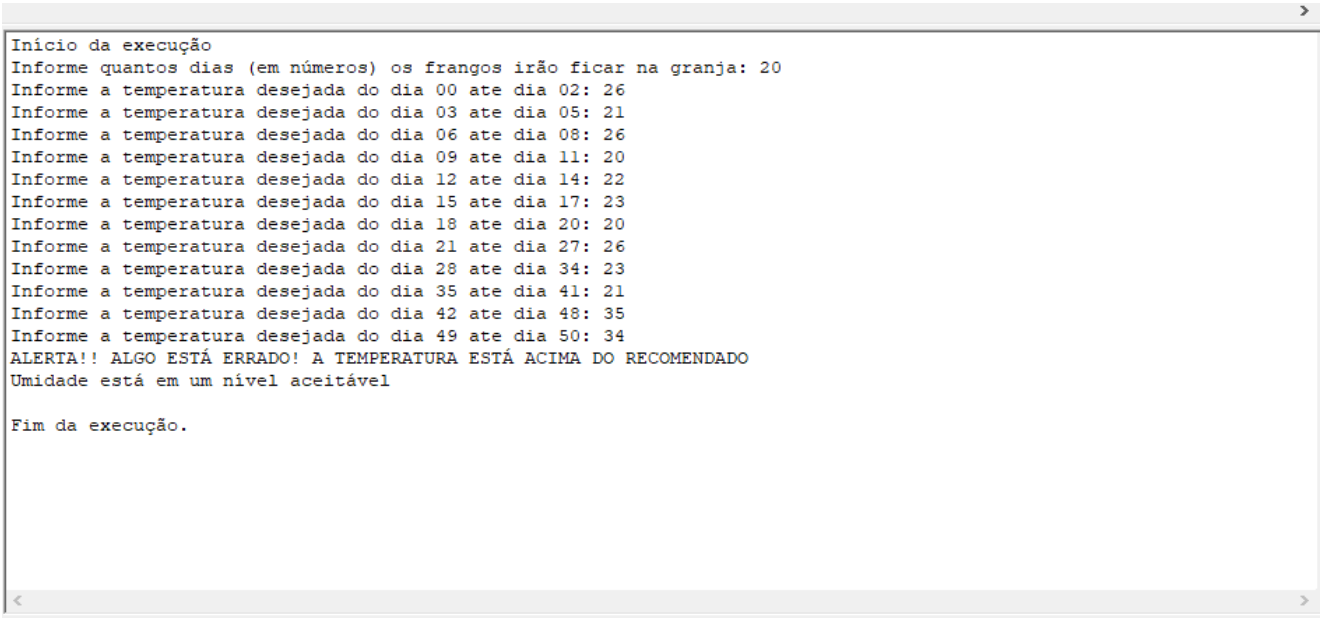
Fimse

Fimse

fimse

Fimalgoritmo

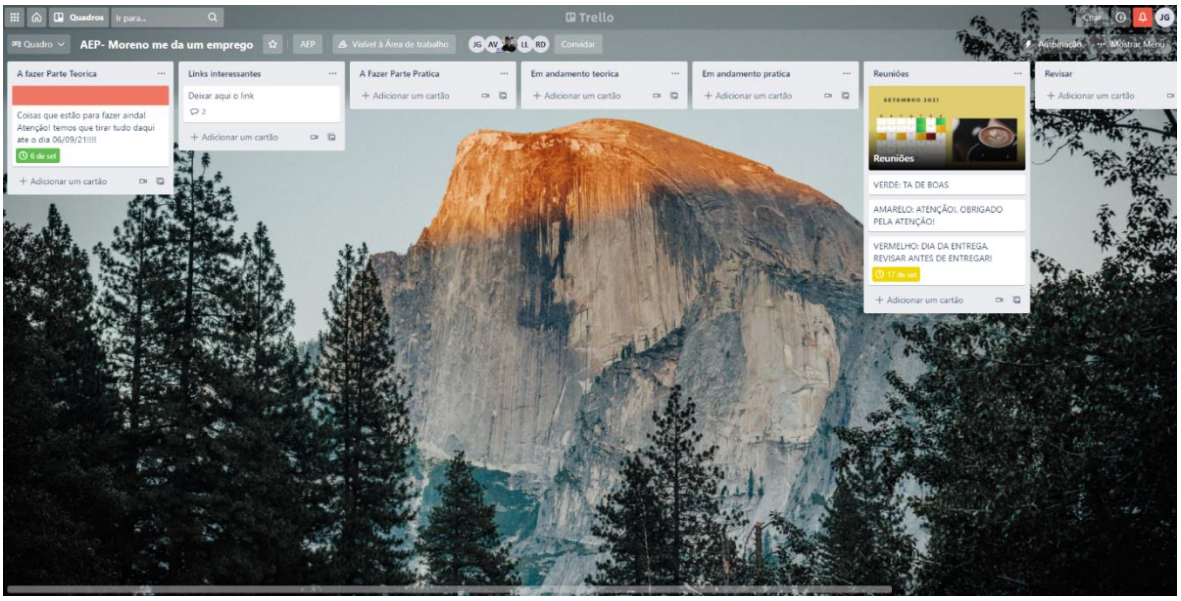
Tendo como resultado o seguinte retorno:

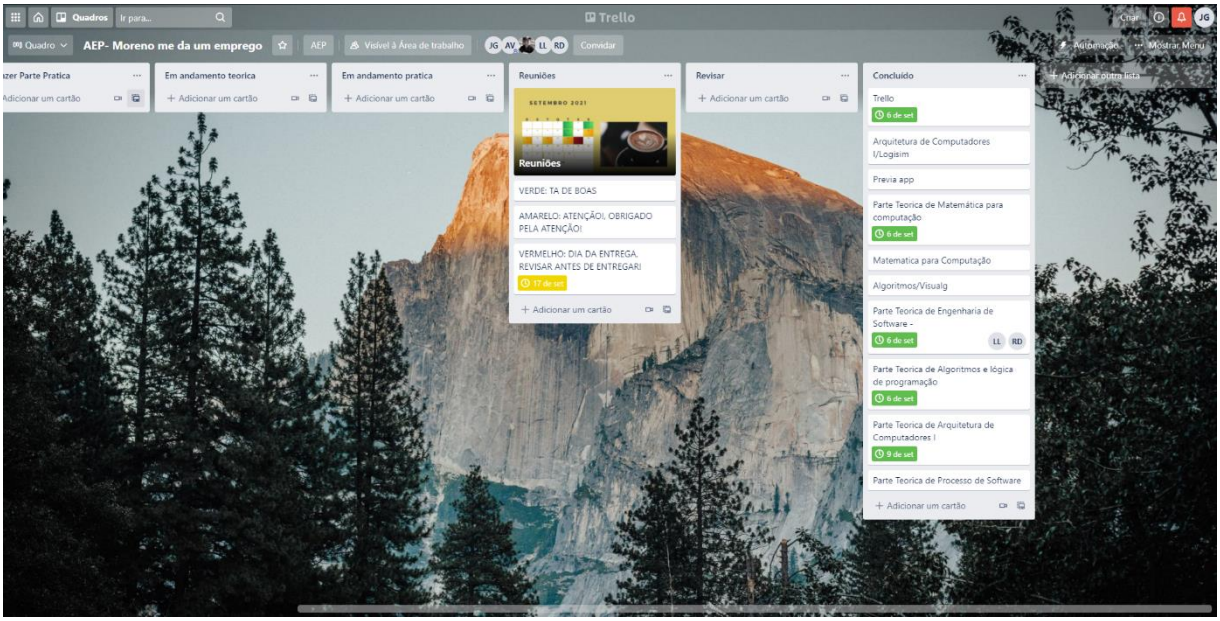



## 3) PROCESSOS DE SOFTWARE

Como dito anteriormente para a resolução desse projeto, escolhemos o modelo em cascata como processo de software, não é um projeto grande e não será feito um sistema completo, então podemos ficar em uma etapa durante um tempo grande sem causar problemas. E para realizar a divisão de tarefas, foi escolhido o Trello, que funciona como um organizador de projetos e é inspirado no Kanban, ele consegue facilitar muito a organização de tarefas dividindo entre os colaboradores do projeto.

As atividades realizadas no Trello são divididas em formas de “post-it”, facilitando e organizando o fluxo de trabalho, cada “post-it” é possível criar descrições, checklists, links para o Google Drive, calendário e muitos outros benefícios.




## 4) MATEMÁTICA PARA COMPUTAÇÃO

Na solução técnica de matemática, usamos a seguinte frase para montar uma tabela da verdade que representasse a solução que o nosso software propõe.

*“Temperatura do ambiente interno está ideal, então os frangos crescem saudáveis, então os frangos vivem”*

[T→(S→V)] →[ (T^S)→V]

| S | T | V | S→V | T→(S→V) | T^S | (T^S)→V | [T→(S→V)] → (T^S)→V |
|---|---|---|---|---|---|---|---|
| v | v | v | V | V | V | V | V |
| v | v | f | F | F | V | F | V |
| v | f | v | V | V | F | V | V |
| v | f | f | F | V | F | V | V |
| f | v | v | V | V | F | V | V |
| f | v | f | V | V | F | V | V |
| f | f | v | V | V | F | V | V |
| f | f | f | V | V | F | V | V |

T = Temperatura do ambiente está ideal
S = Os frangos crescem saudáveis
V = Os frangos vivem

Resultando assim em uma tautologia, que é quando uma dada proposição é sempre verdadeira, sem exceções, é a expressão onde na tabela-verdade a última coluna sempre resultará em verdadeiro.

## 5) ENGENHARIA DE SOFTWARE

Com base em pesquisas, achamos um aparelho já existente, que possui um software que oferece um serviço semelhante ao que oferecemos no nosso produto. O AC Touch, é um dispositivo que auxilia no controle de temperatura de um aviário, podendo ser definido pelo usuário a temperatura máxima e mínima, função essa semelhante à do nosso software. Ele também exibe ao usuário, após 30 dias de uso, um gráfico sobre o andamento do aviário, facilitando a visualização de um progresso ou regresso.

Com base nisso, nos reunimos com nosso cliente para fazermos uma entrevista para que possamos entender os desejos do cliente e realizar o levantamento de requisitos.

Durante a nossa conversa, o cliente nos disse várias coisas que queria que tivesse em sua granja, tais como, quando a umidade baixasse ou aumentasse os exaustores e os nebulizadores ligassem, compreendemos que é preciso de alguns sensores como, de umidade, temperatura e de luminosidade, bem como monitores para ficar observando cada sensor em funcionamento.

Esses sensores serão usados para estabelecer horários para quando a temperatura subisse os sensores ligassem os resfriadores e quando a temperatura caísse muito ligassem os aquecedores. E os sensores de umidade avisará o sistema para que ele acione os exaustores somente quando a umidade estiver alta e os nebulizadores para aumentar a umidade.

Os sensores de luminosidade acionarão as luzes da granja inteira em seus respectivos horários pré-estabelecidos pelo sistema, para que tudo fique automatizado.

### 5.1) Elicitação de requisitos

Precisamos construir um software para a gestão de aviário que resolva os problemas de climatização e controle do ambiente das aves para empresa GALLANIS. Após a entrevista constatamos que o software irá demonstrar ao proprietário e aos funcionários se o aviário está com as condições certas para ter um melhor desempenho na criação de suas aves.

**O sistema deve:**

- Alertar por meio de aplicativo mobile ou PC as alterações de iluminação, umidade e temperatura;
- Alertar o não funcionamento de equipamentos;
- Informar a temperatura, umidade e luminosidade no local;
- Informar como deve estar a climatização no local;
- Suportar um ou mais aviários;
- Poder inserir mais de um lote por aviário;
- Deve ter a inserção de dados referentes as preferencias da climatização;
- Inserção de fornecedores e mantimentos;
- Lançamento e retirada de mantimentos;

**Mobile/PC:**

- O aplicativo mobile apenas monitora e alerta;
- Cadastros de aviários, fornecedores e mantimentos pelo PC;

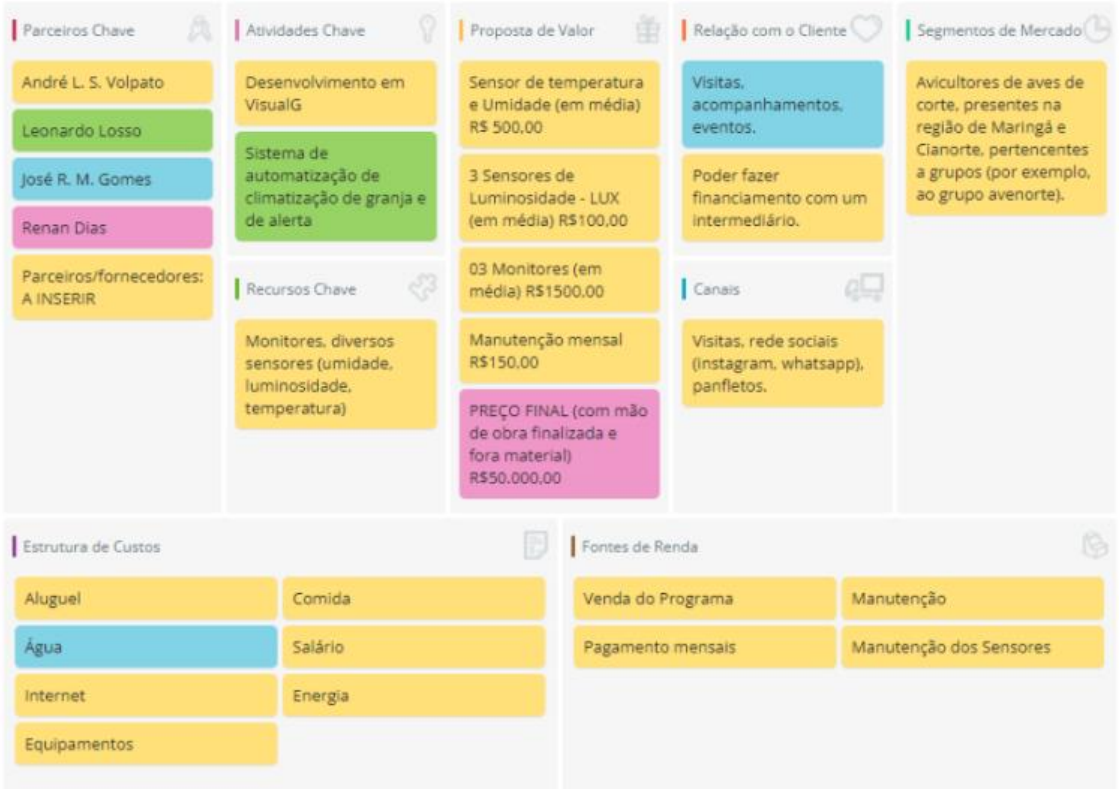
- O PC deve poder inserir dados de controle do ambiente, lançamentos e retiradas de mantimentos;

**Observações:**

- Deve haver menu e submenus;
- O controle das informações deve ser de forma vetorial;
- No aviário deverá ter um monitor e luzes de alerta sem efeito sonoro;
- No aplicativo mobile e na casa do “caseiro” deverá ter um alerta sonoro e visual;

### 5.2) Sebrae Canvas

Aplicamos também no trabalho a metodologia Lean Startup, onde pode ser traduzida como ‘’enxuta’’, onde envolve um trabalho de identificação e eliminação de desperdícios nos processos. Com isso usamos o Sebrae Canvas para nos ajudar a ter uma ideia do que precisamos para desenvolver a nosso software de monitoramento.



## 6) PESQUISA

Para a presente pesquisa começarei falando sobre o método de Processo de Software que escolhemos. Após analisarmos todos os métodos disponíveis e após uma análise sobre as vantagens e desvantagens, escolhemos o método de desenvolvimento de software em cascata. Por se tratar de um projeto pequeno, sem muitas funcionalidades, poderemos ficar em uma etapa por um longe período de tempo sem que afete o andamento do processo de desenvolvimento do software.

Dividimos o projeto em três etapas (planejamento, desenvolvimento e conclusão). Durante a primeira etapa fizemos reuniões para poder elaborar um mockup do projeto para nos servir de esboço para o projeto final, delimitando quais funcionalidades o aplicativo teria, o que ele poderia ou não poderia controlar.

Após concluirmos a primeira etapa, fomos para o desenvolvimento do projeto, usando o mockup desenvolvido como auxilio para o projeto final, criamos um protótipo do produto que terminamos de desenvolver durante esse bimestre em questão.

Partindo do fim do desenvolvimento do aplicativo, voltamos desde o início para polir o que construímos até agora, arrumar erros se for preciso, melhorar o aplicativo com um todo, esse é o intuito dessa fase final.

Em arquitetura de computadores, fizemos a análise combinacional de uma funcionalidade presente em nosso sistema e apresentamos a sua simplificação usando o Mapa de Karnaugh. Onde o Mapa de Karnaugh serve para montar a expressão para fazer um circuito lógico. As funcionalidades do sistema que foram simplificadas para o Mapa foram os controles de luminosidade, temperatura e umidade, que julgamos serem as mais importantes.

| TEMPERATURA | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Aquecedor | | | | Resfriador | | | | Erro | | | |
| | B~ | B | | | B~ | B | | | B~ | B | |
| A~ | 0 | 1 | | A~ | 0 | 0 | | A~ | 0 | 0 | |
| A | 0 | 0 | | A | 1 | 0 | | A | 1 | 1 | |
| R: A~B | | | | R:AB~ | | | | R: A | | | |

## LUZ

| 100% | | | | |
|---|---|---|---|---|
| | B~ | | B | |
| A~ | 0 | 0 | 0 | 0 |
| A | 1 | 0 | 0 | 0 |
| | C~ | C | | C~ |
| R:AB~C~ | | | | |

| 100% 24h | | | | |
|---|---|---|---|---|
| | B~ | | B | |
| A~ | 0 | 1 | 0 | 0 |
| A | 0 | 0 | 0 | 0 |
| | C~ | C | | C~ |
| R:A~B~C | | | | |

| 15% | | | | |
|---|---|---|---|---|
| | B~ | | B | |
| A~ | 0 | 0 | 0 | 1 |
| A | 0 | 0 | 0 | 0 |
| | C~ | C | | C~ |
| R:A~BC~ | | | | |

| Erro | | | | |
|---|---|---|---|---|
| | B~ | | B | |
| A~ | 1 | 0 | 1 | 0 |
| A | 0 | 1 | 1 | 1 |
| | C~ | C | | C~ |
| R:AC;AB; BC; A~B~C~ | | | | |

## UMIDADE

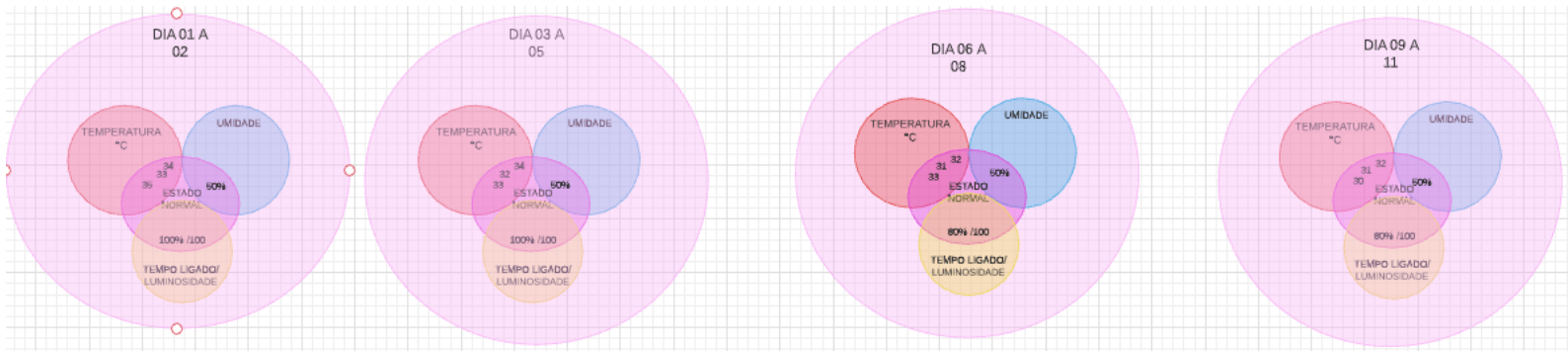
| Exaustores | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 0 |
| A | 1 | 0 |
| R: AB~ | | |

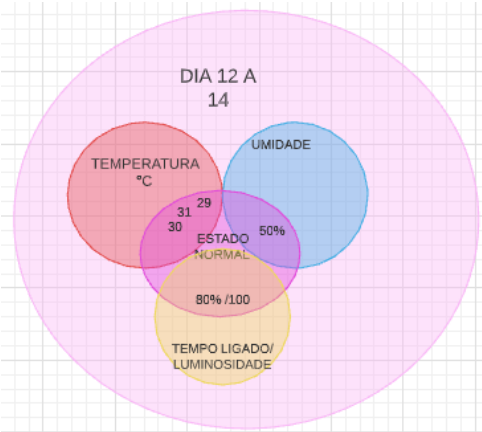
| Nebulizadores | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 1 |
| A | 0 | 0 |
| R: A~B | | |

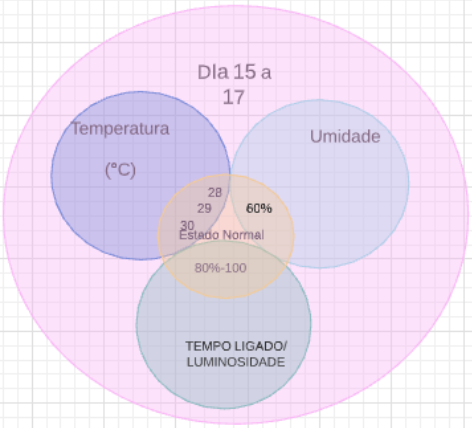
| Erro | | |
|---|---|---|
| | B~ | B |
| A~ | 0 | 0 |
| A | 0 | 1 |
| R: AB | | |

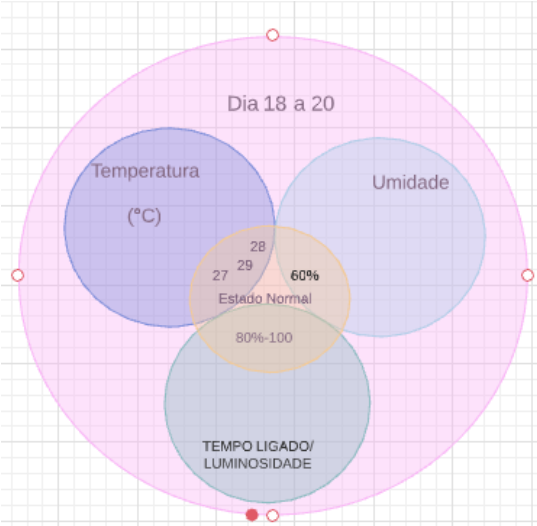
Na parte de Matemática para Computação, fizemos um diagrama de Venn, com os principais aspectos que ajudarão a climatização da granja e da luminosidade, que são eles: Luminosidade, temperatura, umidade. Esses três são os pilares do projeto para que não aconteça nada de errado com as aves é preciso que tudo fique balanceado. O estado normal é a interseção dos valores permitidos para a granja fluir bem, para que o ambiente esteja equilibrado, para que as aves cresçam saudáveis.

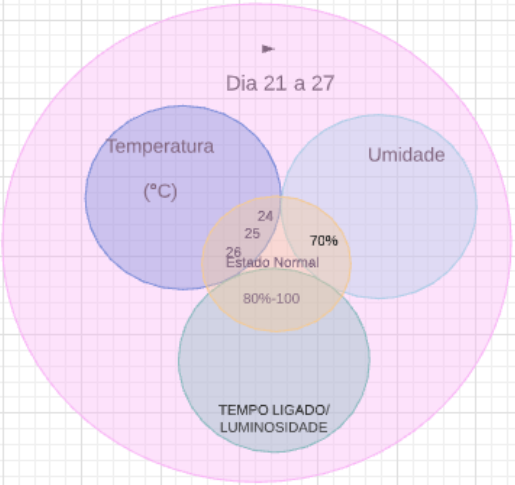
Se os valores não forem iguais ao de dentro do estado normal, o software emitirá um alerta e logo em seguida deverá ligar os equipamentos para corrigir tal alteração para que volte ao estado normal.

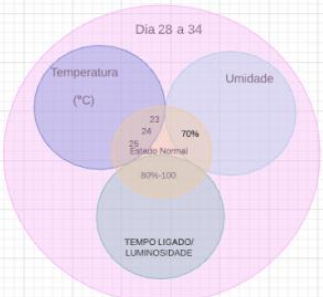


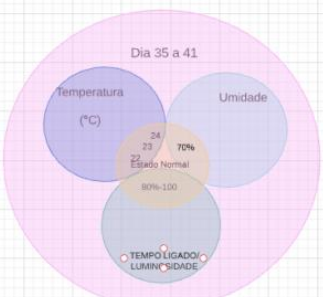

DIA 12 A
14
TEMPERATURA
°C
UMIDADE
31
29
30
ESTADO
NORMAL
50%
80% /100
TEMPO LIGADO/
LUMINOSIDADE

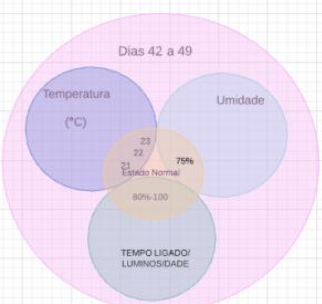

DIa 15 a
17
Temperatura
(°C)
Umidade
28
29
60%
30
Estado Normal
80%-100
TEMPO LIGADO/
LUMINOSIDADE

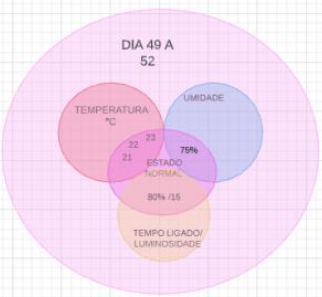

Dia 18 a 20
Temperatura
(°C)
Umidade
28
29
27
60%
Estado Normal
80%-100
TEMPO LIGADO/
LUMINOSIDADE


Dia 21 a 27
Temperatura
(°C)
Umidade
24
25
70%
26
Estado Normal
80%-100
TEMPO LIGADO/
LUMINOSIDADE


Dia 28 a 34
Temperatura
(°C)
Umidade
23
24
70%
25
Estado Normal
80%-100
TEMPO LIGADO/
LUMINOSIDADE


Dia 35 a 41
Temperatura
(°C)
Umidade
24
23
70%
22
Estado Normal
80%-100
TEMPO LIGADO/
LUMINOSIDADE


Dias 42 a 49
Temperatura
(°C)
Umidade
23
22
75%
21
Estado Normal
80%-100
TEMPO LIGADO/
LUMINOSIDADE


DIA 49 A
52
TEMPERATURA
°C
UMIDADE
22
23
75%
21
ESTADO
NORMAL
80% /15
TEMPO LIGADO/
LUMINOSIDADE

## 7) Links Úteis

https://www.canva.com/design/DAEqG9td0Qk/0jD1jNhX4R4XOx6HI85k0w/view?utm_content=DAEqG9td0Qk&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton. – Protótipo para dispositivos móveis.

https://www.canva.com/design/DAEvw57_Jvk/3IsEzX0X7xqz5bkcbsmD3w/view?utm_content=DAEvw57_Jvk&utm_campaign=designshare&utm_medium=link&utm_source=publishsharelink#1. – Protótipo para o monitoramento.

https://drive.google.com/file/d/1HqLgMukuQBED3PurbBjSZf83ijQfY4Z-/view Algoritmo do Software feito no VisualG versão 3.0.7.

https://docs.google.com/presentation/d/19HftarFlqBaQemyjDTjGPGQacTbrXp0D/edit?usp=sharing&ouid=102245950094287672426&rtpof=true&sd=true – Slides Apresentação.

https://youtu.be/FJ_QyZOBdkE - Apresentação da AEP

## 8) Conclusão

Com o término desse trabalho concluímos que o nosso software é benéfico para o produtor, ajudando a manter a qualidade do seu produto, criando as condições ambientais perfeitas para o desenvolvimento dos frangos (como os alertas sonoros e visuais que ocorrem sem incomodar os frangos), nosso software conta também com algo a mais, um controle dos mantimentos da granja, sejam eles mantimentos para os frangos ou peças para o maquinário do local. Gallanis é um software muito versátil, seu código foi escrito pensando em um sistema orgânico, de fácil mudança para incrementar as novas demandas dos produtores.

## REFERÊNCIAS:


PRESSMAN, Roger S. Engenharia de Software: Uma abordagem profissional. 2016. 8° Edição. Acesso em 16 set. 2021.

SOMMERVILLE, Ian. Engenharia de Software. 2011. 9° Edição. Acesso em 05 nov. 2021.

https://acadtec.com.br/blog/desenvolvimento-backend/qual-a-importancia-de-algoritmos-na-programacao. Acesso em: 16 set. 2021

https://kenzie.com.br/blog/logica-de-programacao/. Acesso em: 16 set. 2021

https://www.devmedia.com.br/logica-uma-ferramenta-indispensavel-na-programacao-de-computadores/28386. Acesso em: 16 set. 2021

http://www.skillfulreasoning.com/propositional_logic/well-formed_formulas.html. Acesso em: 16 set. 2021

https://dl.acm.org/doi/10.5555/1074100.1074917. Acesso em: 16 set. 2021

https://www.portaleducacao.com.br/conteudo/artigos/biologia/modelos-de-processo-de-softwares/53061#. Acesso em: 16 set. 2021

https://edisciplinas.usp.br/pluginfile.php/839466/mod_resource/content/1/Aula02_ModelosProcessos.pdf. Acesso em: 16 set. 2021

https://medium.com/contexto-delimitado/o-modelo-incremental-b41fc06cac04. Acesso em: 16 set. 2021

https://medium.com/contexto-delimitado/o-modelo-em-espiral-de-boehm-ed1d85b7df. Acesso em: 16 set. 2021

https://medium.com/@mikiasoliveira/re%C3%BAso-de-software-8406766d9eb8. Acesso em: 16 set. 2021

https://www.crmvpb.org.br/no-dia-da-avicultura-cfmv-destaca-a-importancia-da-atividade/. Acesso em: 16 set. 2021

https://diarural.com.br/pais-e-o-maior-exportador-de-carne-bovina-e-o-4o-maior-produtor-de-graos-do-globo/. Acesso em: 16 set. 2021

https://campovivo.com.br/avicultura/cna-em-2020-pib-do-frango-aumentou-quase-10-e-o-do-ovo-mais-de-18/. Acesso em: 16 set. 2021

https://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/eletronica_basica_capitulo_09_karnaug_tocci_2014.pdf. Acesso em: 16 set. 2021

https://dicasdeprogramacao.com.br/o-que-sao-funcoes-e-procedimentos/. Acesso em: 16 set. 2021

https://endeavor.org.br/estrategia-e-gestao/lean-startup/. Acesso em: 16 set. 2021.