UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO

Bruno de Oliveira Campos RA: 2013833

Edson Donizetti de Mello Filho RA:2003314

Gabriel Pires Arnecke RA: 2005294

José Augusto Lopes Martins RA: 2007198

Pedro Otavio Rodrigues dos Reis RA: 2014202

Renan Felipe Silva de Jesus RA: 2005129

E-Agricultor

Vídeo de apresentação do Projeto Integrador

https://youtu.be/cSNIZXby7s0

Repositório do código

https://github.com/Gabriel4Arnecke/E-Agricultor.git

São Paulo - SP
2022

UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO

E-Agricultor

Relatório Técnico-Científico apresentado na disciplina de Projeto Integrador para o curso de Engenharia da Computação da Universidade Virtual do Estado de São Paulo (UNIVESP).

São Paulo - SP
2022

CAMPOS, Bruno de Oliveira; FILHO, Edson Donizetti de Mello; ARNECKE, Gabriel Pires; MARTINS, José Augusto Lopes; REIS, Pedro Otavio Rodrigues dos; JESUS, Renan Felipe Silva de Jesus. **E-Agricultor.** 24f. Relatório Técnico-Científico. Engenharia da Computação – **Universidade Virtual do Estado de São Paulo**. Tutor: Eduardo Palhares Junior. Polos Américo Brasiliense, Araraquara, Paraibuna, Santa Cruz do Rio Pardo, São José dos Campos (CEFE/Santana), 2022.

RESUMO

Este trabalho tem como objetivo o desenvolvimento de uma plataforma WEB, utilizando as tecnologias bases como HTML5, CSS3 e JAVASCRIPT, também como complemento para a aplicação do projeto será necessário o uso de banco de dados (MySQL) e armazenamento em nuvem, destinado a reunir informações e exibi-las de uma maneira que possa suportar o processo decisório de produtores rurais. Uma dificuldade a ser solucionada pela plataforma é grande problemática que o produtor rural tem ao necessitar consultar informações em diversos meios para então poder sustentar o processo de decisão relacionado a sua cultura.

PALAVRAS-CHAVE: HTML5, CSS3, JAVASCRIPT; Banco de dados; SQL; Bootstrap; Versionamento; GitHub; FLASK; API.

SUMÁRIO

1. INTRODUÇÃO

O trabalho em destaque é um estudo de desenvolvimento de uma plataforma web que tem por objetivo organizar em um único portal informações importantes para suportar o processo decisório do produtor rural em ralação a sua cultura.

Na sétima edição da pesquisa de hábitos do produtor rural pela ABMRA (Associação Brasileira de Marketing Rural e Agro) 24% dos produtores entrevistados apontaram que os problemas com o clima é o fator principal no quesito preocupação com a cultura.

É de conhecimento comum que o clima é um fator importantíssimo no que diz respeito a culturas, já que para cada tipo de cultivo se faz necessário que vários parâmetros meteorológicos estejam dentro de uma certa curva característica, podemos citar como exemplo o nível de precipitação e a média de temperatura no período.

A tomada de decisão errada, não tomando por base dados meteorológicos, pode gerar atraso no ciclo produtivo entre o plantio e a colheita, ou no pior dos casos, a perda total do cultivo.

É comum, para aqueles que acompanham o agronegócio, verificarem notícias a respeito de produtores que perderam toda a sua cultura por algum engano no processo decisório, muitas das vezes essa perda se dá relacionada a condições meteorológicas.

Contudo, o acesso as informações meteorológicas, que podem auxiliar o produtor a tomar a melhor decisão referente ao planejamento e ações efetivas com relação a sua cultura, pode demandar muito tempo entre acessar diversos portais ou aplicativos para obter um painel dos dados de maneira a garantir uma quantidade mínima para que se faça uma boa definição.

Levando em consideração a importância de uma escolha assertiva, já que uma decisão tomada sem embasamento pode e como é possível de se verificar nas mídias gerais e especificas significar a perda de toda uma cultura causando grande dano financeiro e social, este projeto tem por objetivo construir um portal que possa reunir as principais informações, que foram requisitadas na pesquisa de campo, afim de que após realizado o login, o produtor tenha acesso a informações organizadas de acordo com a geolocalização da sua cultura (informado no cadastro).

O portal a ser desenvolvido tem características para que possa ser evoluído, permitindo a integração com sensores das características intrínsecas da cultura, gerando assim ainda maior assertividade na tomada de decisão.

Com o portal na web, produtores rurais de qualquer lugar com acesso à internet, poderão tomar suas decisões de forma mais assertivas com o auxílio de dados probabilísticos do clima em tempo real e outros mais, tornando assim um ambiente mais dinâmico e democrático.

2. DESENVOLVIMENTO

2.1 Objetivos

Os objetivos específicos do projeto de pesquisa são:

- Identificar um problema que possa ser solucionado através do tema proposto pela universidade.
- Interagir e verificar junto à comunidade específica de pequenos produtores. informações quanto a aceitação de uma plataforma web para centralização de informações dos mais diversos tipos para suportar a tomada de decisão.
- Criar um front-end ligado a um banco de dados que seja amigável e de fácil acesso para assim facilitar a exibição da informação
- Analisar a ferramenta desenvolvida e verificar junto à comunidade se ela atende os requisitos iniciais.

2.2. Justificativa e delimitação do problema

O problema levantado pelo grupo foi: “Como podemos facilitar o processo decisório do produtor rural a fim de alavancar sua produção?”

Em nossas pesquisas observamos que grande parte dos produtores rurais consultados não ficam confortáveis ao tomar decisões baseados em informações encontradas na rede mundial de computadores, já que ela se encontra fragmentada em diversos portais.

Por tanto surgiu a ideia de criar uma plataforma web amigável para que as informações principais, também por eles citadas em nossas pesquisas, estejam mais visuais e reunidas em um único ponto de acesso.

2. 3. Fundamentação teórica

Público-alvo

O público a qual o projeto se destina são os produtores rurais no qual se encontram com problemas de controle e gestão do seu cultivo.

Com certa frequência pode-se observar notícias como a seguinte na mídia falada ou escrita “Agronegócio tem perdas de R$ 45 bilhões com seca e onda de calor no Sul e Centro-Oeste" (Portal DBO, 13/01/2022).

Para o pequeno produtor, uma decisão tomada pode ser a escolha entre a fartura e o total prejuízo.

Framework para desenvolvimento web

Os frameworks são ferramentas que funcionam como uma biblioteca de arquivos, armazenando diversas funções básicas para o desenvolvimento de uma aplicação. O objetivo dos frameworks é diminuir o nível de complexidade e, consequentemente, o tempo gasto para o desenvolvimento web. Cada framework é diferente com seus prós e contras e ditam o fluxo de trabalho de desenvolvimento, portanto, a seleção do framework é uma decisão estratégica dentro do projeto (SAKS, 2019).

Dentre os vários frameworks disponíveis para o desenvolvimento web alguns tomaram destaques para a necessidades que apareceram no cenário. Bootstrap sanou todas as dificuldades para a base da plataforma no quesito front-end, com este framework o site se tornou responsivo e de fácil desenvolvimento. Bootstrap é um framework web com código-fonte aberto para desenvolvimento de componentes de interface e front-end para sites e aplicações web usando HTML, CSS e JavaScript, baseado em modelos de design para a tipografia, melhorando a experiência do usuário em um site amigável e responsivo. Com um vasto fórum e suporte da ferramenta, todas as dúvidas e aplicações tiveram fluxo para chegar na solução proposta. Bootstrap contém utilizações no CSS3 e Jquery já prontas para o uso.

Flask também foi um dos frameworks aplicados na plataforma, desta vez voltado para o back-end com foco na manipulação e controle dos dados ali trafegados. A ferramenta se dispõe de uma gama de aplicações e exemplos que facilitaram em sua utilização. Como ponto a se destacar na decisão da escolha dessa tecnologia, foi o uso a linguagem Python como base de seu código fonte e sua documentação bem estabelecida.

Sistemas de controle de versão

O sistema de controle de versão (VCS – *version control system*) corresponde a um sistema que gerencia a atividades e processos para o controle da evolução de diferentes de componentes ou itens de configuração durante o desenvolvimento ou a manutenção de um software. O controle de versão pode ser feito por ferramentas de automatização que registram

todas as alterações feitas pelos desenvolvedores e garantem a manutenção das aplicações (ZOLKIFLI; NGAH; DERAMAN, 2018).

O processo de desenvolvimento de softwares, modificações como adição ou exclusão de recursos, são frequentes. Essas alterações demandam um sistema que possa gerenciar e organizar os códigos de forma a tornar o processo mais simples e rápido. Ademais, o VCS possibilita que vários desenvolvedores atuem no projeto conjuntamente e simultaneamente (ZOLKIFLI; NGAH; DERAMAN, 2018).

O Git é um dos VCS mais conhecidos e utilizados no mundo. O VCS do Git não armazena arquivos redundantes e tem quase todas as suas operações armazenadas localmente, garantindo que problemas devido à latência de comunicação de uma rede não afetem o desempenho. O conteúdo passa por verificação de integridade, o que significa que é impossível mudar o conteúdo de qualquer arquivo ou diretório sem que o Git tenha conhecimento. O VCS permite reversão das alterações, permitindo que erros sejam reparados rapidamente (VUORRE; CURLEY, 2018).

A interface de usuário e o mecanismo de distribuição para repositórios Git é fornecida pelo GitHub. Com o GitHub é possível criar um hiperlink para um arquivo ou local específico em um arquivo, em uma versão específica e controlar as permissões de quem pode ver, editar e administrar um projeto (VUORRE; CURLEY, 2018).

Arquitetura cliente -servidor

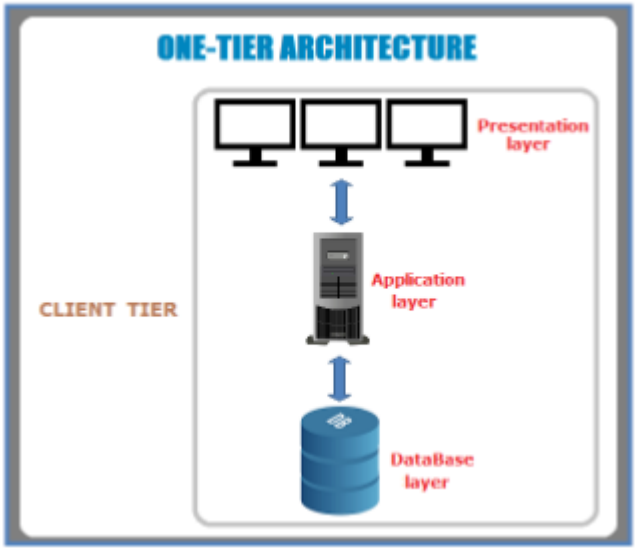
A arquitetura cliente-servidor pode ser definida como uma plataforma base para que as aplicações executam um processamento, onde o cliente solicita a ação e uma servidora responde por meio de protocolos de comunicação. O cliente é um processo ativo na relação que interage com o usuário por meio de uma interface gráfica ou não, permitindo consultas ou comandos para recuperação de dados e análise. O servidor, por sua vez, é um processo reativo que fornece um determinado serviço disponível para qualquer cliente que o requisite. O servidor recebe e responde às solicitações dos clientes, presta serviços e atende a vários clientes simultaneamente (RENAUD,1994). Dentre as vantagens desse tipo de arquitetura, podemos destacar (SALEMI,1993):

- Confiabilidade
- Capacidade de processamento: tarefas podem ser feitas sem a monopolização dos recursos e usuários finais podem trabalhar localmente.
- Escalabilidade do sistema

- A individualização dos ambientes operacionais (cliente/servidor) permite a conjugação de plataformas para melhor atender às necessidades específicas.

A arquitetura cliente-servidor categorizada em quatro tipos: arquitetura de uma camada, arquitetura de duas camadas, arquitetura de três camadas e arquitetura de n-camadas. Todos os tipos de arquitetura possuem todas as camadas – apresentação, lógica de negócios e dados – entretanto, para a arquitetura de uma camada, todas as camadas estão em um único pacote de software (Figura 1).

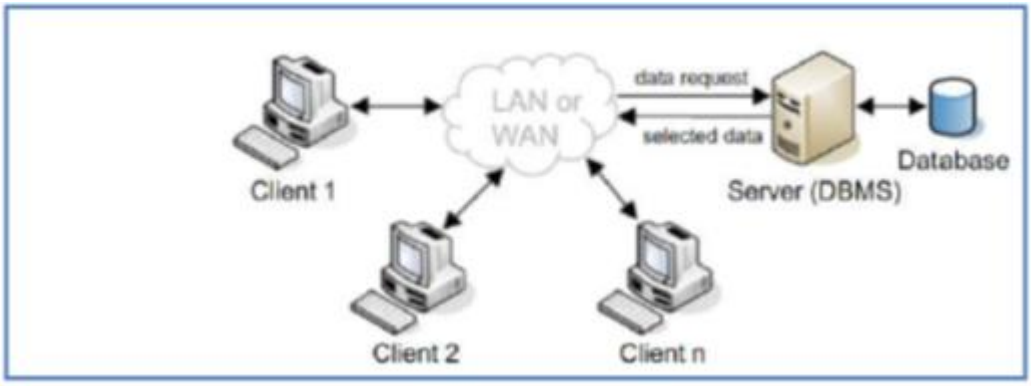
Figura 1 – Arquitetura cliente-servidor de uma camada



Fonte: Kumar (2019).

Na arquitetura de duas camadas, existem duas camadas: cliente (camada cliente) e banco de dados (camada de dados) O sistema cliente tem a camada de apresentação e as camadas de aplicativo e o sistema servidor lida com a camada de dados. O dispositivo cliente envia a solicitação ao servidor e o servidor processa a solicitação e responde ao sistema cliente (Figura 2).

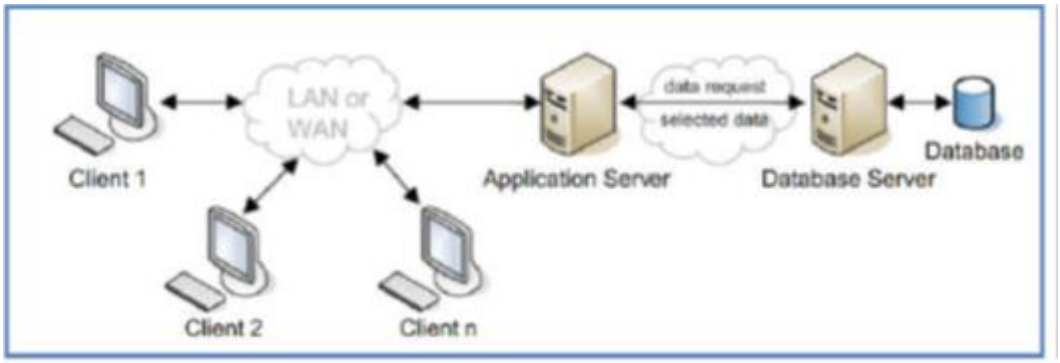
Figura 2 – Arquitetura cliente-servidor de duas camadas



Fonte: Kumar (2019).

O tipo de arquitetura de três camadas é dividido em três partes: a camada de apresentação controlada pelo sistema cliente, a camada de lógica de negócios gerenciada pelo servidor de aplicativos e a camada de dados controlada pelo sistema de servidor de banco de dados (Figura 3).

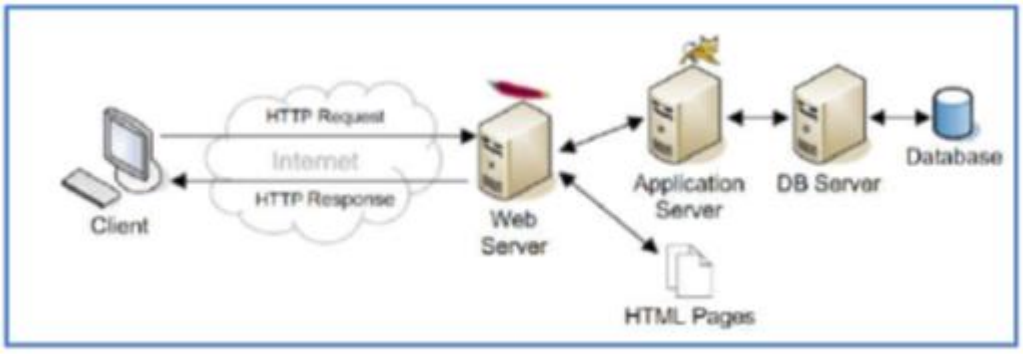
Figura 3 – Arquitetura cliente-servidor de três camadas



Fonte: Kumar (2019).

O tipo de arquitetura de n-camadas é semelhante à arquitetura de três camadas, porém o número de servidores de aplicativos é maior e representa camadas individuais para que a camada de lógica de negócios seja distribuída (Figura 4).

Figura 4 – Arquitetura cliente-servidor de n-camadas



Fonte: Kumar (2019).

Banco de dados

Um banco de dados é uma coleção de dados que ficam armazenados e que tem um significado tácito. O banco de dados é também uma representação de algumas características ou aspectos do mundo real e tem um objetivo específico, como por exemplo criar um cadastro de pessoas (ELMASRI; NAVATHE, 2018).

Os bancos de dados podem ser gerenciados por meio de sistemas de gerenciamento de banco de dados (SGBD) que “facilita o processo de definição, construção, manipulação e

compartilhamento de banco de dados entre diversos usuários e aplicações", permitindo que os usuários criem e mantenham o banco de dados (ELMASRI; NAVATHE, 2018).

Existem quatro tipos de usuários em um SGBD (MACHADO, 2020):

- Administrador de banco de dados (DBA – database administrator): supervisiona e gerencia os recursos, monitora as falhas na segurança e o tempo de resposta do sistema.
- Projetistas de banco de dados (database designers): identificam os dados a serem armazenados e projetam as estruturas apropriadas para representar e armazenar esses dados.
- Analistas de sistemas e programadores de aplicação: responsáveis por criar a solicitação de administração de dados, elaborar os modelos conceitual, lógico e físico do que está sendo solicitado e executar o script em ambiente de desenvolvimento.
- Usuários finais: são pessoas cujo acesso ao banco de dados é requerido para consulta, atualização ou geração de relatórios.

Algumas características são fundamentais para o SGBD, como a natureza autodescrição; o isolamento entre programas e dados e a abstração de dados; suporte de múltiplas visões do dado; e o compartilhamento dos dados e o processamento de transação multiusuário. Além dessas características, um bom SGBD deve apresentar algumas vantagens e capacidades. O controle de redundância impede que o mesmo dado seja armazenado várias vezes, o que pode levar a falhas na atualização do registro, por exemplo. O controle de acesso previne que usuários acessem dados sem que estejam autorizados. Armazenamento persistente para objetos do programa em que códigos e estruturas de dados são armazenados e gerenciados pelos SGBDs como objetos e podem ser acessados a partir de funcionalidades oferecidas pelo sistema. Sistemas de backup e recuperação que realizam a recuperação dos dados após a ocorrência de falhas de software e hardware. Restrições de integridade para os dados armazenados em um banco de dados (ELMASRI; NAVATHE, 2018).

A abordagem relacional representa uma forma de descrever o banco de dados por meio de conceitos que são compreensíveis por parte dos usuários, mas que não estão longe da maneira como os dados são armazenados, pois os usuários veem o banco de dados relacional como um conjunto de tabelas bidimensionais, divididas em linhas e colunas. Esse modelo que constrói um conjunto de objetos básicos a partir da percepção do mundo real e definem uma técnica de diagramação para modelos de dados é chamado Modelo de Entidade-Relacionamento (MER). A técnica de diagramação para MER é conhecida como Diagrama de Entidades e

Relacionamentos (DER) e é composta pelos elementos entidades, atributos, chaves, relacionamentos, cardinalidades (MACHADO, 2020).

A entidade é um objeto básico do MER usado para representar uma coisa no mundo real, física ou conceitual. O atributo é uma propriedade que descreve uma entidade. Chave é um tipo entidade que, usualmente, tem um ou mais atributos que assumem valores distintos para cada entidade individual dentro do conjunto entidade. Os relacionamentos representam as associações existentes entre as entidades. Por fim, a cardinalidade especifica o número máximo de instâncias de relacionamento nas quais uma instância de entidade pode participar (MACHADO, 2020).

JavaScript

JavaScript é uma linguagem de programação de propósito geral, dinâmica e possui características do paradigma de orientação a objetos. Essa linguagem é completa e tem muitas qualidades como: listas associativas, tipagem dinâmica e expressões regulares de Perl e a sintaxe similar a C e C++. Além disso, JavaScript é multiparadigma e possui funções de ordem superior (CROCKFORD, 2008).

Na tipagem dinâmica, as variáveis podem assumir qualquer tipo ou objeto definido pela linguagem, por isso não é necessário definir o tipo da variável no momento da sua declaração. Já as funções de ordem superior possibilitam a criação de funções que a partir de outras funções simples são capazes de realizar ações mais complexas (CROCKFORD, 2008).

O JavaScript permite desenvolver pequenos programas embutidos no próprio código de uma página HTML que possibilitam o processamento de alguns dados, a verificação de formulários e criação e alteração de valor de elementos HTML diretamente no computador cliente, evitando a troca de informações com o servidor. Com isso, o tempo de processamento passa a depender somente do local do cliente (CROCKFORD, 2008).

O núcleo da linguagem JavaScript é composto por comandos e elementos menores que conjuntamente com a lógica de programação para dos relacionamentos possibilita a criação de uma aplicação. Um dos elementos da linguagem é o tipo de dado que pode ser numérico, booleano, indefinido, null, string e array. Os principais operadores que compõem o núcleo JavaScript são: aritmético, comparação, bit a bit, atribuição e lógico. As estruturas if/else, switch/case, while, do/while, for e for/in são denominadas estruturas de controle. As funções possuem um papel muito importante na programação estrutural, pois viabiliza a divisão do programa em partes menores e logicamente relacionadas. Quanto aos objetos, o JavaScript possui dois objetos nativos muito importantes: arrays e strings (CROCKFORD, 2008).

2.4. Aplicação das disciplinas estudadas no Projeto Integrador

Neste tópico apresentaremos as disciplinas que foram norteadoras na execução de todo o projeto, seguindo a seguinte estrutura: Nome da Disciplina, breve resumo da disciplina conforme consta no plano pedagógico e nosso comentário em relação a aplicação da mesma.

2.4.1 Leitura e produção de Texto

Prática de leitura e de produção de textos de diversos gêneros. Noções fundamentais sobre estrutura e conteúdo: coesão, coerência, clareza, informatividade e adequação. Revisão e reescrita orientada dos textos produzidos. Subsidiar os estudantes para a produção textual.

A disciplina de português é e sempre será a principal norteadora na criação dos trabalhos, tendo em vista que um relatório técnico-científico deve ser escrito seguido a norma formal da Língua Portuguesa e em um tipo específico de redação.

2.4.2 Pensamento computacional

Navegação, pesquisa e filtragem de informações. Interação por meio de tecnologias. Compartilhamento de informações e conteúdo. Colaboração por canais digitais. Raciocínio lógico, análise e resolução de problemas. Estudo dos dispositivos computacionais. Noção de algoritmos. Práticas de computação. Jogos de lógica. Desenvolvimento de conteúdo. Construção de narrativas usando programação com blocos.

Utilizamos a disciplina de pensamento computacional como base no momento da realização do nosso *brainstorming* a fim de obter uma análise rápida do problema a ser escolhido e de maneiras que este problema pode ser abordado.

2.4.3 Fundamentos de Internet e Web

Hipertexto; origens de XML e HTML (SGML); Estrutura do conteúdo versus aparência do documento; tags HTML básicas (H1, P, etc) e extensíveis (div, span, etc); tags de HTML5; Atributos básicos de CSS (color, text-align, etc), seletores CSS; formulários HTML; noções de manipulação programática do DOM.

Disciplina largamente utilizada já que o projeto integrador requer a criação de um software WEB, para tanto utilizamos os conhecimentos adquiridos em HTML5 e em CSS para estruturar o *front-end* do protótipo a fim de tornar a plataforma amigável para o usuário.

2.4.3 Algoritmos e Programação de Computadores I e II

Compreender conceitos básicos de programação e descrever algoritmos para resolver problemas utilizando a linguagem de programação Python, incluindo depuração e testes automatizados básicos.

Reforçar a prática de programação em Python, composição de programas com múltiplos arquivos de código fonte, uso de bibliotecas, APIs (WEB) e GUIs. Noções de programação orientada a objetos.

Ao escolher o *framework web* a ser utilizado na criação do projeto, optou-se pela utilização do FLASK, este é um *framework web* que utiliza-se da linguagem python para seu funcionamento, portanto, tomou-se crédito das disciplinas supracitadas a fim de escrever o código do *back-end* que possibilita-se o uso de API e uma boa navegação da página assim que a mesma estivesse publicada.

2.4.4 Banco de Dados

Introdução a banco de dados (Processamento de Arquivos vs SGBDs, arquitetura de SGBDs), modelagem de dados (conceitual, modelo entidaderelacionamento), Linguagem de Consulta e Manipulação de Dados (SQL), Indexação, Bancos de Dados Não Relacionais ou NoSQL (Bancos de Dados Orientados a Documentos, Bancos de Dados em Colunas, Bancos de Orientados a Grafos). Mapeamento objeto-relacional (ORM).

Para o gerenciamento de uma plataforma web, visando controle individual de culturas, como é a proposta de nossa solução, faz-se necessário um controle de acesso ao portal para que garanta que o usuário receba apenas informações relacionadas a sua região, para tanto utilizamos da disciplina de banco de dados para cadastrar usuários e senhas bem como as informações inseridas pelos mesmos no portal.

2.4.4 Engenharia de Software

Noções de Processo de Software (Modelos de Ciclos de Vida Clássicos e Ágeis), Engenharia de Requisitos (Técnicas de Elicitação ou Levantamento de Requisitos), Arquitetura e Projeto de Software (Estilos Arquiteturais, Padrões de Projeto, Refatoração, Anomalias, Reutilização com Componentes e Frameworks), Testes (Estratégias de Teste, Desenvolvimento Dirigido por Testes, Teste Funcional e Estrutural, Teste de Desempenho e Segurança), Entrega Contínua (Integração Contínua, Testes Automatizados, Contêineres, Gerência de Configuração e Processos de Liberação de Software).

Com nosso público alvo, foi necessário levantar as características que a demanda fosse atendida e para que o projeto da aplicação atendesse os requisitos do padrão do projeto estipulado no início da criação. A disciplina foi norteadora na definição do escopo do projeto bem como a forma de se aplicar os testes e os requisitos de segurança do sistema.

2.5. Metodologia

O objetivo do capítulo é apresentar os métodos para o desenvolvimento da aplicação Web, desde a identificação do problema até a prototipação da solução.

Tipo de Estudo

A pesquisa tem caráter aplicado porque visa o desenvolvimento de uma aplicação Web para conectar os produtores rurais a um portal com as informações que podem ser convenientes de forma a suportar o processo decisório em relação a cultura que eles desenvolverão. A metodologia pode ser dividida em dois blocos:

- Pesquisa de campo e documental – com aplicação da metodologia de *design thinking* para idealização da solução e revisão bibliográfica para o estado da arte da tecnologia.
- Pesquisa tecnológica - com o desenvolvimento funcional da solução Web, utilizando-se de vários tipos de ferramentas e técnicas computacionais.

Local do estudo

Todas as etapas do estudo foram construídas através do WhatsApp, videochamadas, pesquisas e sob supervisão do orientador do curso (UNIVESP).

Métodos de desenvolvimento

Para o cumprimento dos objetivos específicos apresentados, o presente estudo foi dividido em três etapas distintas:

- Etapa I: Pesquisa de campo e documental;
- Etapa II: Desenvolvimento da aplicação Web;
- Etapa III: Avaliação da aplicação Web.

Etapa I: Pesquisa de campo e documental

No primeiro passo para o desenvolvimento do projeto foi aplicada a abordagem do *design thinking*. Esse método é uma abordagem que vai além da necessidade de criar um

produto ou serviço para oferecer soluções inovadoras focadas nas necessidades do público-alvo. Os pilares do *design thinking* são a praticabilidade, a viabilidade e a desejabilidade, que devem estar em equilíbrio para que o resultado do projeto seja satisfatório. A praticabilidade considera se é possível, tecnologicamente, entregar o que está sendo proposto, isto é, a viabilidade técnica. Enquanto esses dois pilares são pontos de atenção na maioria dos projetos, a desejabilidade por vezes é menosprezada pelos gestores. Um dos diferenciais do *design thinking* em relação a outros métodos está na busca pela harmonia entre os três pilares a partir do entendimento das necessidades dos clientes. Um Mapa de Empatia foi usada como ferramenta na primeira etapa do *design thinking*, conhecida como entendimento. Neste mapa, seis perguntas foram feitas a fim de vislumbrar um panorama geral da situação: o que o público-alvo pensa e sente? O que ele escuta? O que ele vê? O que ele fala e faz? Quais são suas dores? Quais são seus desejos? (BROWN, 2020).

Depois de definido o problema e a possível solução, foram feitos uma revisão bibliográfica para estudo aprofundado do tema e prospecção de soluções já existentes. A partir do estado da arte do tema pesquisado, foi dado início a criação da front-end do portal:

Etapa II: Desenvolvimento da aplicação Web

Esta etapa tem como propósito desenvolver um protótipo de uma aplicação Web, destinado a indexação dos dados e links via API para que o publico alvo possa se cadastrar e acessar diretamente informações relevantes a respeito da sua cultura.

O JavaScript foi a linguagem escolhida pelo grupo para o desenvolvimento da aplicação, dadas as características e a familiaridade do grupo com essa linguagem. O framework para desenvolvimento Web será o FLASK. O controle de versão será feito por meio do GitHub. O MySQL será usado como banco de dados relacional.

O desenvolvimento será realizado por meio de um processo interativo onde será possível retornar a qualquer fase anterior sempre que for necessário aperfeiçoar o sistema.

Etapa III: Avaliação da aplicação Web

A última etapa desta pesquisa está relacionada ao segundo e último objetivo específico: o protótipo deverá ser avaliado pelo público-alvo, para garantir a usabilidade do portal e se atende os requisitos levantados na pesquisa inicial.

3. RESULTADOS

3.1. Solução inicial

Visto que o resultado da pesquisa mostrou-nos que grande maioria dos produtores que se dispuseram a responder a pesquisa proposta, apontou que centralizar em apenas um lugar onde se pode encontrar toda a informação relacionada a meteorologia, temperatura local e medidas sensoriadas seria uma forma de facilitar e agilizar o processo decisório de pequenos agricultores. O grupo propôs o desenvolvimento de uma plataforma onde tenha todas a necessidades citas implementada em um só lugar.

A primeira ideia levantada foi um serviço web onde pode-se reduzir os prejuízos agrícolas ocasionado pelo fator climático e físicos. A falta de dados para se embasar e tomar as decisões necessários conforme o cenário meteorológico de cada dia foi o principal fato que a pesquisa apontou, visto isso, surgiu a ideia da integração de uma API de dados meteorológicos com o serviço web.

3.2. Solução Final

Os sistemas tecnológicos necessários para o desenvolvimento da estrutura foram *home page (imagem 2)* onde explica o contexto do projeto, *login page (imagem 3)* onde acontece o controle e gerenciamento de acesso, uma parte de *dashboard (imagem 4)* onde apresente todos os dados necessário ao usuário.

Home page é o portal inicial do projeto onde temos em detalhe como é a aplicação do projeto. Essa etapa foi desenvolvida para mostrar ao usuário um pouco mais do que a plataforma é em si e também para ter uma interface de navegação inicial.

Foram usados estrutura do front-end (html5, css3, javascript) para o desenvolvimento da página e para controle de rotas foi utilizado o flask framework python.

Imagem 2 – home page E-agricultor

Login page é a parte onde os usuários podem acessar seu dashboard personalizados conforme os cadastros de cada usuários em si. Essa etapa acontece a verificação e validação do usuário

Foram usados estrutura do front-end (html5, css3, javascript) para o desenvolvimento da página de *login* e para controle de rotas e verificação da validação do usuário foi utilizado o flask framework python juntamente com o banco de dados SQLITE.

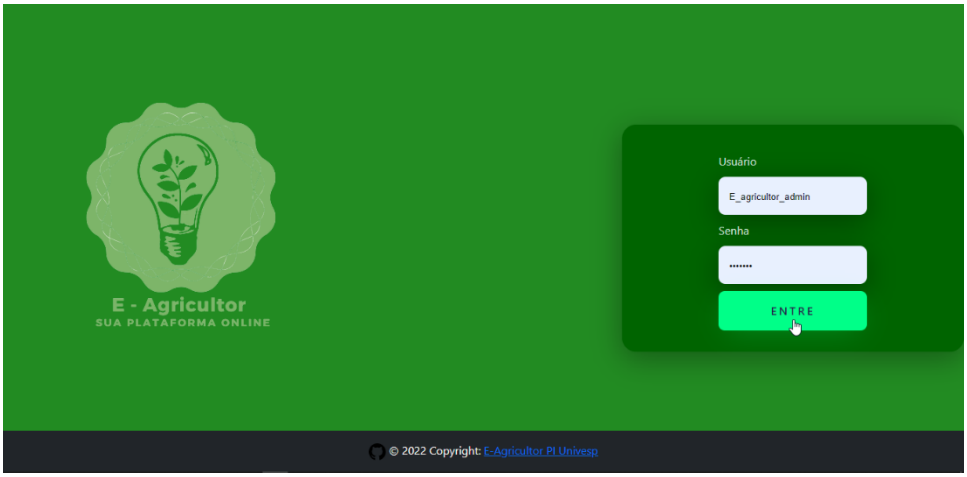


Imagem 3 – login page E-agricultor

A página de Dashboard é onde se concentra todo os dados necessários para o usuário, aqui encontra-se dados metrológicos especifico da cidade cadastrada e também tem a opção de pesquisar por uma nova cidade.

Foram usados estrutura do front-end (html5, css3, javascript) para o desenvolvimento da página de *dashboard*, para controle de rotas e verificação da validação do usuário, controle do

sensoriamento foi utilizado o flask framework python juntamente com o banco de dados SQLITE e para finalizar essa etapa foi utilizado uma API do *OpenWeather* que fornece os dados meteorológico de lugar no qual o usuário foi cadastrado.

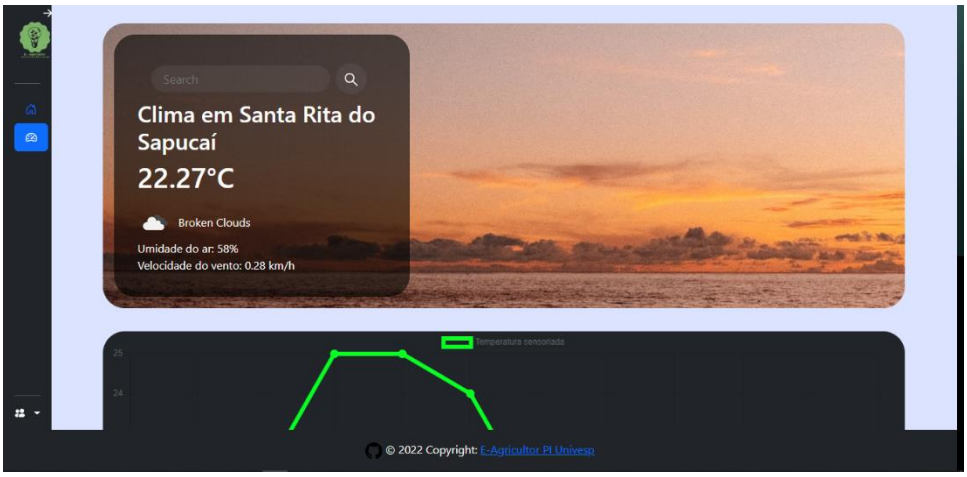


Imagem 4 – Dashboard E-agricultor

Para a concluir o projeto, foi usado o HerokuApp (imagem 5) como plataforma *HOST* para armazenar e hospedar o software web. O link para acesso é https://eagricultor.herokuapp.com

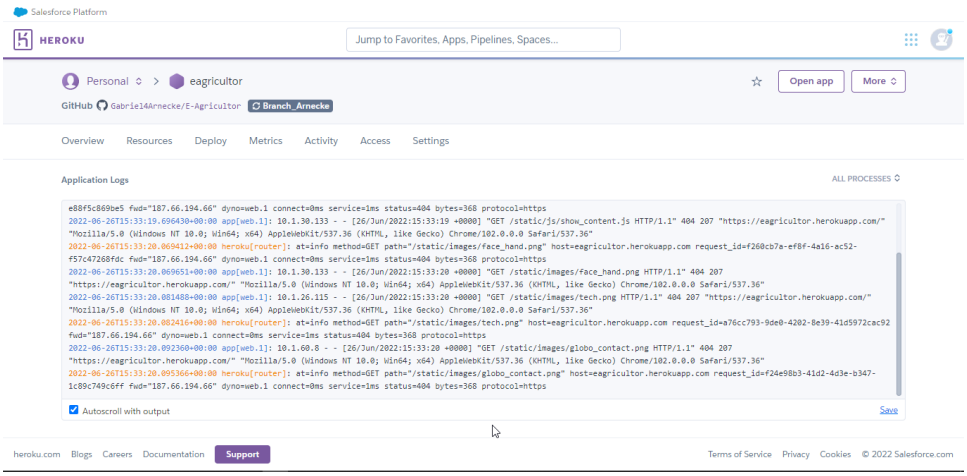


Imagem 5 - E-agricultor hospedado no HerokuApp

O versionamento do projeto foi realizado pela plataforma do GitHub (imagem 6). O link para acesso do projeto no GitHub é https://github.com/Gabriel4Arnecke/E-Agricultor/

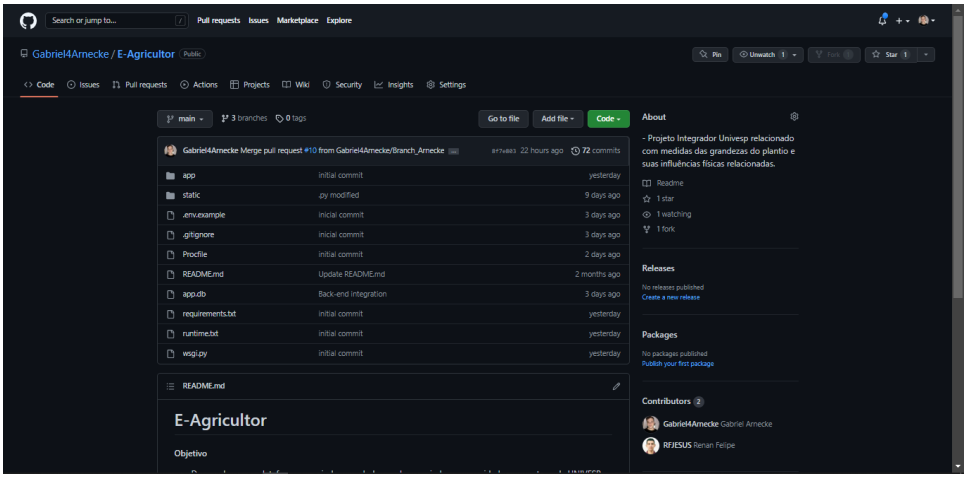


Imagem 6 – Versionamento GitHub E-agricultor

4. CONSIDERAÇÕES FINAIS

O desenvolvimento da plataforma denominada E-Agricultor tem como principal objetivo ser uma ferramenta que poderá facilitar e suportar um processo de decisão por parte do produtor rural.

Como podemos observar na contextualização deste relatório, dentre os fatores que podem vir a trazer grandes prejuízos para os produtores é o fator clima. Através da construção do portal, controlado via inserção de usuário e senha para adentrar no portal onde os dados específicos de sua região são mostrados em tempo real e onde é possível coletar um período amostral para tomar decisões como momento do plantio, tipo de irrigação e ou até mesmo métodos de controle de praga.

Nosso website visa fornecer autonomia e centralização de dados de uma forma mais clara e concisa em comparação com as diversas fontes que a comunidade tem a sua disposição desconsiderando a plataforma ora criada.

A plataforma está limitada a um local de cultura bem como a um pequeno armazenamento de dados, sendo assim um grande ponto disposto para melhoria, pois o produtor pode e é prática comum, setorizar as culturas e cada tipo de cultura demanda um tipo de alerta climático.

Ao retornar à comunidade onde observamos a necessidade de tal ferramenta e a implantar, pode-se notar a melhora no acesso aos dados e o contentamento do produtor em ter a ferramenta na palma da mão quando o desejar.

Afim de atender os requisitos deste etapa do projeto integrador, cujo qual também é um dos objetivos principais do desenvolvimento da plataforma, pode-se concluir que de fato atendemos os requisitos requisitados, contudo é possível melhorar a plataforma e profissionalizar de forma que pode vir a ser uma ferramenta elementar para a manutenção e melhora de cadeia de produção de diversos setores da agricultura.

REFERÊNCIAS

ABNT – Associação Brasileira de Normas Técnicas. NBR 14724: Informação e documentação. Trabalhos Acadêmicos - Apresentação. Rio de Janeiro: ABNT, 2002.

NIELD, T.; Introdução à Linguagem SQL: Abordagem prática para iniciantes. 1 ed 2019 São Paulo: Novatec Editora, 2019

O trabalho deverá ser redigido conforme recomendações das Diretrizes para confecção de teses e dissertações da Universidade de São Paulo (USP), disponíveis em: <http://www.teses.usp.br/index.php?option=com_content&view=article&id=52&Itemid=67>. Acesso em 14 abr. 2022.

ZOLKIFLI, Nazatul Nurlisa et al. Version Control System: A Review. 2018. Disponível em: https://www.sciencedirect.com/science/article/pii/S1877050918314819?ref=pdf_download&fr=RR-2&rr=7107cfee9d3ab051. Acesso em: 24 maio 2022.

AGRONEGÓCIO TEM PERDAS DE R$ 45 BILHÕES COM SECA E ONDA DE CALOR NO SUL E CENTRO-OESTE. São Paulo, 13 jan. 2022. Disponível em: https://www.portaldbo.com.br/agronegocio-tem-perdas-de-r-45-bi-com-seca-e-onda-de-calor-no-sul-e-centro-oeste/. Acesso em: 24 maio 2022.

SAKS, Elar. JavaScript Frameworks: Angular vs React vs Vue. 2019. 42 f. Tese - Curso de Business Information Technology, Haaga-Helia University, Finlandia, 2019. Disponível em: https://www.theseus.fi/bitstream/handle/10024/261970/Thesis-Elar-Saks.pdf?sequence=2&isAllowed=y. Acesso em: 24 maio 2022.

VUORRE, Matti & Curley, James. A Tutorial on the Git Version Control System. Advances in Methods and Practices in Psychological Science, 2018. Disponível em: https://journals.sagepub.com/doi/10.1177/2515245918754826. Acesso em: 24 maio 2022.

A IMPORTÂNCIA DO CLIMA PARA O AGRONEGÓCIO. Canal Agro - Estadão, 19 maio 2020. Disponível em: https://summitagro.estadao.com.br/noticias-do-campo/importancia-clima-agronegocio/. Acesso em: 24 maio 2022.

COMO A TECNOLOGIA PODE AJUDAR A AGRICULTURA FAMILIAR A LIDAR COM O CLIMA? Campinas, 14 abr. 2022. Disponível em: https://agrosmart.com.br/blog/agricultura-familiar/. Acesso em: 24 maio 2022.

PESQUISA APONTA OS PRINCIPAIS DESAFIOS ENFRENTADOS PELO PRODUTOR RURAL NO BRASIL. Bauru, 27 maio 2019. Disponível em: https://g1.globo.com/sp/bauru-marilia/especial-publicitario/jacto-agricola/noticia/2019/05/27/pesquisa-aponta-os-principais-desafios-enfrentados-pelo-produtor-rural-no-brasil.ghtml. Acesso em: 24 maio 2022.

SÃO PAULO. UNIVESP. (org.). **Cursos de Bacharelado em Tecnologia da Informação, Ciência de Dados e Engenharia de Computação**: projeto pedagógico dos cursos. Projeto Pedagógico dos Cursos. 2020. Disponível em: https://apps.univesp.br/manual-do-aluno/assets/PPC/engenharia-da-computacao/PPC-BTI.pdf. Acesso em: 24 jun. 2022.