BANDTEC – DIGITAL SCHOOL

CURSO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

NOME DOS AUTORES  
aNTonio augusto fonseca monteiro  
brunno costa castigrini  
gustavo quaresma da costa  
luiz henrique oliveira nardi  
vitória da silva eleutério pinto

tECCHICKEN

SÃO PAULO

2021

Sumário

1 VISÃO DO PROJETO 5

1.1 **APRESENTAÇÃO DO GRUPO** 5

1.2 **CONTEXTO** 5

1.3 **Problema / justificativa do projeto** 6

1.4 **objetivo da solução** 6

1.5 **diagrama da solução**

2 PLANEJAMENTO DO PROJETO 9

2.1 **Definição da Equipe do projeto** 9

2.2 **PROCESSO E FERRAMENTA DE GESTÃO DE PROJETOS** 9

2.3 **Gestão dos Riscos do Projeto** 11

2.4 **PRODUCT BACKLOG e requisitos** 12

2.5 **Sprints / sprint backlog** 13

3 desenvolvimento do projeto 16

3.1 **Solução Técnica – Aquisição de dados Arduino/SIMULADOR** 16

3.2 **Solução Técnica - Aplicação** 16

3.3 **Banco de Dados** 16

3.4 **Protótipo das telas, lógica e usabilidade** 16

3.5 **MÉTRICAS** 16

4 implantação do projeto 18

4.1 **Manual de Instalação da solução** 18

4.2 **Processo de Atendimento e Suporte / FERRAMENTA** 18

5 CONCLUSÕES 20

5.1 **resultados** 20

5.2 **Processo de aprendizado com o projeto** 20

5.3 **Considerações finais sobre A evolução da solução** 20

ReferÊncias 21

VISÃO DO PROJETO

# VISÃO DO PROJETO

## **APRESENTAÇÃO DO GRUPO**

* **Nome do grupo:** TecChicken.
* **Integrantes:** Antonio Augusto Fonseca Monteiro, Brunno Costa Castigrini, Gustavo Quaresma da Costa, Luiz Henrique Oliveira Nardi e Vitória da Silva Eleutério Pinto.
* **Logomarca:**
* **Posicionamento no mercado/acadêmico:** Ramo aviário com foco no monitoramento de temperatura de granjas.

## **CONTEXTO**

Mercado direcionado as granjas das raças: ISA Brown e Bovans White. Aproximadamente 20.000 granjas em São Paulo que podem adquirir o nosso serviço.

Em relação à sustentabilidade, o projeto pode auxiliar na prevenção de incêndios nas granjas, assim como a preservação da vida das aves, evitando desperdícios de galinhas, reduzindo possíveis custos de reestruturação das granjas, causados por problemas que poderemos identificar.

Em relação aos custos da empresa, os sensores serão adquiridos sob demanda e serão repassados ao cliente, os custos variáveis se darão em torno da equipe (TI, atendimento, marketing, suporte), que deverá aumentar conforme a quantidade de clientes.

## **Problema / justificativa do projeto**

O Brasil é uma grande potência no ramo da agroindústria, tendo 6% do PIB (Produto Interno Bruto) brasileiro e sendo comercializado com o mundo inteiro. As criações de vacas, porcos e galinhas são as mais destacadas.

Entre os países que mais produzem e exportam produtos derivados das galinhas, o Brasil foi o 3° que mais produziu carne de frango, atrás de Estados Unidos e China, e o 1° que mais exportou no ano de 2019, segundo a USDA (United States Departament of Agriculture).

No lucrativo ramo de produção de ovos existem diversos processos necessários para garantir o bem-estar da galinha, e consequentemente, garantir a qualidade dos ovos que ela irá botar. Um desses itens importantes para garantir a qualidade do produto gerado é o controle de temperatura.

Além do controle padrão e essencial, há o perigo de morte das aves por falta de controle de temperatura. Um exemplo é o recente caso da morte de 50mil galinhas na cidade de Bastos, onde a temperatura chegou à marca de 41°C e gerou um prejuízo de 3 milhões de reais para o produtor.

Portanto, o objetivo do projeto envolve identificar e avisar ao cliente, os momentos em que a termperatura está passando por picos, tanto abaixo quanto acima da temperatura ideal, para que ele possa tomar uma atitude para resolver esse problema.

## **objetivo da solução**

A solução envolve a instalação de sensores, conectados a um arduíno para coletar constantemente as temperaturas da granja, para que possamos fazer o tratamento dos dados e mostrá-los ao cliente de uma forma didática e intuitiva, envolvendo as temperaturas em tempo real, assim como as médias, separadas por sensor.

## **diagrama da solução**

Diagrama

Descrição gerada automaticamente

PLANEJAMENTO DO PROJETO

# PLANEJAMENTO DO PROJETO

## **Definição da Equipe do projeto**

Antonio fez parte do time de desenvolvimento back-end, tomando conta do desenvolvimento da API

Brunno foi o Product Owner e Scrum Master do projeto, cuidando da organização do projeto, definindo os rumos.

Gustavo fez parte do time de desenvolvimento back-end, tomando conta do desenvolvimento da API.

Luiz fez parte da equipe do time de TI, auxiliando na execução da metodologia ágil, bem como gerindo a documentação e o Help Desk.

Vitória fez parte do time de desenvolvimento front-end, tomando conta dos elementos visuais do projeto.

## **PROCESSO E FERRAMENTA DE GESTÃO DE PROJETOS**

A metodologia ágil adotada foi auxiliada pela plataforma Trello, a qual nos auxiliava a enxergar as tarefas pendentes, bem como revisar os assuntos discutidos em reuniões, todos pautados nas atas de reunião.

No início do projeto, as reuniões eram realizadas diariamente, para conseguirmos nos habituar com o projeto e trazer algum resultado rápido, no entanto, durante a Sprint 3 do projeto, as reuniões se tornaram menos recorrentes, duas vezes por semana apenas, aproveitando esse tempo no desenvolvimento do projeto.

As tarefas eram definidas buscando o MVP, ou seja, apresentar um resultado rápido, para mostrar constante evolução no projeto, o que fazia com que as últimas semanas da entrega aparentassem ter pouco progresso, pois tudo já estava quase finalizado, cuidando apenas de elementos como apresentação e documentação.

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Ata de Reunião:

Data: 19/03/2021

Participantes presentes: Todos

Participantes ausentes: -

Assuntos discutidos e principais decisões:

- Definição das regras de condução

- Definição do modelo da ata

- Configuração da ferramenta de gestão (Trello)

- Definição da pauta para próxima reunião

- Definir as páginas do site

- Definição do banco de dados

Plano de ação:

- Definir os pontos de evolução individual dos membros do grupo

- Começar a prototipação do site

- Remodelar as tabelas do banco de dados

## **Gestão dos Riscos do Projeto**

* Distrações:
  + Probablidade: muito provável
  + Impacto: alto
  + Fator de risco: 9
  + Ação: eliminar
  + Plano de ação: reconhecer os momentos ideais para conversar e trabalhar, evitando reuniões longas.
* Desafio técnico:
  + Probablidade: provável
  + Impacto: médio
  + Fator de risco: 4
  + Ação: mitigar
  + Plano de ação: Identificar as dificuldades e sanar as dúvidas com alguém preparado.
* Faltas em reuniões ou dias de trabalho:
  + Probabilidade: pouco provável
  + Impacto: baixo
  + Fator de risco: 1
  + Ação: mitigar
  + Plano de ação: as faltas ocasionais não devem afetar o rumo do projeto, desde que o projeto esteja dentro do prazo.
* Falta de parametrização na documentação:
  + Probabilidade: provável
  + Impacto: alto
  + Fator de risco: 6
  + Ação: eliminar
  + Plano de ação: documentar as decisões tomadas nas reuniões e parametrizar as métricas adotadas.
* Saída de um integrante:
  + Probabilidade: pouco provável
  + Impacto: alto
  + Fator de risco: 3
  + Ação: mitigar
  + Plano de ação: manter o projeto adiantado, para que a saída do membro afete o mínimo do projeto.

## **PRODUCT BACKLOG e requisitos**

Product Backlog

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requisitos** | **Tamanho** | **Ordem de Execução** |
| 1 | Site | 21 | 4 |
| 2 | Banco de dados | 8 | 5 |
| 3 | Documentação | 13 | 1 |
| 4 | Desenvolvimento da API | 8 | 6 |
| 5 | Nuvem | 2 | 7 |
| 7 | Arduíno/Sensor (simulador) | 3 | 3 |
| 8 | GitHub | 1 | 2 |

Requisitos

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requisitos** | **Classificação** | **Tamanho** |
| 1 | Site institucional | Essencial | 21 |
| 2 | Sensores funcionando | Essencial | 5 |
| 3 | Banco de dados em nuvem | Essencial | 8 |
| 4 | API conectando o site com as informações em nuvem | Essencial | 13 |
| 5 | Indicadores (gráficos, média dos sensores) | Essencial | 13 |
| 6 | Criptografia do banco de dados | Desejável | 8 |
| 7 | Help Desk | Essencial | 5 |
| 8 | Página adaptável conforme a quantidade de sensores | Importante | 21 |

## **Sprints / sprint backlog**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Requisitos** | **Tamanho** | **Ordem de Execução** | **Sprint** |
| 1 | Projeto criado e configurado no GitHub | 1 | 3 | 1 |
| 2 | Documento de contexto de negócio | 8 | 2 | 1 |
| 3 | Documento de justificativa do projeto | 8 | 1 | 1 |
| 4 | Desenho de solução | 5 | 4 | 1 |
| 5 | Protótipo do site institucional | 13 | 9 | 1 |
| 6 | Tela de simulador financeiro | 21 | 10 | 1 |
| 7 | Planilha de requisitos / backlog do projeto | 8 | 5 | 1 |
| 8 | Tabelas criadas no MySQL | 5 | 6 | 1 |
| 9 | Execução de script de inserção de registros | 2 | 7 | 1 |
| 10 | Execução de script de consulta de dados | 2 | 8 | 1 |
| 11 | Preparar arduíno/simulador | 13 | 12 | 1 |
| 12 | Escolher e testar o sensor no simulador | 5 | 11 | 1 |
|  |  |  |  |  |
| 13 | Planilha de riscos do projeto | 3 | 2 | 2 |
| 14 | Site estático institucional | 21 | 7 | 2 |
| 15 | Site estático dashboard | 21 | 9 | 2 |
| 16 | Site estático cadastro e login | 13 | 8 | 2 |
| 17 | Ferramenta de gestão de projeto funcionando | 3 | 1 | 2 |
| 18 | HLD do projeto / LLD do projeto | 13 | 4 | 2 |
| 19 | Planilha de Backlog / Planilha de Sprints | 5 | 3 | 2 |
| 20 | Modelagem lógica do projeto | 8 | 5 | 2 |
| 21 | Script de criação do banco | 5 | 6 | 2 |
| 22 | Teste com sensor / Gráficos | 13 | 11 | 2 |
| 23 | Especificação do Analytics / Métricas | 8 | 10 | 2 |
|  |  |  |  |  |
| 24 | Manual de instalação | 3 | 12 | 3 |
| 25 | Documento final do projeto | 13 | 13 | 3 |
| 26 | PPT da apresentação do projeto | 8 | 14 | 3 |
| 27 | Prévia (demonstração da solução + apresentação) | 8 | 11 | 3 |
| 28 | Site institucional na Nuvem | 21 | 8 | 3 |
| 29 | Cadastro, login e dashboard na Nuvem, conectado com BD | 21 | 6 | 3 |
| 30 | Mapeamento das tabelas em classes Javascript | 8 | 7 | 3 |
| 31 | Fluxograma do processo de atendimento do suporte | 13 | 4 | 3 |
| 32 | Ferramenta de Help Desk configurada e integrada à solução | 5 | 5 | 3 |
| 33 | Modelagem lógica (final) | 8 | 1 | 3 |
| 34 | Script de criação do banco (final) | 3 | 3 | 3 |
| 35 | Tabelas criadas no Azure (final) | 3 | 2 | 3 |
| 36 | Teste integrado do Analytics | 8 | 9 | 3 |
| 37 | Teste integrado da solução de IoT (simulador + BD) | 13 | 10 | 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Requisitos** | **Tamanho** | **Ordem de Execução** | **Sprint** |
| 1 | Site | 21 | 4 | 2, 3 |
| 2 | Banco de dados | 8 | 5 | 2, 3 |
| 3 | Documentação | 13 | 1 | 1, 2, 3 |
| 4 | Desenvolvimento da API | 8 | 6 | 3 |
| 5 | Nuvem | 2 | 7 | 3 |
| 7 | Arduíno/Sensor (simulador) | 3 | 3 | 2, 3 |
| 8 | GitHub | 1 | 2 | 1, 2, 3 |

desenvolvimento do projeto

# desenvolvimento do projeto

## **Solução Técnica – Aquisição de dados Arduino/SIMULADOR**

Ferramentas utilizadas: JavaScript e SQL server.

Nossa solução se baseia na simulação de uma placa Arduino UNO equipada com um sensor LM35, por meio de uma API NodeJS que gera e insere valores randômicos simulando então o sensor.

Nessa primeira imagem é possível vermos onde está sendo definido o mínimo e máximo da simulação dos dados.

Texto

Descrição gerada automaticamente

Já nessa imagem, podemos ver como está funcionando a inserção dos dados no Banco de Dados, basicamente, está sendo inserido na tabela “captura” a temperatura, o momento, e o sensor em que os dados serão armazenados.

Texto

Descrição gerada automaticamente com confiança média

Armazenado dessa forma no banco, é possível realizarmos futuramente um Select para retornar a temperatura para o gráfico, deixando algo totalmente automático mesmo sendo apenas um simulador.

Diagrama de arquitetura

Uma imagem contendo Linha do tempo

Descrição gerada automaticamente

## **Solução Técnica - Aplicação**

Ferramentas utilizadas: JavaScript, HTML, CSS e SQL server.

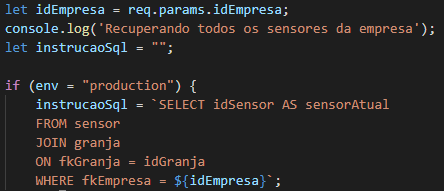
Resumidamente, os dados são recuperados através de “fetch” s no Javascript, que fazem SELECT’S no banco de dados e em seguida são exibidos no documento HTML.  
Detalhadamente, no documento **dashboard.html** (a primeira página com a qual o usuário se depara após efetuar o login), os dados são recuperados através da função “obterSensores”, onde um fetch é realizado com o “idEmpresa” como parâmetro (que identifica á qual empresa pertence aquele usuário) para recuperar todos os sensores relativos á aquela determinada empresa.



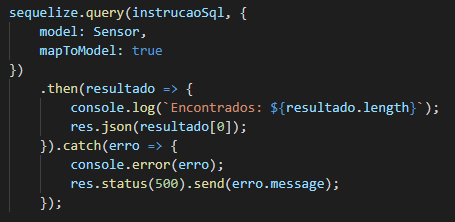
Esse fetch busca no arquivo sensores.js o router.get (‘:/iEmpresa



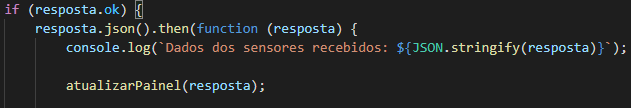
Recupera o “idEmpresa” no “let idEmpresa = req.params.idEmpresa” e faz um SELECT no banco de dados utilizando esse dado como parâmetro para o WHERE:



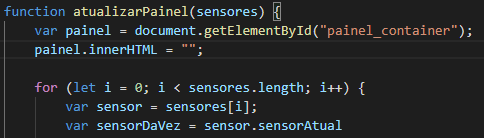
Em seguida, retorna o resultado desse select como o objeto “resultado[0]”



Voltando para o dashboard.html, caso não ocorra nenhum erro, a função “atualizarPainel” é invocada com o objeto “resposta” como parâmetro.



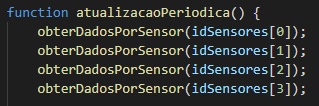
Na função “atualizarPainel” uma div com o id “painel\_container” tem seu conteúdo apagado, e nela são colocados novos paíneis com informações e funções relativas aos sensores daquela empresa.

(“for” que repete de acordo com o tamanho da resposta/select/sensores + As tag’s sendo criadas com sua devidas propriedades e informações inseridas)

  
  
Também dentro dessa função, acontece um push que guarda qual o ID dos sensors relativos a dessa empresa:



Que são posteriormente usados na função “atualizacaoPeriodica”, sendo que esta é invocada no final da função “atualizarPainel”.

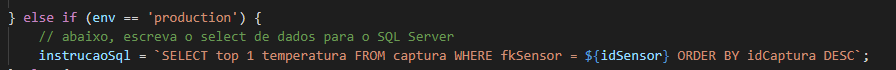
  


A função “obterDadosPorSensor” por sua vez, realiza um fetch no arquivo capturas.js com o idSensor como parâmetro.

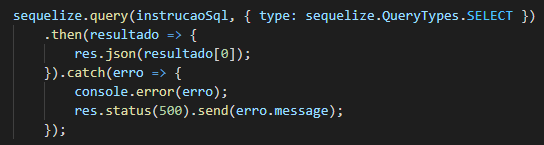




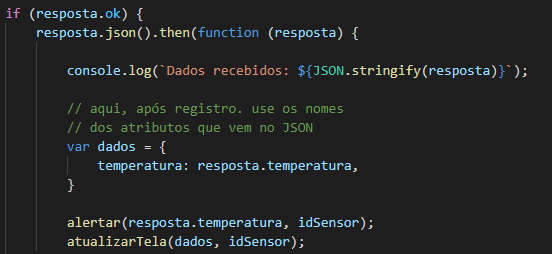
Dentro do endereço /capturas/tempo-real/:idSensor, um select é feito em busca do último registro de temperatura capturado por aquele sensor:



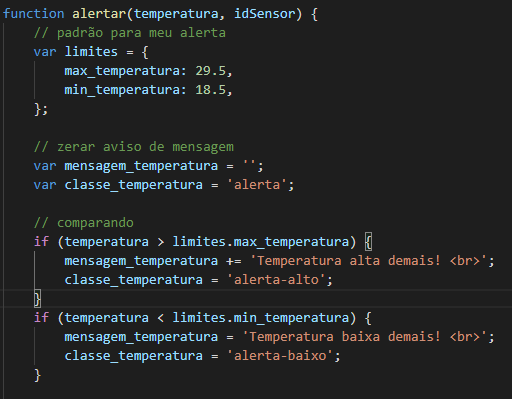
Que é retornado como “resultado[0]” para o dashboard.html



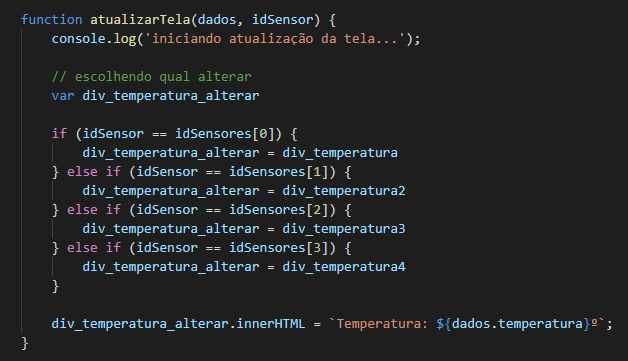
Com tudo dando certo, a “resposta” é guardada dentro da var/objeto “dados”, que em seguida, junto com o “idSensor”, é utilizada como parâmetro para invocar a função “atualizarTela” e “alertar”.



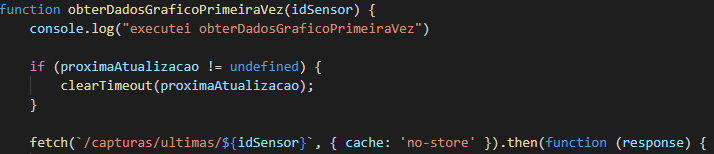
A função “alertar”, muda o alerta colocado no paínel da dashboard de acordo com os limites estabelecidos na var “limites”.

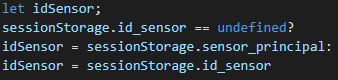


Já a função “atualizarTela”, atualiza no front-end qual foi a última temperatura capturada por aquele sensor, sendo esta mostrada na “div\_temperatura” definida.

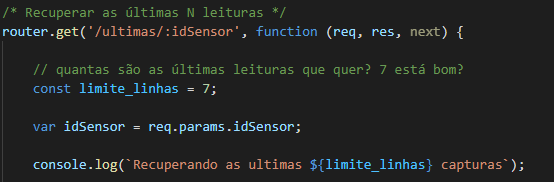


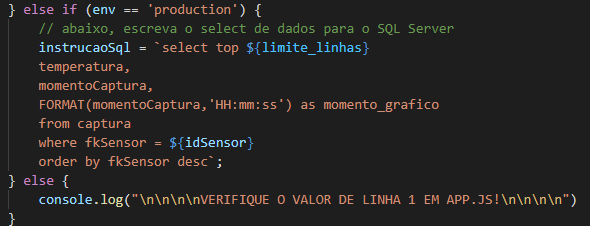
Já no documento **tela-sensor.html -** onde as capturas são mostradas em tempo real - os últimos dados são recuperados, primeiramente, na função “obterDadosGraficosPrimeiraVez”, que utiliza como parâmetro o “idSensor”, sendo este definido de acordo com qual botão o usuário clicou no dashboard.html e que é guardado em um sessionStorage.



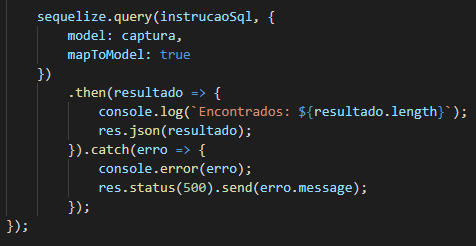


Após invocada, o endereço capturas/ultimas/:idSensor é acessado, onde é feito um SELECT em busca dos 7 primeiros dados do sensor.

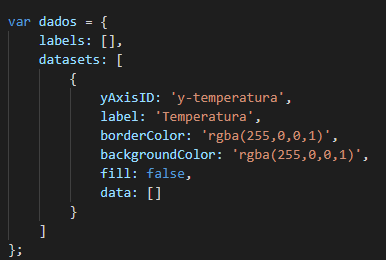




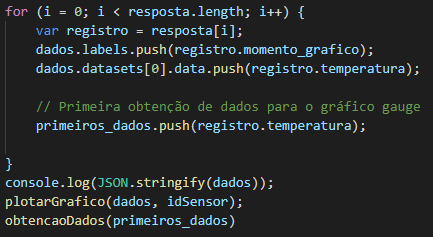
Com o SELECT sendo realizado, o resultado é enviado para o endereço que o buscou como o objeto “resultado”.



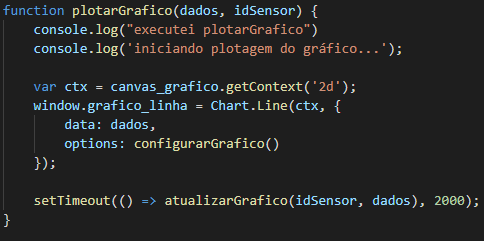
Se não ocorrer nenhum erro, a variável “dados” é criada, inicialmente sem dados ou labels.



Em seguida, um “for” é realizado, adicionando os dados recebidos na variável “dados” e no fim, a função “plotarGraficos” é invocada, com “dados” e “idSensor” como parâmetros.

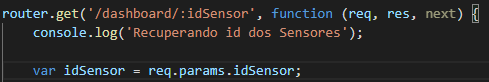


Na função “plotarGrafico”, esses dados capturados são colocados em um gráfico da página anteriormente criado na função “configurarGrafico”, e com o intervalo de 2 segundos, a função “atualizarGrafico) é invocada, com “idSensor” e “dados” como parâmetro.

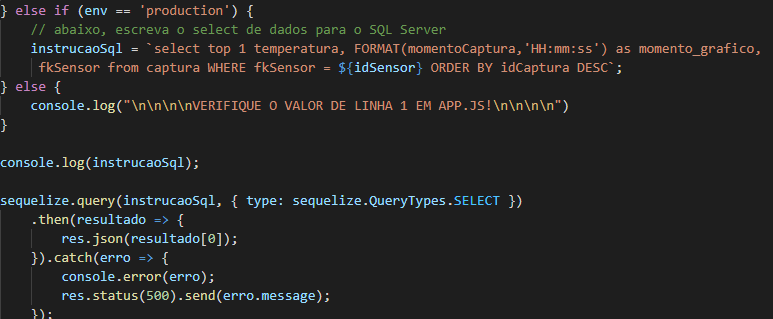


Na função “atualizarGrafico”, um fetch é feito com a intenção de recuperar o último dado capturado por aquele sensor.



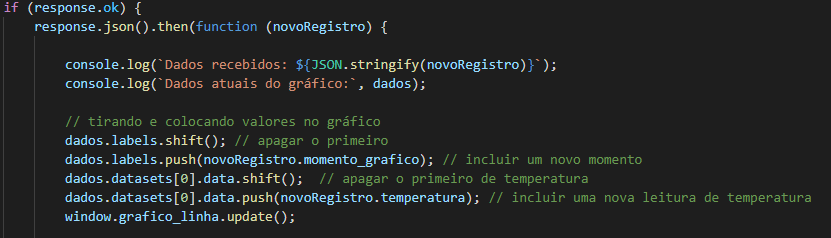


Por fim, o resultado é retornado como “resultado[0]” para o documento onde esse fetch foi solicitado.



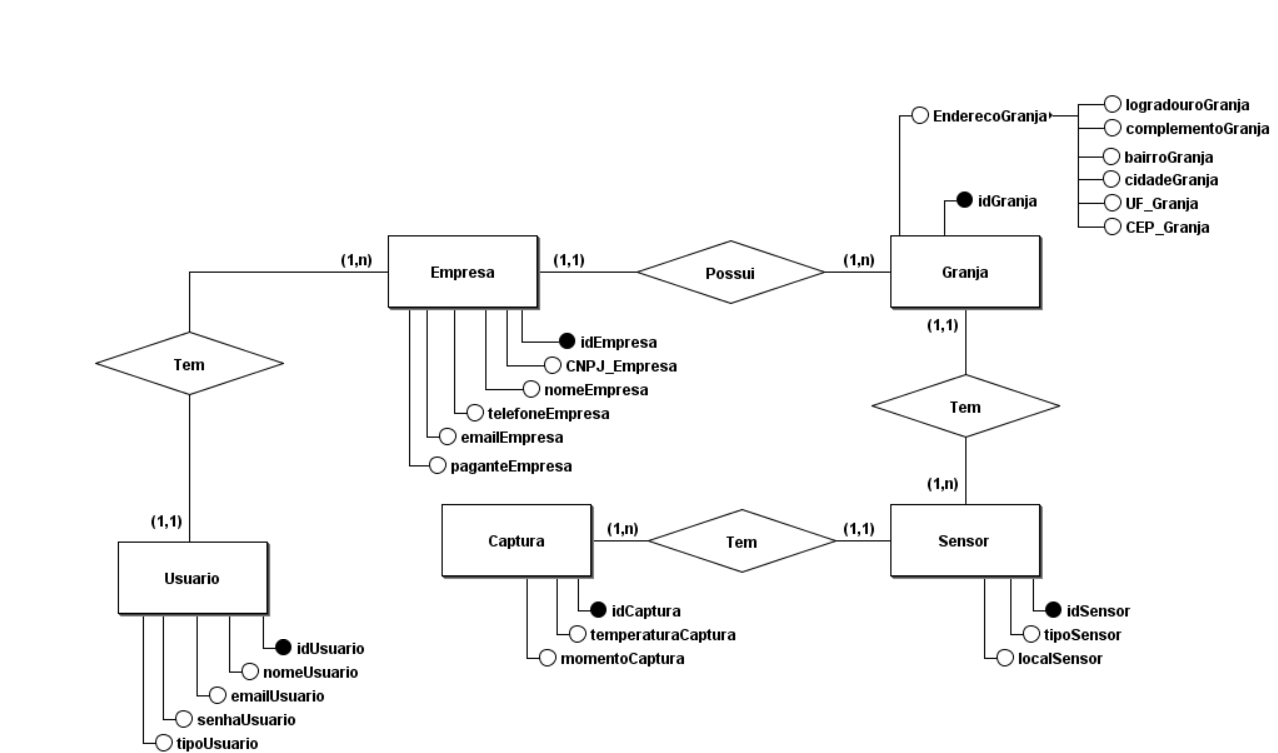
O resultado é utilizado na função “novoRegistro”, que servirá para tirar o último dado do “dados” e adicionar o último resgatado.

O gráfico é atualizado na função update(), sendo mostrado para o usuário no front-end.

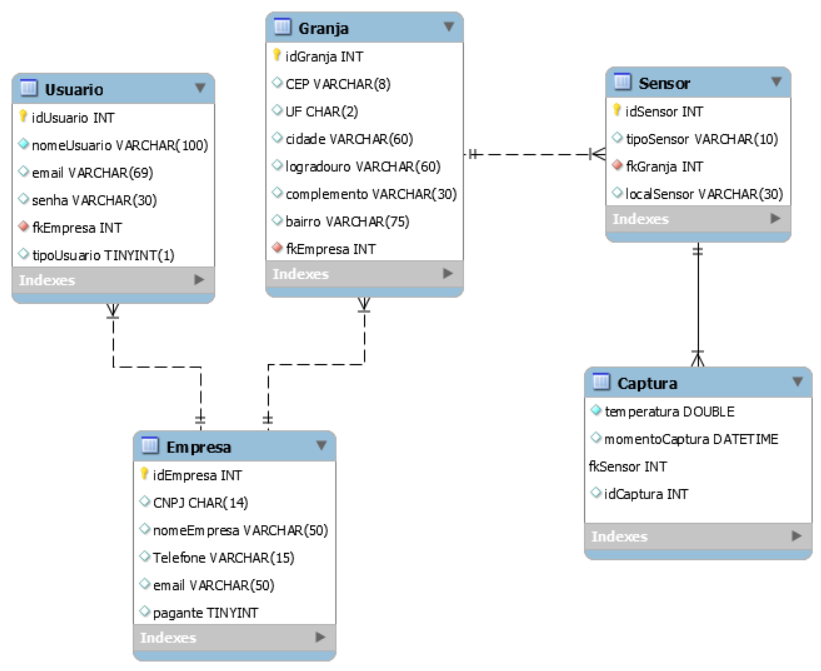


## **Banco de Dados**

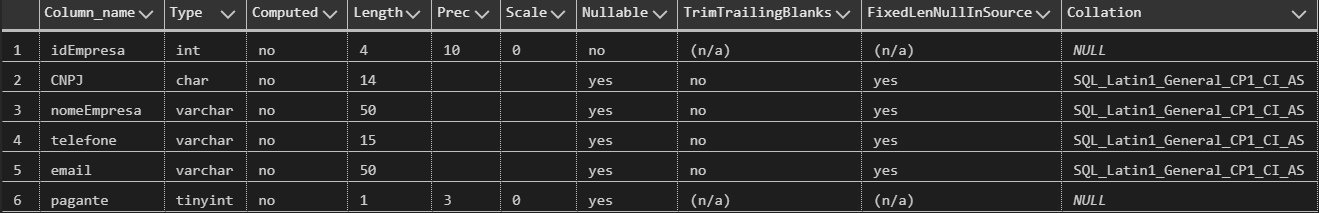
Modelo Conceitual

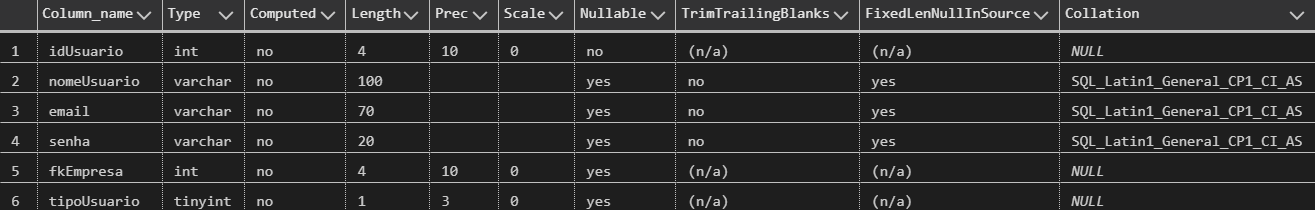


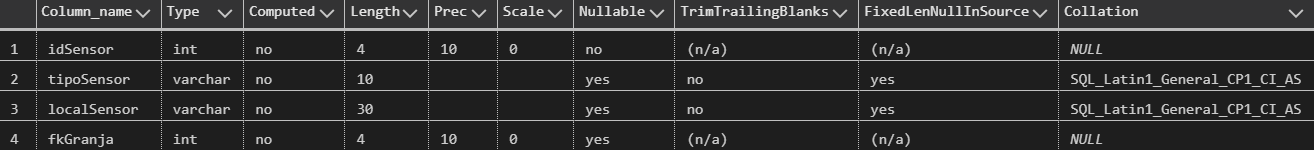
Modelo Lógico

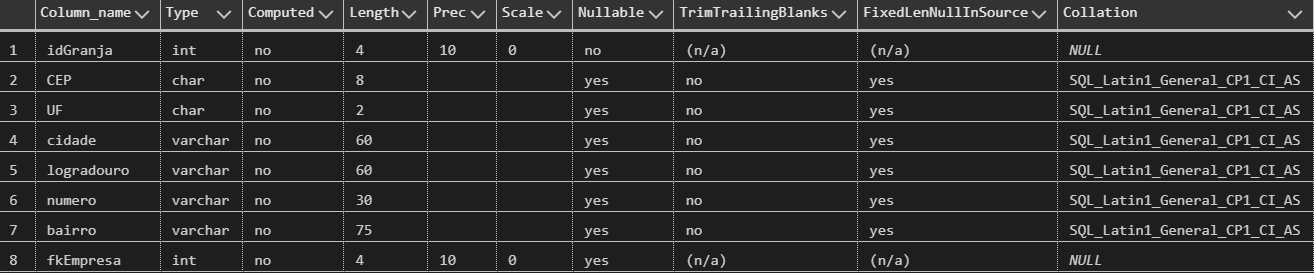


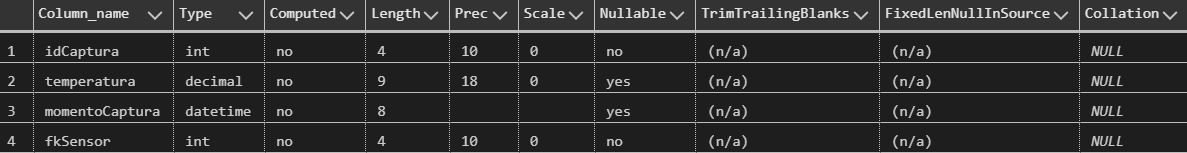
Modelo Físico



****

****

****

****

## **Protótipo das telas, lógica e usabilidade**

Apresentar as telas construídas e sua lógica de navegação

## **MÉTRICAS**

As métricas adotadas seguem os seguintes parâmetros:

Gráfico

Descrição gerada automaticamente

Indicador de temperatura muito baixa, indicando temperaturas menores que

13°C.

Indicador de temperatura baixa, indicando as temperaturas entre 13°C e

20,3°C.

Indicador de temperatura ideal, indicando as temperaturas entre 20,3° e

27,6°C.

Indicador de temperatura alta, indicando as temperaturas entre 27,6°C e

35°C.

Indicador de temperatura muito alta, indicando as temperaturas maiores que

35°C.

Os alertas serão acionados quando a temperatura estiver abaixo de 18,5°C ou acima de 29,5°C, indicando que a temperatura está baixa ou alta, respectivamente.

A métrica foi definida de acordo com o risco de as galinhas sofrerem danos prejudiciais a saúde, onde os valores de temperatura alta e temperatura baixa devem ser motivo de atenção, mas ainda não causaram danos irreversíveis. No entanto, os valores de temperatura muito alta e temperatura muito baixa, indicam um grave problema, que pode ter afetado a saúde das galinhas e qualidade dos ovos.

Portanto, os alertas do site foram definidos de maneira a se evitar o problema e não apenas indicar o problema, por isso, eles são executados pouco antes dos limites críticos para a saúde das galinhas, de maneira que o cliente possa tomar alguma atitude a fim de regular a temperatura.

implantação do projeto

# implantação do projeto

## **Manual de Instalação da solução**

Descritivo básico da instalação da solução e principais cuidados. Guia de instalação e uso.

## **Processo de Atendimento e Suporte / FERRAMENTA**

Desenho e apresentação do Processo de Suporte (diagrama BPM-N);

Apresentação e detalhamento da ferramenta utilizada para Help Desk/Suporte;

Canais de atendimento (telefone,e-meil, chat), níveis de suporte, base de conhecimento na ferramenta selecionada.

# CONCLUSÕES

## **resultados**

Cumprimento dos requisitos, performance, usabilidade.

## **Processo de aprendizado com o projeto**

Detalhamento e visão do grupo em relação ao aprendizado durante o desenvolvimento do projeto.

## **Considerações finais sobre A evolução da solução**

Qual a visão do grupo em relação à evolução deste projeto. Caso haja mais tempo e dedicação no projeto em versões futuras, como ele seria ofertado/apresentado.

ReferÊncias

CRIAR E PLANTAR. Galinha. Estresse calórico.2011. Disponível em: <http://www.criareplantar.com.br/pecuaria/lerTexto.php?categoria=21&id=285>.  
Acesso em: 06/03/2021

MFRURAL. Galinhas e Frangos. Poedeiras 051. 2021. Disponível em:< https://www.mfrural.com.br/detalhe/342657/galinhas-e-frangos>.  
Acesso em: 13/04/2021

MFRURAL. Aquecedor para granjas.Equipamentos para avicultura. 2021. Disponível em:<https://www.mfrural.com.br/detalhe/365669/aquecedor-para-granjas>.  
Acesso em: 13/04/2021

SENAR. Santa Catarina. Vai construir seu aviário. Não perca as dicas com especialista. 2017. Disponível em: < http://www2.senar.com.br/Noticias/Detalhe/9709>  
Acesso em: 13/04/2021

MERCO AVES. Bovans White. Guia do produto. 2017. Disponível em: < https://www.mercoaves.com.br/pdf/bw\_cs\_c\_pguide\_7150.pdf>

Acesso em: 15/04/2021