UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO, DE CIÊNCIAS EXATAS E EDUCAÇÃO
DEPARTAMENTO DE ENG. DE CONTROLE, AUTOMAÇÃO E COMPUTAÇÃO
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Guilherme Renkel Wehmuth

Ferramenta Web para Processamento de NDVI Através de Imagens de Satélite com Alta Resolução Temporal

Blumenau
2022

Guilherme Renkel Wehmuth

Ferramenta Web para Processamento de NDVI Através de Imagens de Satélite com Alta Resolução Temporal

Trabalho de Conclusão de Curso de Graduação em Engenharia de Controle e Automação do Centro Tecnológico, de Ciências Exatas e Educação da Universidade Federal de Santa Catarina como requisito para a obtenção do título de Engenheiro de Controle e Automação.
Orientador: Prof. Mauri Ferrandin, Dr.

Blumenau
2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Wehmuth, Guilherme Renkel
Ferramenta web para processamento de NDVI através de imagens de satélite com alta resolução temporal / Guilherme Renkel Wehmuth ; orientador, Mauri Ferrandin, 2022.
59 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Blumenau, Graduação em Engenharia de Controle e Automação, Blumenau, 2022.

Inclui referências.

1. Engenharia de Controle e Automação. 2. Agricultura. 3. Aplicação Web. 4. Índice de Vegetação. 5. Sensoriamento Remoto. I. Ferrandin, Mauri. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Controle e Automação. III. Título.

Guilherme Renkel Wehmuth

Ferramenta Web para Processamento de NDVI Através de Imagens de Satélite com Alta Resolução Temporal

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Engenheiro de Controle e Automação" e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Controle e Automação.

Blumenau, 01 de 12 de 2022.

Banca Examinadora:

Documento assinado digitalmente
Mauri Ferrandin
Data: 15/12/2022 11:47:05-0300
CPF: ***.580.869-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Mauri Ferrandin, Dr.
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Carlos Roberto Moratelli
Data: 15/12/2022 13:53:49-0300
CPF: ***.999.369-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Carlos Roberto Moratelli, Dr.
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Maiquel de Brito
Data: 15/12/2022 12:32:49-0300
CPF: ***.657.150-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Maiquel de Brito, Dr.
Universidade Federal de Santa Catarina

Dedico este trabalho a todos aqueles que, de alguma forma,
auxiliaram para a concretização desta etapa.

AGRADECIMENTOS

Agradeço a meu pai, minha mãe e a todos aqueles que estiveram presente durante essa jornada.

"Somewhere, something incredible is waiting to be known." (Carl Sagan)

RESUMO

O agronegócio desempenha um papel fundamental na economia brasileira, respondendo por mais de 43% das exportações do país e com perspectivas para crescer ainda mais nos próximos anos. Nesse cenário altamente competitivo, a busca por métodos para impulsionar os índices de produtividade das lavouras é uma crescente demanda e abre portas para tecnologias que buscam promover esse feito. Baseado nessa necessidade mercadológica, o presente trabalho descreve o desenvolvimento de uma plataforma web que permite aos agricultores realizarem em suas culturas análises do índice de vegetação por diferença normalizada (NDVI) através do processamento de imagens de satélite, índice este que apresenta forte correlação com os aspectos biofísicos da cultura. Para isso, a arquitetura do sistema desenvolvido faz uso das tecnologias React e Node.js, além de ser integrada ao Google Maps e ao Earth Engine para possibilitar a obtenção de dados georreferenciados. Ao final do desenvolvimento, a aplicação foi disponibilizada em servidor configurado na nuvem, possibilitando ser acessada pela internet através de uma URL pública.

Palavras-chave: 1. Agricultura; 2. Aplicação Web; 3. Google Maps; 4. Índice de Vegetação; 5. Javascript; 6. Sensoriamento Remoto.

ABSTRACT

The agrobusiness plays a fundamental role at brazilian economics, responding to more than 43% of its exportations and with perspectives to grow even more in the next years. In this highly competitive scenario, the search for methods to boost crop's productivity rates is a growing demand and brings opportunitys to technologies that seek to promote this accomplishment. Based on this market need, the present paper develops a web plataform that allows farmers to perform analysis at their cultures using normalized difference vegetation index (NDVI) - an index that presents a strong correlation with the biophysical aspects of the culture - through the processing of satellite images. For this purpose, the developed system design uses technologies such as React and Node.js, in addition to being integrated to Google Maps and Earth Engine to permit the processing of georeferenced data. At the end of the development process, the application was published on a cloud server, making it possible to be accessed over the internet through a public URL.

Keywords: 1. Agriculture; 2. Google Maps; 3. Javascript; 4. Remote sensing; 5.Vegetation Index; 6. Web Application.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
AWS	Amazon Web Services
CPU	Central Processing Unit
DER	Diagrama Entidade Relacionamento
DOM	Document Object Model
EC2	Elastic Compute Cloud
GEE	Google Earth Engine
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IV	Índice de Vegetação
JSON	JavaScript Object Notation
JWT	JSON Web Token
MDE	Modelo Digitais de Elevação
MDS	Modelo Digital de Superfície
MODIS	Moderate Resolution Imaging Spectroradiometer
MVC	Model View Controller
MVP	Minimum Viable Product
NAIP	National Agriculture Imagery Program
NASA	National Aeronautics and Space Administration
ORM	Object-Relational Mapping
PC	Personal Computer
SATVeg	Sistema de Análise Temporal da Vegetação
SGBD	Sistema de Gerenciamento de Banco de Dados
SPA	Single Page Applications
SQL	Structured Query Language
SR	Sensoreamento Remoto
SRTM	Shuttle Radar Topography Mission
SSH	Secure Shell
UFSC	Universidade Federal de Santa Catarina
URL	Uniform Resource Locator
USGS	United States Geological Survey
VANT	Veículo Aéreo Não Tripulado

SUMÁRIO

1 INTRODUÇÃO

No presente capítulo, é realizada uma contextualização da problemática abordada ao longo do trabalho, além de serem expostos os objetivos gerais e específicos que orientaram o seu desenvolvimento. Ao final, a estrutura do texto e seus principais tópicos são também apresentados.

1.1 CONTEXTUALIZAÇÃO DO PROBLEMA

A agricultura pode ser definida como um conjunto de procedimentos, técnicas e métodos que possibilitam a produção de insumos que tenham como finalidade o consumo ou a geração de outros produtos ou alimentos. Nesse contexto, o agronegócio pode ser compreendido como um empreendimento agrícola que busca, através da aplicação de tecnologia, intensificar a produtividade de uma propriedade rural visando o lucro (ARAUJO, M. J., 2022).

A economia brasileira tem o agronegócio como um dos seus principais geradores de riqueza, respondendo por 21% da soma de todas as riquezas produzidas no país, um quinto de todos os empregos e 43,2% das exportações brasileiras que em 2019 alcançaram o marco de US$ 96,7 bilhões (EMBRAPA, 2017). O Brasil é atualmente enquadrado como o principal exportador de suco de laranja, açúcar e café, e entre os anos de 1977 e 2017, a sua produção de grãos apresentou um crescimento superior a 500%, contribuindo para que também se tornasse o segundo maior exportador de soja e de milho no cenário global (EMBRAPA, 2018). Dentro desse cenário, o agronegócio impõe que os produtores rurais busquem altíssimos níveis de especialização e de profissionalismo com o objetivo de aumentar a capacidade gerencial e produtiva de seus empreendimentos, o que só é possível através da constante coleta de dados e informações à cerca do seu processo produtivo (SILVA NUNES, 2010). Buscando permitir ao produtor rural aumentar a eficiência do seu negócio através do entendimento dos fatores que interferem em sua lavoura, a prática da Agricultura de Precisão vem sendo adotada pelo agronegócio brasileiro desde meados de 1990.

A Agricultura de Precisão (AP) é caracterizada como um sistema de gestão de produção agrícola que busca otimizar os sistemas agrícolas através da coleta de dados georreferenciados que possibilitam identificar e atuar sobre a variabilidade espacial e temporal das áreas de cultivo, analisando para isso tanto aspectos do solo quanto da vegetação e do clima (SILVA NUNES, 2010). Para isso, a AP combina diferentes tecnologias e fontes de informação, como: características físicas e químicas do solo, mapas de produtividade e dados oriundos do processamento de imagens de satélite como os índices de vegetação. Desse modo, a gestão oriunda da AP passa a considerar a heterogeneidade de cada área da lavoura, gerenciando as plantações metro a metro para promover uma atuação pontual que gere um aumento da resposta produtiva e uma redução nos gastos com insumos agrícolas

(SILVA, 2018).

Em frente a esse cenário que evidencia a necessidade de tecnologias que promovam a rentabilidade do produtor e aumentem a produtividade da agricultura brasileira, este trabalho trata do desenvolvimento de um MVP (Minimum Viable Product) de sistema web orientada à agricultura de precisão. Por seu caráter enxuto, o produto tem como foco permitir o monitoramento do índice de vegetação de áreas agrícolas dinamicamente informadas pelo usuário. Para isso, o trabalho aborda: as tecnologias empregadas na construção da aplicação, a arquitetura utilizada para separar os diferentes contextos do sistema proposto, o cadastro das regiões agrícolas de interesse, a integração de dados por meio de diferentes serviços e, por fim, o processamento das imagens de satélite obtidas através de um mapa interativo e suas aplicações.

1.2 OBJETIVO GERAL

Este trabalho tem como objetivo desenvolver um MVP que permita o monitoramento de culturas agrícolas através da geração automatizada de índices de vegetação por diferença normalizada (NDVI). Para esse fim, serão processadas as bandas de imagens de satélite de alta resolução temporal obtidas a partir de uma região de interesse informada pelo usuário.

1.3 OBJETIVOS ESPECÍFICOS

- Obter e processar imagens de satélite;
- Estudar e integrar diferentes sistemas distribuídos;
- Estudar e aplicar a arquitetura cliente-servidor;
- Desenvolver um sistema que seja acessível através da Web;
- Desenvolver um MVP que possa ser validado pelo usuário final;
- Desenvolver uma interface gráfica que permita ao usuário cadastrar sua plantação e visualizar o índice de vegetação por diferença normalizada através de um mapa interativo;
- Garantir que o sistema seja responsivo, ou seja, que sua interface gráfica seja utilizável tanto em computadores quanto em dispositivos móveis;
- Aprimorar os conhecimentos relacionados a engenharia de software;

1.4 ESTRUTURA DO TRABALHO

O presente trabalho é dividido em 5 capítulos, estando organizado conforme descrito a seguir. No capítulo 2, uma revisão da literatura é realizada, onde as tecnologias empregadas são apresentadas e seu uso justificado. No capítulo 3, a solução proposta é

descrita, abordando detalhadamente aspectos relacionados a arquitetura e funcionalidades da ferramenta desenvolvida. O capítulo 4 aborda os aspectos funcionais alcançados pela ferramenta, detalhando suas funcionalidades e utilização. Por fim, o capítulo 5 realiza as considerações finais a cerca do trabalho e discorre sobre propostas futuras para melhoria e ampliação da plataforma.

2 REVISÃO DE LITERATURA

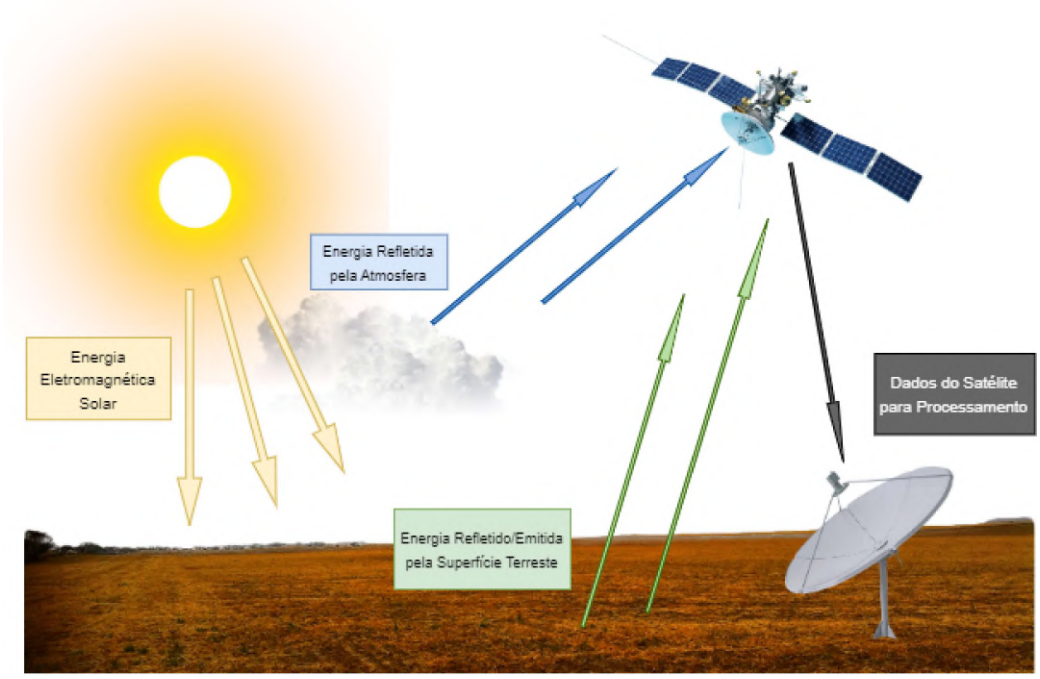
Nesse capítulo, é feita uma revisão da literatura consultada e das ferramentas tecnológicas empregadas durante o desenvolvimento do trabalho.

2.1 SENSORIAMENTO REMOTO

O sensoriamento remoto (SR) é a tecnologia empregada na obtenção de informações de um objeto, área ou fenômeno localizado no planeta Terra, sem que haja para isso a necessidade de contato físico. O uso dessa técnica advém da década de 60, onde as primeiras medições eram feitas com o uso de câmeras acopladas em aeronaves, pipas, foguetes e até pássaros (SHIRATSUCHI *et al.*, 2014). Atualmente, seu uso é instrumentalizado e preferencialmente realizado por satélites, aviões ou veículos aéreos não tripulados (VANTs).

As informações obtidas pelos sensores são registradas em diferentes bandas espectrais captadas através da energia eletromagnética que é refletida ou emitida pela superfície terrestre, conforme ilustrado na Figura 1. As bandas espectrais oriundas desse processo representam o intervalo entre dois comprimentos de onda no espectro eletromagnético e normalmente seus nomes fazem referência a região do espectro onde ela se localiza (vermelho, azul, infravermelho próximo) (QUARTAROLI; VICENTE; ARAUJO, L. S., 2014). Para o caso de sensores imageadores - sensores que geram como produto final uma imagem -, uma imagem do alvo de medição é gerada para cada uma das bandas contempladas pela sua resolução espectral.

Figura 1 – Processo de Sensoreamento Remoto

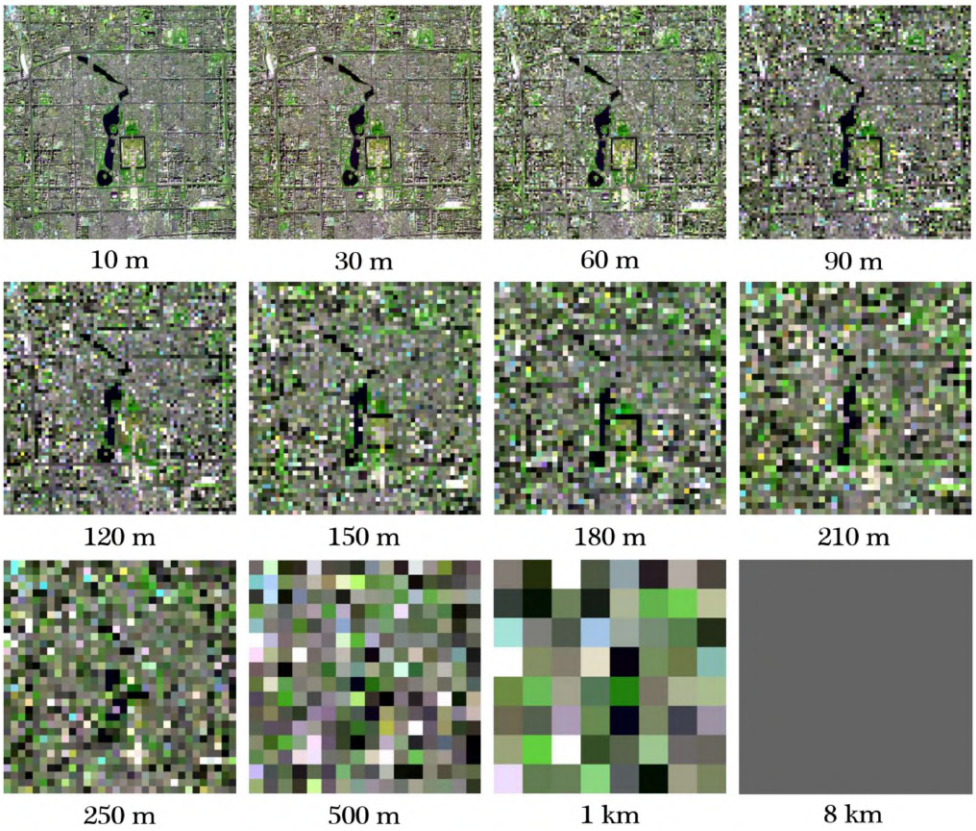


Fonte: O Autor

Diversos programas espacias disponibilizam imagens obtidas da superfície terrestre por meio de diferentes satélites. Cada um desses satélite possui sensores e órbitas distintas, fazendo com que para sua melhor aplicação sejam avaliados os critérios de: resolução temporal, resolução espectral e resolução espacial (BLOSFELD, 2014). A seguir, cada um desses critérios é brevemente explicado.

- Resolução Temporal: a resolução temporal, ou também chamada de taxa de revisita, está associada ao tempo necessário para que o satélite revisite uma região de interesse e capture novas imagens ou adquira novos dados, ou seja, é o tempo exigido para se obter duas imagens consecutivas de uma mesma região;
- Resolução Espectral: a resolução espectral está relacionada aos comprimentos de onda que os sensores espectroradiômetros do satélite são capazes de registrar e está associada ao número de bandas que os sensores do satélite são capazes de discretizar;
- Resolução Espacial: a resolução espacial, ou também chamada de geométrica, se refere ao tamanho do menor objeto que pode ser identificado em uma imagem obtida, conforme exemplifica a Figura 2.

Figura 2 – Diferentes Resoluções Espaciais



Fonte: (JIAQI *et al.*, 2020)

Entre as possíveis aplicações dos dados obtidos por sensoriamento remoto, seu uso na agricultura se destaca por permitir, através de dados de refletância espectral,

obter índices de vegetação (IVs) para inferir questões como: produtividade da cultura, níveis de nutrientes da vegetação e identificação de deficit hídrico. Dessa forma, os dados obtidos através de SR podem ser processados via softwares, resultando em informações fundamentais para guiar a estratégia de um empreendimento agrícola.

2.2 GOOGLE EARTH ENGINE

O Google Earth Engine (GEE) é uma plataforma baseada em nuvem para análise e processamento de dados geoespaciais. Seu imenso catálogo possui terabytes de imagens de satélite e dados de sensoreamento remoto, os quais são utilizados pela comunidade científica para análises espaço-temporais a nível planetário.

O GEE possui APIs desenvolvidas em Python e Javascript, possibilitando que o poder de processamento da infraestrutura da Google seja utilizado por outras aplicações para adquirir e processar dados de sensoriamento remoto. O catálogo oferecido pela plataforma conta com imagens de satélite gratuitas de alta resolução, além de também disponibilizar uma vasta quantidade de dados climáticos já processados. Abaixo, as bases de dados oferecidas pelo GEE são apresentadas:

- **Landsat**: projeto realizado conjuntamente pela USGS e NASA com o objetivo de observar e armazenar imagens de satélite do planeta terra. Sua base dados possui imagens de toda a superfície terrestre com resolução espacial de 30m, disponibilizando dados datados de 1972 até os dias atuais. A resolução temporal dessas imagens é de em média 14 dias.
- **Sentinel**: conjunto de dados composto pelos satélites da série *Sentinel* do projeto *The Copernicus Program*. Seus dados contemplam imagens de radar captadas pelo Sentinel-1A e 1-B, imagens ópticas de alta resolução do Sentinel 2A e 2B, além de dados terrestres e oceânicos obtidos pelo Sentinel 3.
- **MODIS**: os sensores Moderate Resolution Imaging Spectroradiometer (MODIS) estão presentes nos satélites Terra e Aqua da NASA, e obtêm dados da superfície terrestre desde 1999. Suas orbitas são muito superiores aos satélites da série Sentinel e Landsat, possuindo uma resolução espacial de 250m. Entretanto, sua elevada orbita os permite disponibilizar imagens com uma resolução temporal de apenas 15 minutos.
- **Imagens de Alta Resolução**: imagens multiespectrais de alta resolução obtidas a partir dos satélites espaciais da série *SkySat* e do projeto NAIP (*National Agriculture Imagery Program*). As imagens do dataset da SkySat possuem resolução espacial de até 0.8m, enquanto as do projeto NAIP atingem a resolução de 1m. Entretanto, as imagens do projeto NAIP abrangem somente determinadas regiões agrícolas da américa do norte.

- **Dados Geofísicos**: a base de dados do GEE disponibiliza uma série de dados geofísicos, entre eles é possível citar: imagens SRTM empregadas na geração de curvas de nível, índices de vegetação (NDVI), modelos tridimensionais (MDS, MDE) e mapa de ocupação de solo.
- **Dados Climáticos**: são disponibilizados dados oriundos de análises metereolócias da atmosfera terrestre, como: temperatura da superfície e oceânica, precipitação, umidade, velocidade do vento, etc.

2.3 ÍNDICE DE VEGETAÇÃO POR DIFERENÇA NORMALIZADA (NDVI)

A agricultura moderna tem se amparado fortemente na tecnologia para promover a produtividade e a otimização de recursos. Dentre tais aparatos tecnológicos, a utilização de índices de vegetação possibilita monitorar a variação de produtividade de uma cultura através da análise do comportamento espectral de sua vegetação (SILVA, 2018).

Os Índices de Vegetação (IV) são caracterizados como modelos matemáticos que operam em conjunto com o sensoriamento remoto para descrever e avaliar a cobertura vegetal de uma determinada área com base na leitura do espectro de onda refletido por sua superfície. Dessa forma, seu uso permite inferir dados relativos ao desenvolvimento da vegetação, sua biomassa, área de cobertura e até mesmo detectar deficiências nutricionais. Em (SILVA, 2018), são apresentados os principais fatores fisiológicos que influenciam diretamente na resposta espectral da vegetação, sendo eles:

- Déficit Hídrico: a restrição de água para a planta causa o fechamento de seus estômatos (estruturas presentes nas plantas que garantem a realização de trocas gasosas), fazendo com que diminua a transpiração vegetal e a absorção de CO2 e, consequentemente, diminua também sua taxa fotossintética.
- Déficit Nutricional: ocorre devido à deficiência de macro e micronutrientes indispensáveis para o desenvolvimento das plantas. Esse déficit nutricional acarreta na morte prematura das folhas e na diminuição da concentração de clorofila, provocando um aumento significativo de sua reflectância[1] no espectro do vermelho.
- Idade da Planta: no processo de maturação fisiológica, a capacidade de realizar fotossíntese tende a aumentar. Após atingir sua completa maturação, sua taxa fotossintética passa a decair. Desse modo, folhas saudáveis apresentam uma maior reflectância no infravermelho próximo que uma folha em idade avançada.

Entre os diversos índices de vegetação aplicados na agricultura, o cálculo de NDVI (Normalized Difference Vegetation Index) se destaca por sua forte correlação com os

[1] É a proporção entre o fluxo de radiação eletromagnética incidente numa superfície e o fluxo que é refletido.

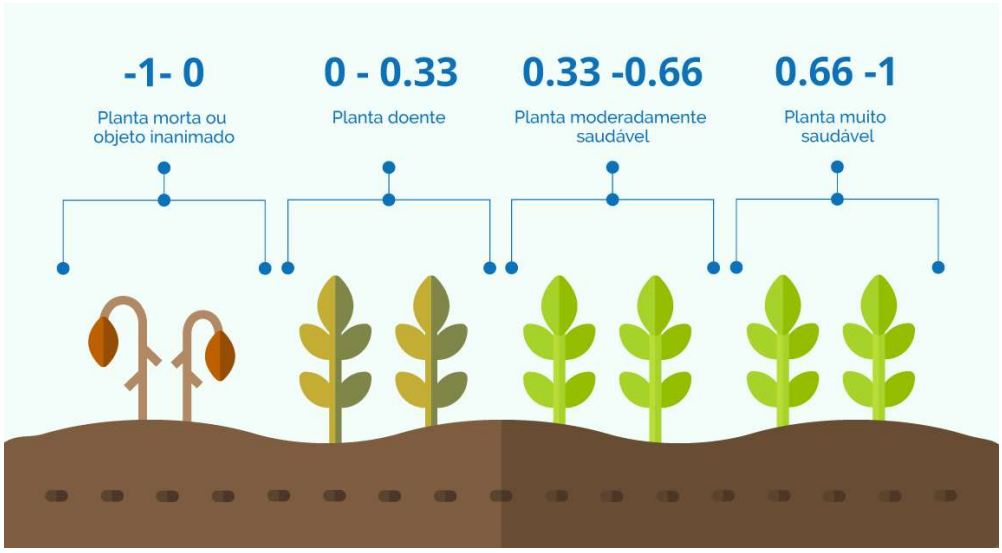
parâmetros biofísicos de uma cultura, permitindo determinar sua densidade de fitomassa foliar fotossinteticamente ativa (BLOSFELD, 2014).

O NDVI é obtido a partir de sensores que operam nas bandas do infravermelho próximo e do vermelho. Com esses dados, a razão entre as respostas espectrais de cada banda é computada para um determinada área utilizando a equação (1), onde: o parâmetro *NIR* representa a reflectância da banda no Infravermelho próximo e *RED* a reflectância da banda Vermelho.

$$NDVI = \frac{NIR - RED}{NIR + RED} \tag{1}$$

Desse modo, o índice de vegetação por diferença normalizada permite gerar um indicador gráfico que possibilita analisar a vegetação de uma área através de imagens que com escalas que podem variar de -1 a 1, onde os valores próximos a 1 são indicativos de vegetação fotossinteticamente ativa e valores próximos ou abaixo de zero relacionados áreas de solo nu, água ou baixa quantidade de vegetação, conforme ilustra a Figura 3.

Figura 3 – Relação do índice de vegetação por diferença normalizada e a saúde de uma planta



Fonte: (SYSTEM, 2019)

Na Figura 4 é apresentado o NDVI computado para um talhão - unidade mínima de cultiva dentro de um propriedade agrícola -, onde as regiões com baixa taxa fotossintética são representadas por cores mais avermelhadas. Nessa Figura, os valores do índice foram normalizados para estarem contidos no intervalo de 0 a 1.

Figura 4 – Índice de vegetação por diferença normalizada para uma região de plantio

Fonte: (PIX4D, 2018)

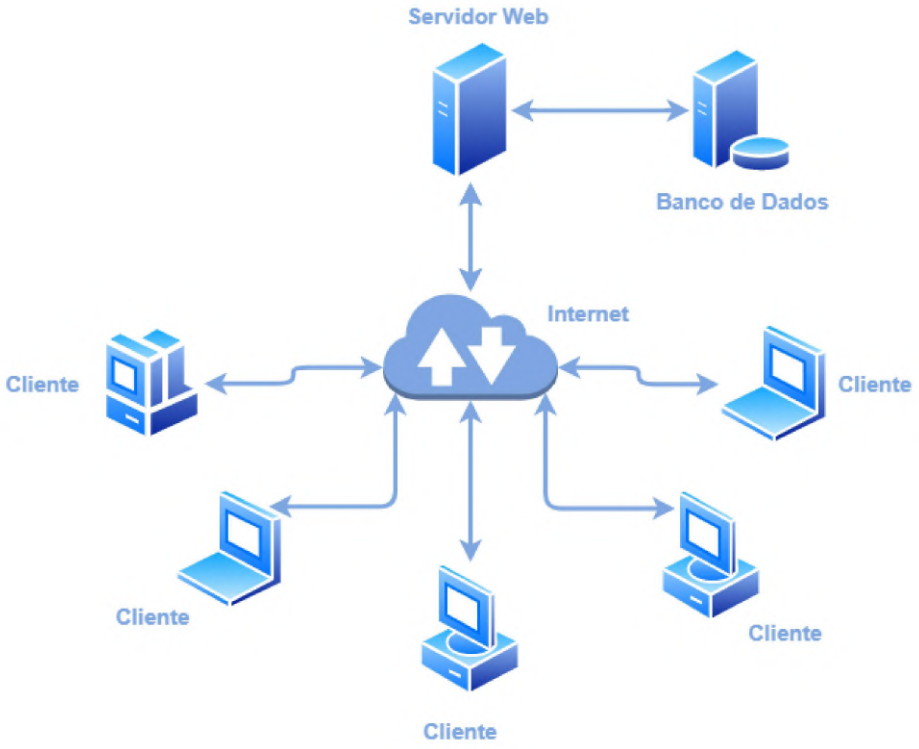
2.4 ARQUITETURA DE SISTEMAS

O processo de arquitetura de software é uma etapa fundamental dentro da construção de um novo produto ou serviço. Nesse momento, são selecionadas as tecnologias que visam garantir que os requisitos funcionais e técnicos da aplicação sejam cumpridos, garantido assim que o resultado final esteja alinhado com as necessidades do usuário.

2.4.1 Arquitetura Cliente-Servidor

A arquitetura cliente/servidor é um modelo computacional onde um computador, denominado servidor, é encarregado de compartilhar recursos com diversos outros computadores, chamados de clientes, que se conectam a ele por meio de uma rede ou através de uma conexão com a internet, conforme a arquitetura apresentada na Figura 5. Sua origem advém de meados de 1980, sendo impulsionada principalmente pelo surgimento dos computadores pessoais (PCs) e por sua maior viabilidade econômica quando comparada aos legados sistemas *mainframe* (DUARTE, 2022).

Figura 5 – Representação da Arquitetura Cliente Servidor



Fonte: O Autor

O servidor se caracteriza como o conjunto de hardware e software capaz de atuar como produtor, hospedando dados e fornecendo seus serviços. Devido ao seu hardware mais robusto, os servidores normalmente são encarregados de realizar a parte computacional mais intensiva dentro dessa arquitetura. Nesse modelo, não existe compartilhamento de recursos por parte do cliente, ou seja, esse atua apenas como um consumidor que solicita conteúdos ou funções de serviço.

O cliente é sempre responsável por iniciar a comunicação com o servidor. O início da comunicação acontece quando o cliente envia através da rede uma solicitação de processo ou recurso ao servidor, que então se responsabiliza por processar e responder com os dados solicitados. A origem da solicitação pode partir desde microcontroladores que realizam requisições para um determinado serviço, quanto de um usuário acessando um aplicativo de celular. Nesse cenário, o servidor normalmente lida com n conexões paralelas de clientes, podendo se comunicar também com outros servidores para atender a uma requisição de um cliente (SULYMAN, 2014).

Dentro do contexto de aplicações web, esse modelo de arquitetura é amplamente empregado, sendo adotado para o servidor o termo *back-end* e para o cliente o termo *front-end*. Nessas aplicações, as chamadas ao servidor são normalmente realizadas para atender a um usuário que interage por meio de uma interface gráfica acessada através de um navegador de internet.

2.4.2 Protocolo HTTP

O HTTP (Hypertext Transfer Protocol) é um protocolo cliente-servidor presente na camada de aplicação que é empregado para a transferência de documentos de hipertexto. É reconhecido como o protocolo base para transferência de informação em toda a Web, sendo responsável por nela definir como a troca de mensagens entre navegadores e servidores deve ocorrer (CHEN; CHENG, 2016).

A comunicação via HTTP é feita no modelo requisição/reposta através de uma conexão estabelecida via protocolo TCP (Transmission Control Protocol), onde o cliente é responsável por submeter uma mensagem de requisição para o servidor. O servidor, que fornece os recursos ou realiza outras funções de interesse do cliente, retorna uma mensagem de resposta para o solicitante. Por se tratar de um protocolo sem estado, cada requisição é feita de forma independente e o servidor as trata como requisições distintas, não guardando nenhuma informação referente ao estado da requisição, mesmo se realizadas por uma mesma conexão TCP.

Para requisitar um recurso ou funcionalidade de um servidor, o cliente informa na estrutura da requisição o método HTTP - também chamado de verbo - que será utilizado e o endereço do recurso que está solicitando. O endereço é normalmente expresso através de uma URL ou um caminho absoluto dentro do servidor. Além dos campos já mencionados, a estrutura da requisição também possui um cabeçalho e um corpo que fornecerão ao servidor todas as informações necessárias para realizar a ação solicitada. Na Tabela 1, os verbos HTTP comumente empregados em aplicações distribuídas são sumarizados conforme apresentado em (ALAM; CARTLEDGE; NELSON, 2014).

Verbo	**Funcionalidade**
GET	Normalmente empregado para a consulta de recursos de um determinado domínio, ou seja, para operações que não modificam ou inserem dados.
POST	Empregado para a criação de novos recursos ou para operações computacionais como cáculos.
PUT	Utilizado em operações de atualização de recursos. Normalmente irá sobreescrever um recurso existente ou criá-lo caso não exista.
DELETE	Deleta um recurso.

Tabela 1 – Métodos HTTP empregados na aplicação.

Dentro do servidor, uma rota - também chamada de *endpoint* - é empregada para associar o método HTTP e a URL à uma função responsável por processar a solicitação e gerar a resposta dessa requisição. A resposta do servidor ao cliente será composta por um código de status, um cabeçalho de resposta e um corpo que, caso necessário, irá conter os dados que serão retornados ao cliente. Para viabilizar índices ainda mais

elevados de interoperabilidades entre as aplicações, as funcionalidades do servidor podem ser disponibilizadas através de uma API (Application Programming Interface), garantindo que outros sistemas consigam programaticamente utilizar suas funcionalidades de forma simples e padronizada.

Para indicar ao cliente o resultado do processamento de sua requisição, os códigos de resposta HTTP são empregadas na comunicação entre as partes. De modo geral, os status da resposta são representados através de um número inteiro de três dígitos, onde o primeiro algarismo representa a qual das cinco possíveis classes de status está associada a resposta, enquanto os dois algarismos seguintes são utilizados para detalhar seu significado. As cinco possíveis classes de resposta são: **1XX** - Informativa, **2XX** - Sucesso na requisição, **3XX** - Redirecionamento, **4XX** - Erro do Cliente e **5XX** - Erro no servidor. Na tabela 2, são apresentados exemplos de códigos e suas respectivas descrições.

Código	**Descrição**
200	A requisição foi bem sucedida.
201	Enviada após uma requisição do POST, indicando que um novo recurso foi criado com sucesso.
400	Um erro de sintaxe foi encontrado na requisição enviada e impediu seu processamento.
401	O cliente deve se autenticar para obter o recurso desejado.
404	O recurso solicitado não foi encontrado. Normalmente retornado para um URL inválida.
500	Um erro interno ocorreu no servidor.

Tabela 2 – Exemplos de Códigos de resposta HTTP empregados por servidores.

2.4.3 Controle de Acesso a Recursos

Para gerenciar o acesso a recursos dentro de uma aplicação e disponibilizar dados de maneira segura, as etapas de autenticação e autorização são amplamente difundidas. A etapa de autenticação acontece quando o cliente que irá acessar a aplicação se identifica através da suas credenciais de acesso, o que em sistemas web é feito normalmente a partir de uma tela de login. Após essa etapa, o cliente recebe um identificador único de autorização - normalmente na forma de um *token* - que o permite utilizar as funcionalidades da aplicação que requerem autorização.

O JSON Web Token (JWT) é um padrão definido pela RFC7519 e empregado em cenários de autenticação para possibilitar armazenar informações de forma segura e compacta no formato de um *token*. O JWT consiste em uma única *string* composta por três partes separadas por pontos e codificadas em base64, sendo elas: cabeçalho do *token*, conteúdo do *token* e assinatura. Embora normalmente não seja criptografado e facilmente decodificado, sua assinatura permite validar a autenticidade dos dados recebidos e garantir

que o seu conteúdo não tenha sofrido adulteração. Desse modo, o *token* codificado é gerado e assinado pelo servidor que o envia ao cliente após sua autenticação. Ao solicitar um recurso que tenha controle de acesso, o cliente envia o JWT junto a requisição - normalmente o adicionando ao seu cabeçalho - para que o servidor valide sua identidade (AKANKSHA; CHATURVEDI, 2022).

2.4.4 Persistência de Dados

Os dados são a parte fundamental das aplicações e por isso é preciso garantir que além de acessíveis, sejam também duráveis. A persistência de dados busca garantir que informações sejam armazenadas de modo a serem recuperáveis de forma consistente, fazendo com que, por exemplo, elas não sejam perdidas caso a sessão em uma aplicação seja encerrada (SUN *et al.*, 2010).

Para viabilizar que os dados sejam armazenados em um meio não volátil e assim tornem-se persistentes, manipuláveis e organizados, é fundamental que seja utilizado um Sistema Gerenciador de Banco de Dados (SGBD). Dessa forma, seu uso permite que a aplicação foque nas regras de negócio e transfira a responsabilidade de armazenar os dados para uma aplicação especializada nesse processo.

Embora um SGBD forneça uma interface para a aplicação manipular os dados através de comandos SQL (*Structured Query Language*), realizar a interface com o banco de dados utilizando somente SQL tende a tornar o desenvolvimento pouco produtivo e dificultar o processo de manutenção do código. Para contornar tal limitação, o uso de ferramentas denominadas ORMs (*Object Relational Mapper*) permite atingir altos níveis de produtividade ao possibilitar que o paradigma de orientação a objetos seja utilizado para lidar com bancos de dados.

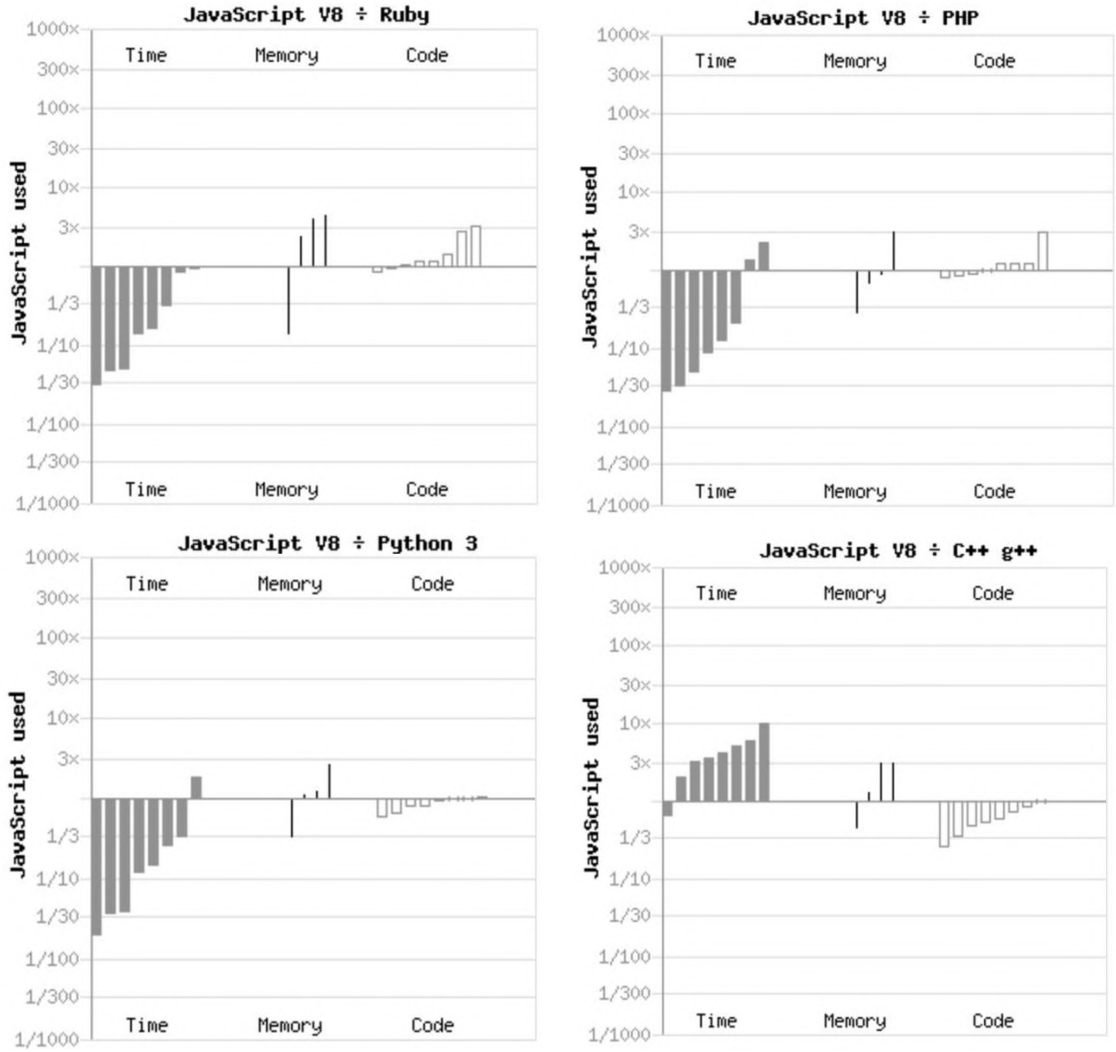
2.4.5 Node.js

O Node.js é uma plataforma de código aberto de natureza assíncrona, orientada a eventos e voltada ao desenvolvimento de aplicações escaláveis. Sua arquitetura é baseada no interpretador V8 da Google que possibilita a execução de códigos JavaScript fora de um navegador web (AGUIAR, 2015),(PULUCENO, 2012), (OLIVEIRA JÚNIOR, 2017). Atualmente, essa tecnologia de desenvolvimento conta com aproximadamente 4 milhões de adeptos e apresenta uma taxa anual de cem por cento de crescimento, se destacando com umas das tecnologias mais populares para o desenvolvimento escalável de aplicações web, móveis, microsserviços e internet das coisas (OLIVEIRA JÚNIOR, 2017).

A *engine* v8 é um interpretador Javascript de código aberto escrito em C++ que possibilita ao Node.js alcançar o desempenho de um código binário compilado. Sua presença na arquitetura do Node.js o faz ter resultados tão performáticos quanto uma linguagem de baixo nível como o C++, ao mesmo passo que entrega ao desenvolvedor a possibilidade de codificar em Javascript, uma linguagem de alto nível e extremamente dinâmica.

Em (AGUIAR, 2015), um estudo é feito para comparar a performance do Javascript V8 com as linguagens PHP, Python 3, Ruby — linguagens comumente empregadas no desenvolvimento Web - e com o próprio C++. Nesse estudo, as métricas avaliadas foram: tempo de execução, memória utilizada e verbosidade (quantidade de palavras necessárias para expressar a intenção do código). Na Figura 6 são apresentados os resultados do estudo, onde as linhas, quando abaixo do eixo principal do gráfico representam um desempenho superior do JavaScript em relação à tecnologia concorrente.

Figura 6 – (a) V8 vs Ruby (b) V8 vs PHP (c) V8 vs Python 3 (d) V8 vs C++

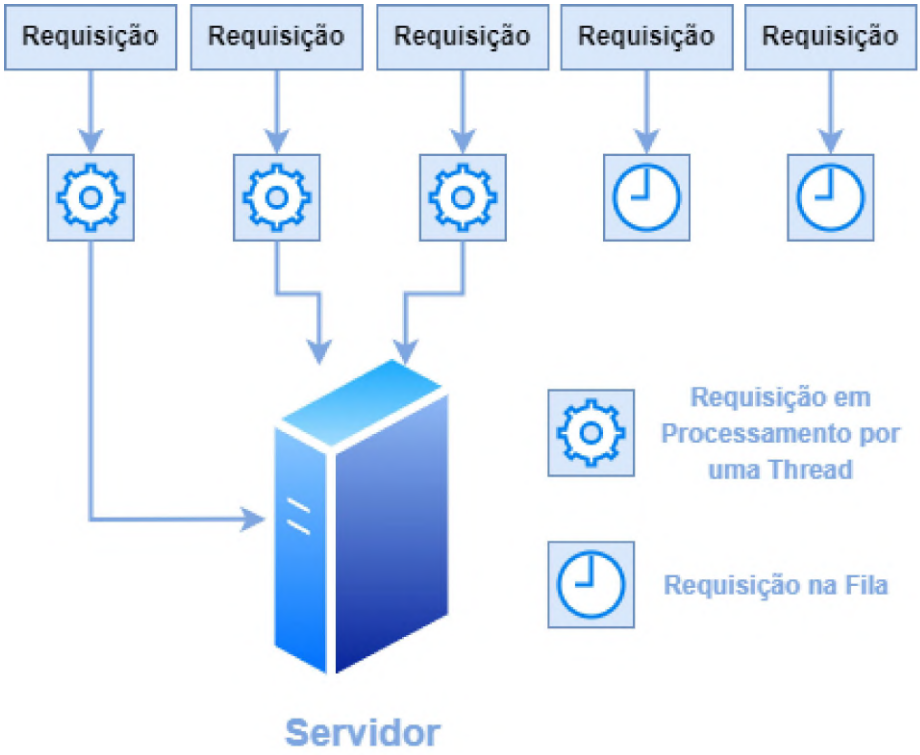


Fonte: (AGUIAR, 2015)

O maior diferencial do Node.js está relacionado a sua natureza *single-thread* orientada a eventos, o que permite a ele suportar rotinas concorrentes sem a necessidade de múltiplas *threads*. Em servidores *multi-threads* convencionais, a cada requisição recebida é criada uma nova *thread* para processá-la, fazendo com que mais recursos computacionais sejam alocados no processo (TILKOV; VINOSKI, 2010). Como os recursos de um servidor são limitados, esse processo de criação não é capaz de seguir infinitamente, fazendo

com que caso o servidor receba um número muito elevado de conexões, novas requisições tenham que esperar em uma fila, conforme representado pela Figura 7.

Figura 7 – Fila de Requisições em Servidores Multi-threads



Fonte: O Autor

Em um servidor Node.js, as funcionalidades da engine V8 são estendidas para permitir a interação com o sistema operacional de forma assíncrona em uma única *thread* não-bloqueável construída em torno de uma arquitetura orientada a eventos. Isso permite ao Node.js economizar recursos de memória e CPU para cenários de múltiplas conexões paralelas (PULUCENO, 2012),(TILKOV; VINOSKI, 2010). Dessa forma, seu uso é amplamente difundido em sistemas que precisam lidar com inúmeras operações de entrada e saída simultâneas.

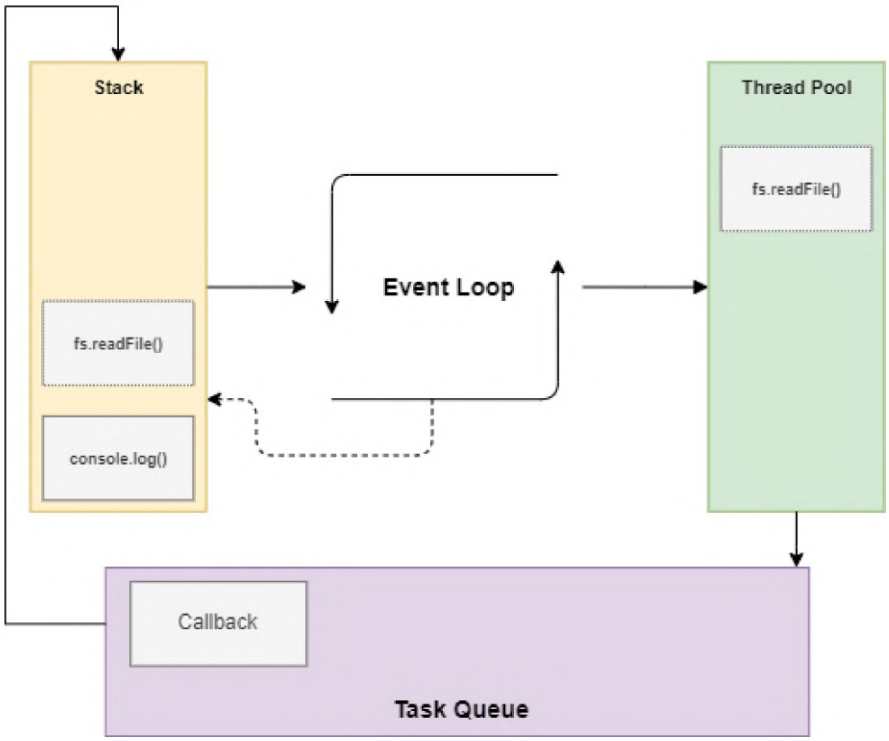
2.4.5.1 Event Loop

Os servidores, em sua maioria, não necessitam lidar com operações computacionais de extrema complexidade, mas sim realizar constantes e lentas operações de leitura e escrita em disco ou envio e processamento de dados através da rede. Esses processos são denominados operações de Entrada & Saída (E/S) (STACK, s.d.).

As características single-thread e orientada a eventos do Javascript possibilitam ao Node.js lidar com operações assíncronas de E/S através de *higher-order function* - funções que recebem outras funções, aqui denominadas de *callbacks*, como argumento - especificando o que deve ser realizado na ocorrência de um evento. Embora o Node seja

single-thread, sua API C++ responsável pela comunicação com o sistema operacional não é, o que o permite realizar o processamento concorrente de operações de E/S que, quando finalizados, chamem a *callback* registrada para aquele evento. Dessa forma, essas *callbacks* podem enfileirar outras tarefas que por sua vez podem enfileirar outras e assim por diante, tudo isso dentro de um contexto denominado *Task Queue*. Desse modo, essas operações passam a ser tratadas de forma não bloqueante através de uma única thread responsável por lidar com as *callbacks* dentro de um laço de repetição infinito, que está implicitamente presente no código, chamado de *Event Loop* (TILKOV; VINOSKI, 2010), (STACK, s.d.). Na Figura 8, uma representação simplificada desse tratamento é apresentada.

Figura 8 – Event Loop



Fonte: O Autor

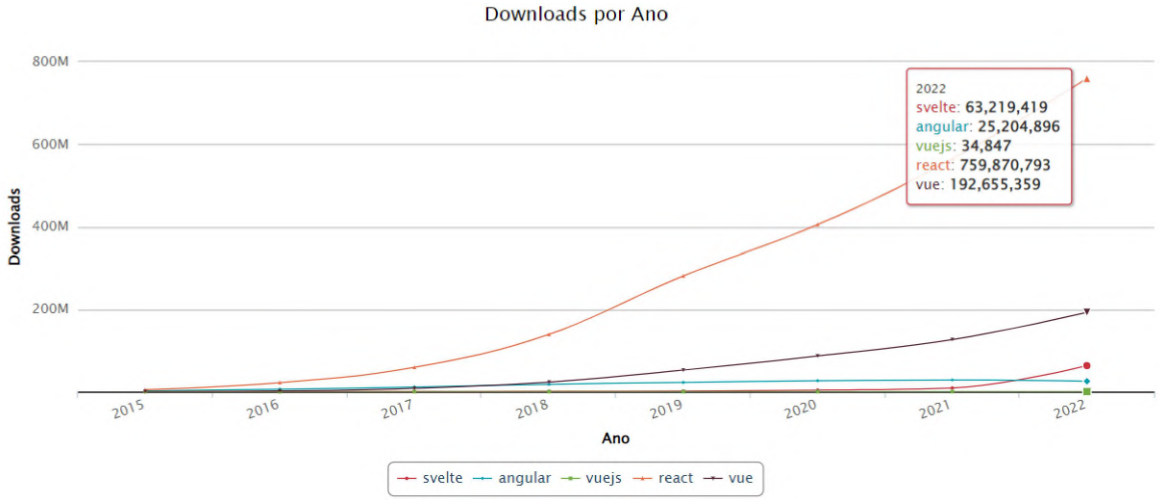
2.4.6 React

O React é um framework Javascript de código aberto desenvolvido pelo time de engenheiros da Meta (empresa proprietária do Facebook, Instagram e WhatsApp) com a proposta de facilitar a construção de interfaces gráficas complexas através do desenvolvimento de componentes reutilizáveis. Uma aplicação desenvolvida com esse framework é categorizada como uma SPA (Single-Page Application), ou seja, uma aplicação que realiza o carregamento de seus recursos de uma única vez, fazendo com que todo seus *scripts* estejam disponíveis ao cliente em seu primeiro acesso. Dessa forma, ao navegar pela aplicação a experiência do usuário passa a ser similar à de uma aplicação *desktop* convencional.

Durante o seu uso, a aplicação se comunica com o servidor apenas para solicitar recursos ou criar novos recursos em reposta a ações realizadas pelo usuário (DESHMUKH; MANE; RETAWADE, 2019). Dentro do padrão de arquitetura MVC - *Model View Controller* - ele está posicionado na camada *View*.

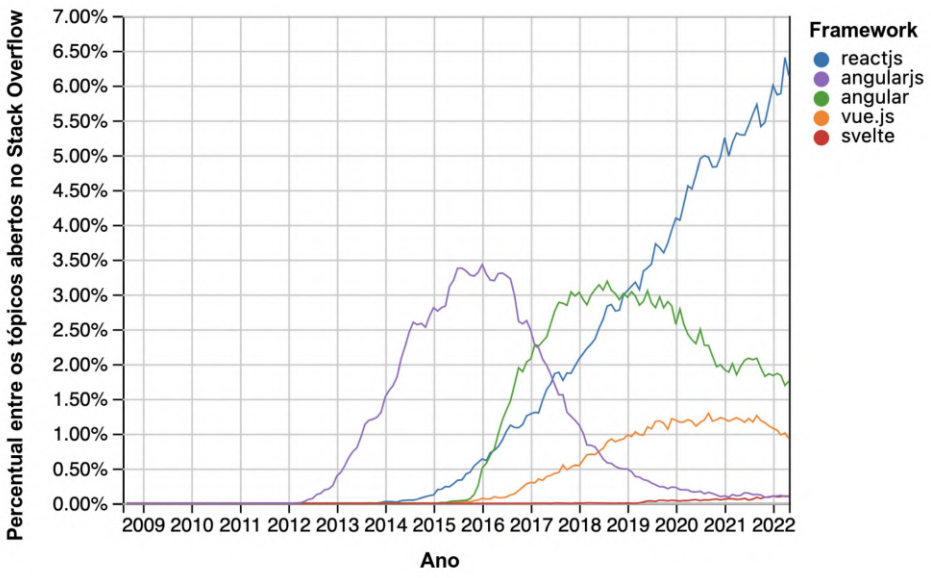
Atualmente, é um dos frameworks mais populares entre a comunidade de desenvolvedores, o que é comprovado ao observarmos as Figuras 9 e 10 que comparam, respectivamente, a quantidade de downloads para diferentes frameworks *front-end* e a quantidade de tópicos abertos dentro da plataforma Stack Overflow para tais tecnologias.

Figura 9 – Análise de downloads por ano de diferentes frameworks front-end



Fonte: Adaptado de (NPM-STAT, s.d.)

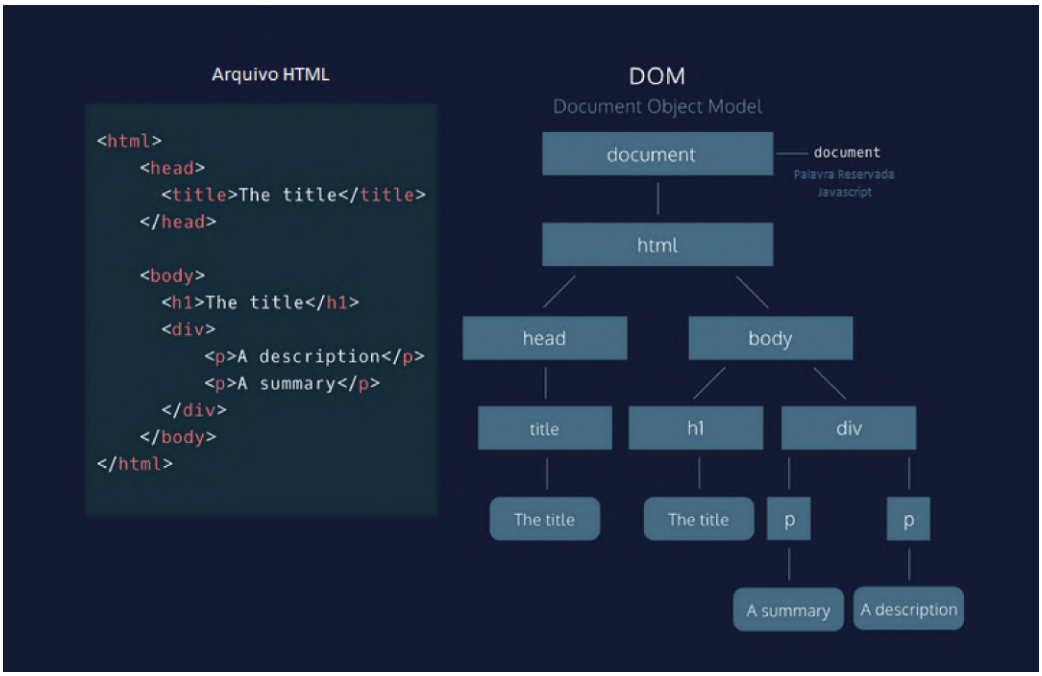
Figura 10 – Análise de popularidade de diferentes tecnologias front-end na plataforma Stack Overflow



Fonte: Adaptado de (STACKOVERFLOW, s.d.)

O grande diferencial do React em termos de performance está intimamente ligado a forma como ele manipula documentos HTML (*HyperText Markup Language*). Um documento HTML é considerado o bloco mais básico da web, sendo responsável por definir o significado e a estrutura de todo o conteúdo nela presente. Nesse cenário, os navegadores utilizam o conceito de DOM (Documento Object Model), uma representação orientada a objetos do HTML de uma página web e que tem como função permitir a manipulabilidade de um documento HTML - ou seja, da estrutura de uma página da web - através do uso de uma linguagem de script (MOZILLA.ORG, 2022). A Figura 11 ilustra um documento HTML e a sua representação no DOM.

Figura 11 – Representação de um HTML no DOM



Fonte: Adaptado de (VELOG.IO, 2021)

Para possibilitar a manipulação mais ágil de páginas, o React cria um DOM virtual em memória, realizando as alterações desejadas e atualizando o navegador de forma mais otimizada. A virtualização do DOM permite ao React aplicar estratégias para decidir quando uma parte da interface gráfica deve ou não ser re-renderizado, evitando renderizações não desejadas e aumentando expressivamente a performance das aplicações. Essa decisão é tomada com base nos *states* e *props*, parâmetros presentes na arquitetura React que, além de carregarem dados, podem determinar quando um componente deve ou não ser re-renderizado. Para toda mudança em um desses parâmetros, o React DOM compara os novos valores aos anteriormente armazenados e, caso exista uma diferença entres eles, realiza uma re-renderização desse componente (JAVEED, 2019).

Na Figura 12 é apresentado um exemplo de aplicação React composta por um componente denominado **CarrinhoCompras** que possui como filho um componente cha-

mado **ProdutoDesejado**. Nesse exemplo, é possível perceber a presença das *props* e dos *states* mencionados anteriormente. Os *states* são declarados utilizando a função *useState*, importada diretamente da biblioteca React, que retorna um vetor que é desestruturado para se obter uma constante que representa o valor atual daquele *state* e uma função empregada para atualizar seu valor. Dessa forma, o botão presente no componente **ProdutoDesejado** dispara um evento que é responsável por alterar o *state* **quantidade** através de uma chamada a função **setQuantidade**, fazendo com que o componente seja renderizado para apresentar seu valor atualizado.

Figura 12 – Exemplo de aplicação React.

```
import React, { useState } from 'react';
import ReactDOM from 'react-dom/client';

function ProdutoDesejado(props) {

    const [quantidade, setQuantidade] = useState(1)

    return (
        <div className="Produto">
            <h2> Sou um Produto {props.name} adicionado ao carrinho de compras!</h2>
            <h3> Quantidade desejada: {quantidade} </h3>
            <button onClick={() => setQuantidade(quantidade + 1)}> Adicionar + 1 </button>
        </div>
    );
}

function CarrinhoCompras() {

    let nomeProduto = "qualquer"

    return (
        <div className="Carrinho">
            <h1>Sou o Carrinho de Compras</h1>
            <ProdutoDesejado name={nomeProduto} />
        </div>
    );
}

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(<CarrinhoCompras />);
```


Fonte: O Autor

2.5 TRABALHOS CORRELATOS

A geração e análise de índice de vegetação por diferença normalizada é amplamente difundida e aplicada na otimização do potencial agrícola de uma fazenda. Portanto, diversos trabalhos utilizam desse índice para avaliar ou propor novas técnicas de cultivo.

Em (RISSINI; KAWAKAMI; GENÚ, 2015), por exemplo, a aplicação do NDVI como ferramenta para análise do potencial produtivo em culturas de trigo com adubação nitrogenada se apresentou muito eficiente. Os resultados obtidos apontaram que as leituras de NDVI obtidas apresentaram correlação com as doses de nitrogênio aplicadas e seu respectivo estádio fenológico - fase de desenvolvimento vegetal em que a planta se encontra - possibilitando a elaboração modelos capazes de descrever a produtividade da lavoura desde os seu estágio inicial de desenvolvimento. Ainda voltada a aplicabilidade de NDVI em cultivos de trigo com adubação nitrogenada, o trabalho apresentado em (REZNICK,

2017) reforça a aplicabilidade desse índice de vegetação no monitoramento em tempo real da produtividade de grãos e da qualidade tecnológica do trigo em reposta as variações na adubação com nitrogênio, evidenciando dessa forma uma correlação entre o NDVI e o teor de proteína e glúten obtidos na farinha, o que possibilita gerar também modelos capazes de estimar tais parâmetros.

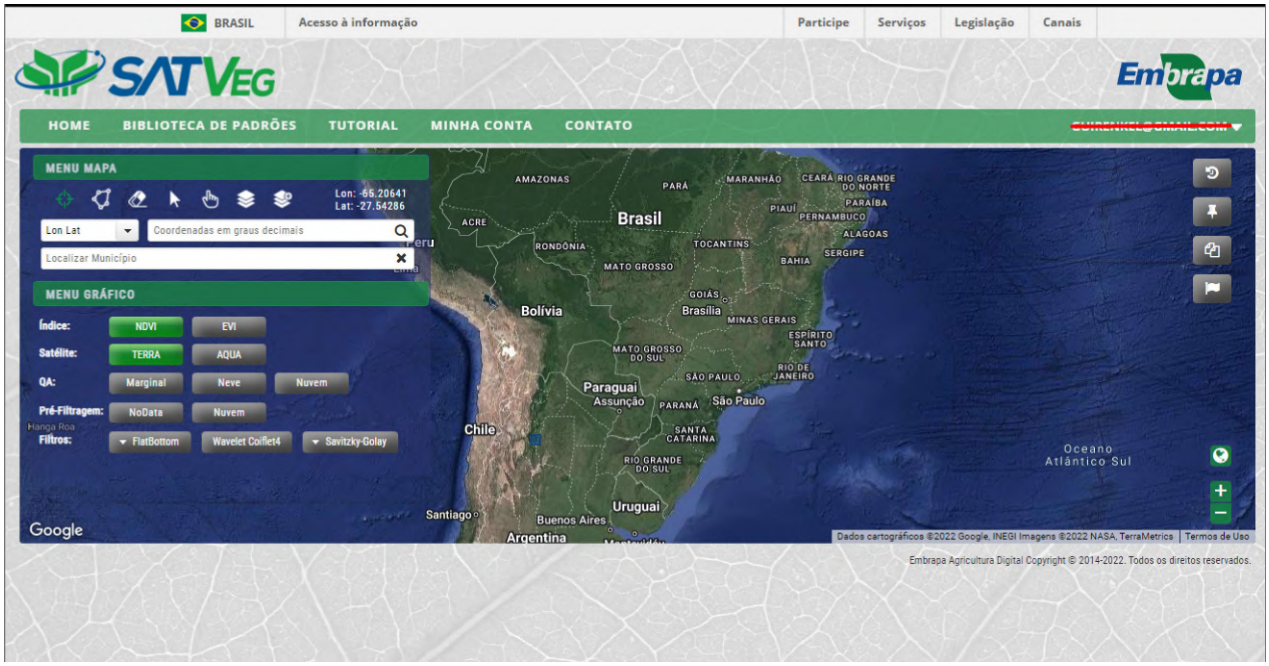
Utilizando também de índices de vegetação obtidos por sensoreamento remoto como uma alternativa para os onerosos e pouco precisos procedimentos de previsão de safras de milho realizados em campo, em (BERTOLIN *et al.*, 2017), é evidenciado que o NDVI foi o único índice a apresentar boa correlação com a produtividade da cultura, permitindo a geração de um modelo matemática exequível em tempo quase que real e capaz de prever os resultados de safras futuras com uma subestimativa de apenas 6,32% quando comparado aos valores da produtividade observada.

O uso de NDVI pode ser ainda estendido para outras aplicações no âmbito da agricultura, sendo integrada, por exemplo, à sistemas mecatrônicos que atuam *in loco*. Em (RIZK; HABIB, 2018), um sistema para monitoramento em campo da saúde de culturas é desenvolvido e embarcado em um robô construído para se locomover por caminhos estabelecidos dentro de uma lavoura, obtendo imagens das plantas, suas localizações e processando seus respectivos índices de vegetação por diferença normalizada para estimar a saúde de suas folhas. O sistema foi testado em uma estufa de tomates para categorizar as plantas em: pouco desenvolvida, saudável ou doente, alcançando uma precisão de 83%.

No mercado atual, ferramentas que oferecerem análises temporais de NDVI são encontradas em versões pagas e gratuitas. Os produtos pagos se destacam por oferecer, integrados ao seu sistema, formas de gerenciamento da fazenda, além de maior precisão através de imagens de satélite de maior resolução. Entre as plataformas de acesso e uso gratuito, a SATVeg (Sistema de Análise Temporal da Vegetação) se destaca.

Construída pela empresa pública Embrapa, a SATVeg tem como propósito permitir o acesso e visualização de perfis temporais dos índices vegetativos através do sensor MODIS presente nos satélites Terra e Aqua. Entretanto, suas funcionalidades possuem limitações, uma vez que a sua baixa resolução temporal permite à plataforma entregar novos dados em um tempo médio de 16 dias e a resolução espaciais de suas imagens é de apenas 250m, prejudicando tomadas de decisões que busquem agilidade e assertividade. Na Figura 13, uma imagem do sistema é apresentada, onde é possível observar uma região de mapa utilizada para interação com o usuário e um menu para seleção de parâmetros referentes a geração dos índices de vegetação.

Figura 13 – Plataforma SATVeg



Fonte: O Autor

3 SOLUÇÃO PROPOSTA

Neste capítulo é apresentada a solução proposta, contemplando os requisitos funcionais da aplicação desenvolvida, sua arquitetura e o emprego das tecnologias supracitadas, além da sua distribuição em ambiente de produção.

3.1 REQUISITOS FUNCIONAIS

A solução proposta é um MVP (*Minimum Viable Product*) voltado à análise de culturas através da geração de NDVI para talhões dentro de uma propriedade agrícola. Seu público alvo são agricultores que buscam realizar o acompanhamento de suas culturas de forma prática, eficaz e econômica. Desse modo, a aplicação desenvolvida busca permitir que seus usuários possam cadastrar suas propriedades e culturas através de uma interface gráfica intuitiva. Para atingir esses objetivos, uma série de requisitos funcionais foram mapeados e são apresentados na Tabela 3.

Requisito	**Descrição**
Acessível através da Web	A aplicação deverá ser acessível através de qualquer dispositivo conectado à internet.
Autenticação de Usuário	O usuário do sistema deve ser capaz de criar uma conta e acessar a aplicação com seus dados cadastrais. Cada usuário deve ser único e seus dados acessíveis somente por ele.
Cadastro de Propriedade Utilizando um Mapa	O usuário autenticado poderá cadastrar sua propriedade utilizando uma interface integrada a um mapa que viabilize o georreferenciamento de suas lavouras.
Listagem de Propriedades e Visualização em Mapa	Todas as propriedades cadastradas pelo usuário devem ser visualizáveis através de um mapa.
Cadastro de Talhão e Visualização em Mapa	Para cada uma de suas propriedades, o usuário poderá cadastrar n talhões referentes a cada uma de suas culturas. No cadastro, o usuário deve interagir com o mapa gerando um polígono que represente a área de interesse.
Geração de NDVI para períodos personalizáveis	Para cada uma de suas culturas, o usuário deve ser capaz de gerar o NDVI para o período informado e visualizá-lo no mapa.

Tabela 3 – Requisitos Funcionais do Sistema

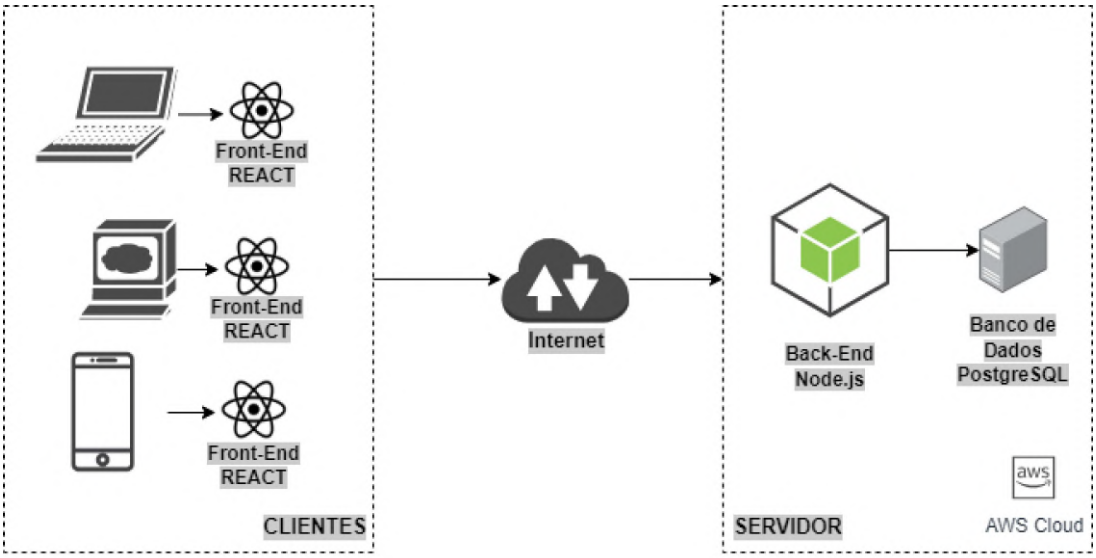
3.2 ARQUITETURA DA APLICAÇÃO

A fim de atingir os requisitos funcionais descritos acima, a arquitetura da aplicação foi desenvolvida para permitir a conexão paralela de múltiplos usuários conectados pela

internet. O modelo Cliente-Servidor foi empregado para gerenciar as solicitações de recursos por parte dos clientes e tornar a aplicação escalável, separando a arquitetura em *front-end* e *back-end*.

A Figura 14 apresenta a arquitetura proposta, onde é possível visualizar o uso das tecnologias React e Node.js empregadas, respectivamente, no *front-end* e *back-end*. Essas tecnologias foram escolhidas estrategicamente para permitir que o fluxo de dados entre cliente e servidor fosse inteiramente estruturado com o formato JSON (*JavaScript Object Notation*), tornando o processo de desenvolvimento e manutenção muito mais ágil.

Figura 14 – Arquitetura Cliente Servidor da Aplicação



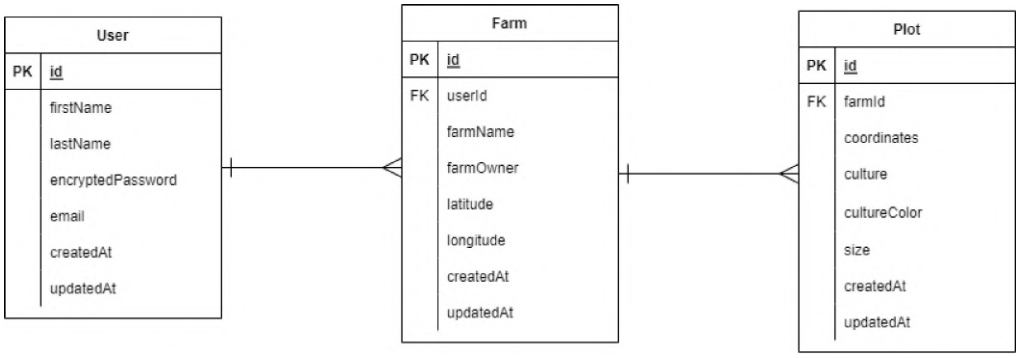
Fonte: O Autor

O Back-End da aplicação foi desenvolvido utilizando a plataforma de código aberto Node.js, onde o *framework* Express foi utilizado para implementar o sistema de *endpoints* da API, ou seja, para lidar com os verbos HTTP em diferentes URLs. Para cada uma dessas rotas está associada uma função que pode executar operações de autenticação de usuário, consulta ou persistência de dados.

Para o armazenamento de informações foi utilizado o sistema de gerenciamento de banco de dados relacional PostgreSQL, que se destaca por ser de código aberto e extremamente popular na comunidade. Em conjunto a ele, foi empregado o ORM Sequelize para realizar as transações e relacionamentos das entidades do banco com Javascript.

Na Figura 15 é apresentado o Diagrama Entidade Relacionamento (DER) do banco implementado na aplicação, onde a entidade *User* possui uma relação **1:N** com a entidade *Farm* através da chave-estrangeira **userId**. A fazenda, por sua vez estabelece um relacionamento **1:N** com a entidade *Plot* através da presença da chave-estrangeira **farmId**. Dessa forma, para cada usuário são armazenadas **N** fazendas, cada uma podendo conter **N** talhões.

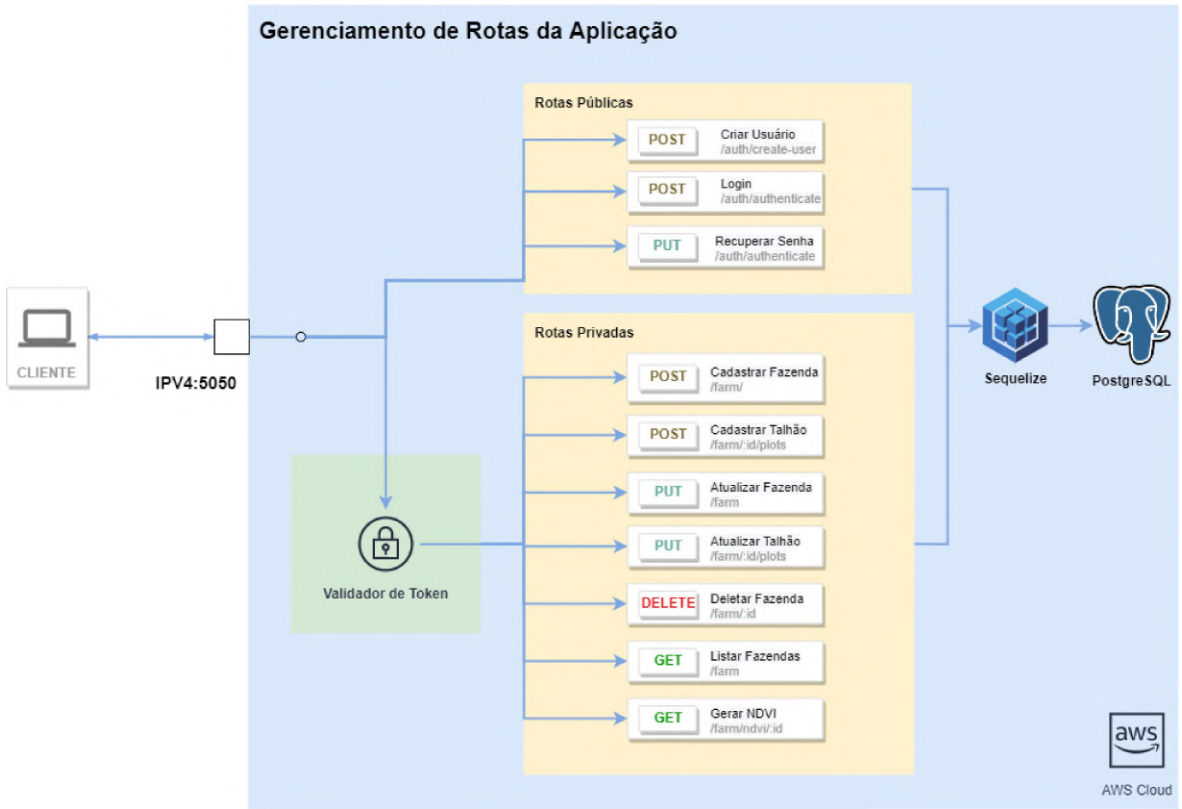
Figura 15 – Diagrama Entidade Relacionamento da Aplicação



Fonte: O Autor

Na Figura 16, uma representação do sistema de rotas da aplicação é apresentado juntamente com as URLS e seus respectivos verbos HTTP. O conjunto de rotas públicas não exige a autenticação dos clientes, uma vez que tais rotas mapeiam recursos relacionados ao cadastro de usuário, autenticação e solicitação de senha. Por sua vez, as rotas denominadas privadas encapsulam funcionalidades restritas, as quais devem ser acessíveis somente via autenticação.

Figura 16 – Rotas da Aplicação



Fonte: O Autor

Para garantir uma comunicação segura entre o cliente e o servidor, o padrão JSON Web Token foi empregado para gerenciar o acesso a serviços e recursos através do controle

de autenticação. Dessa forma, para que rotas privadas sejam acessadas, é indispensável que o usuário esteja autenticado e forneça seu token – que é gerado e retornado pela API no momento em que o usuário faz o login através da interface gráfica – em cada requisição feita à API.

Na construção do *front-end* da aplicação, foi amplamente explorada a capacidade do React de criar componentes reutilizáveis. O serviço do Google Maps foi também incorporado à interface gráfica do sistema como um componente, permitindo que as funcionalidades de cadastro e visualização de fazendas e culturas sejam totalmente interativas, otimizadas e georreferenciadas. Através da integração da aplicação ao Google Maps, o usuário pode realizar o cadastro de sua fazenda informando a sua localização, o que é feito através de um evento Javascript chamado quando o usuário clica em um local do mapa, conforme mostrado na Figura 17. Entre os valores retornados por esse evento estão as coordenadas de latitude e longitude da região apontada pelo usuário, permitindo assim a persistência da localização informada.

Figura 17 – Usuário informando sua localização no mapa.

Fonte: O Autor

Para o cadastro de áreas de cultivo de uma fazenda, a funcionalidade de polígono disponibilizada pela API do Google Maps foi empregada para viabilizar que o usuário possa realizar o contorno dos talhões referentes as suas culturas produzidas. Após finalizado o desenho do polígono, um evento Javascript é chamando, retornando entre seus parâmetros um vetor com pares de latitude e longitude para cada um dos vértices do polígono cadastrado. Os dados de georreferenciamento obtidos da região de cultivo são então persistidos no banco de dados no formato JSON. A Figura 18 apresenta um talhão cadastrado através da interface do Google Maps.

Figura 18 – Área de cultivo cadastrada no mapa.

Fonte: O Autor

3.3 INTEGRAÇÃO COM O EARTH ENGINE

Utilizando as informações georreferenciadas dos talhões, a API Earth Engine do Google, presente no *back-end* do sistema proposto, é aplicada para viabilizar o cálculo do índice de vegetação por diferença normalizada por meio de imagens de satélite disponibilizadas em sua base de dados. O satélite SENTINEL-2, vinculado ao projeto Copernicus da união europeia, fornece imagens do planeta com resolução espacial de 10 metros e recorrência de 5 dias terrestres. Desse modo, sua aplicação permite que novas análises de NDVI sejam fornecidas pelo sistema em um intervalo também de 5 dias.

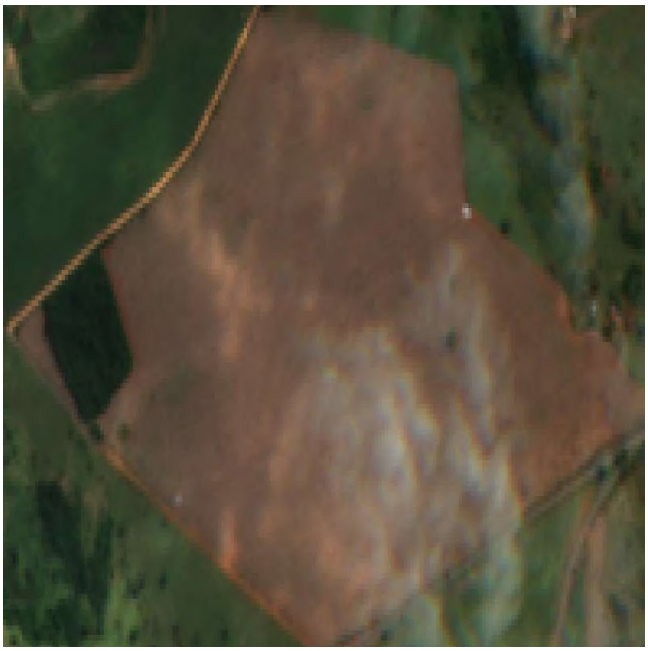
O Earth Engine API possibilita que sejam aplicados aos dados do satélite SENTINEL-2 filtros de data, região de interesse, porcentagem de sobreposição de nuvens, etc. O que torna possível realizar a busca por dados de satélite que contenham a região de interesse, aqui representada pelo polígono cadastrado pelo usuário, e que tenham sido tiradas dentro do intervalo de tempo desejado para a análise.

Nas Figuras 19 e 20, são apresentadas duas imagens obtidas para uma plantação dentro de um intervalo de 7 dias, onde é possível perceber a presença de cobertura de nuvens. A nuvens são artefatos indesejados por causarem distorções nos índices de vegetação calculados e exigem que uma etapa de processamento adicional seja feita para remover dados que possuam mais de 30% de sua área por elas coberta.

Figura 19 – Imagem de Satélite da Região de Interesse Livre de Nuvens

Fonte: O Autor

Figura 20 – Imagem de Satélite da Região de Interesse Com Presença de Nuvens



Fonte: O Autor

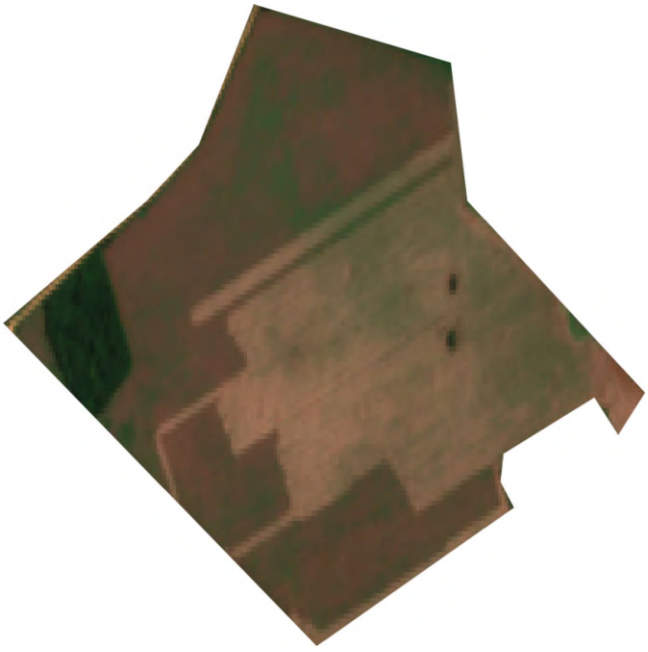
As imagens válidas para uma dada região de interesse são consultadas através de métodos disponibilizados pela API do Earth Engine e o conjunto de resultados obtidos ordenados com base em sua porcentagem de sobreposição de nuvens. A imagem com menor porcentagem de sobreposição de nuvem é então selecionada para extração da região de interesse, onde a área do talhão que será processado é recortada. Para essa região, o NDVI

é computado por meio das bandas do vermelho e do infravermelho próximo nela presente. Posteriormente, o NDVI resultante é retornado ao *front-end* da aplicação para que seja mostrado ao usuário através do mapa presente na interface gráficas. As Figuras 21, 22 e 23, apresentam as etapas de processamento.

Figura 21 – Imagem RGB de satélite contendo a região de interesse

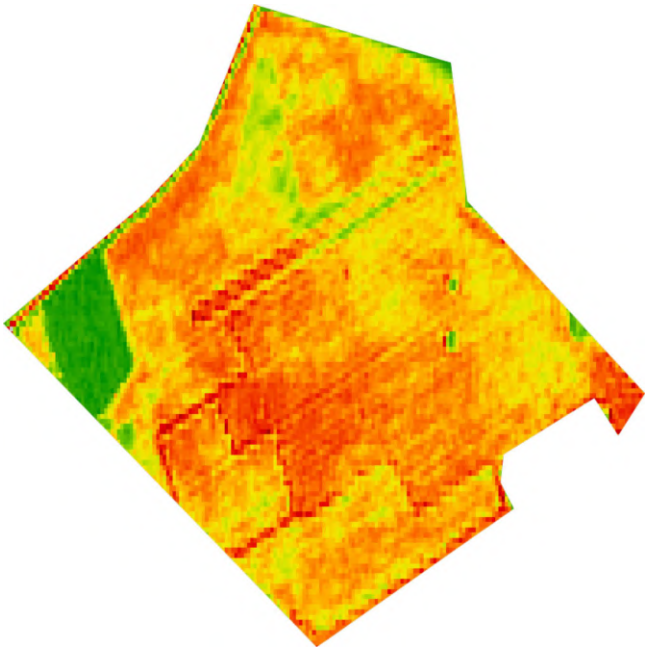
Fonte: O Autor

Figura 22 – Região de Interesse.



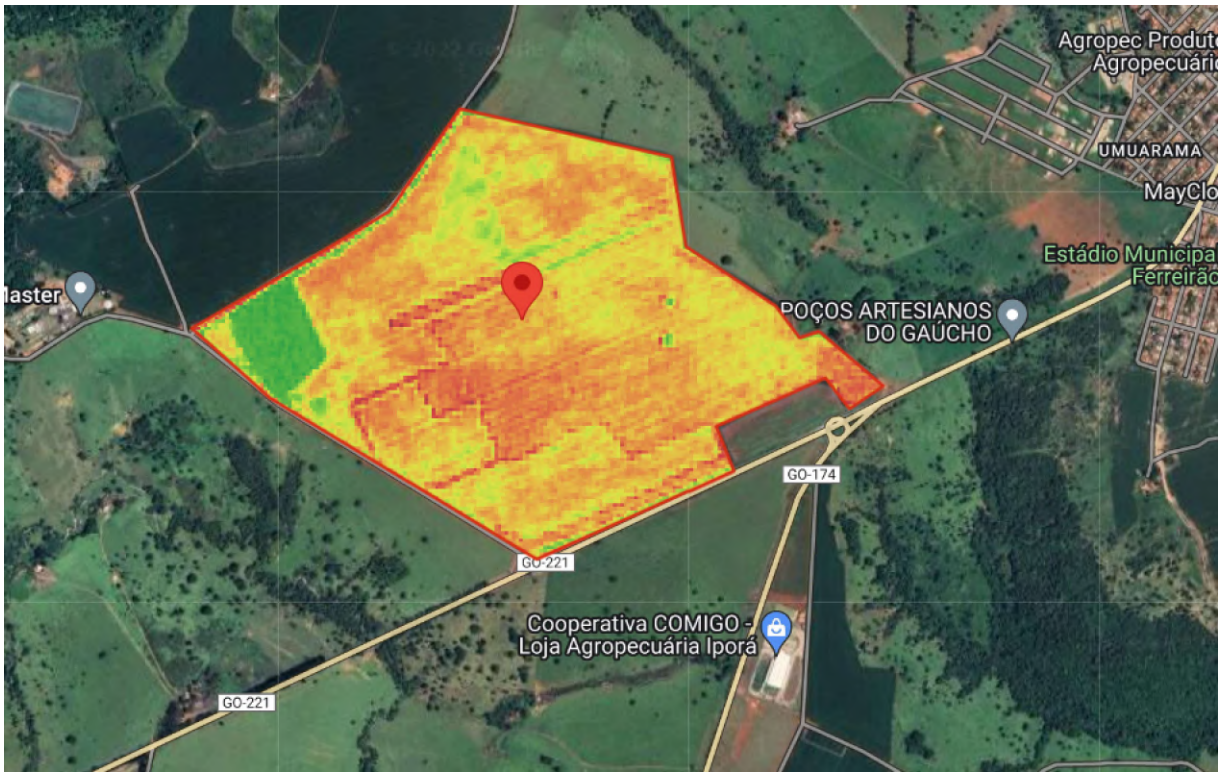
Fonte: O Autor

Figura 23 – NDVI da Região de Interesse



Fonte: O Autor

Figura 24 – NDVI na interface gráfica



Fonte: O Autor

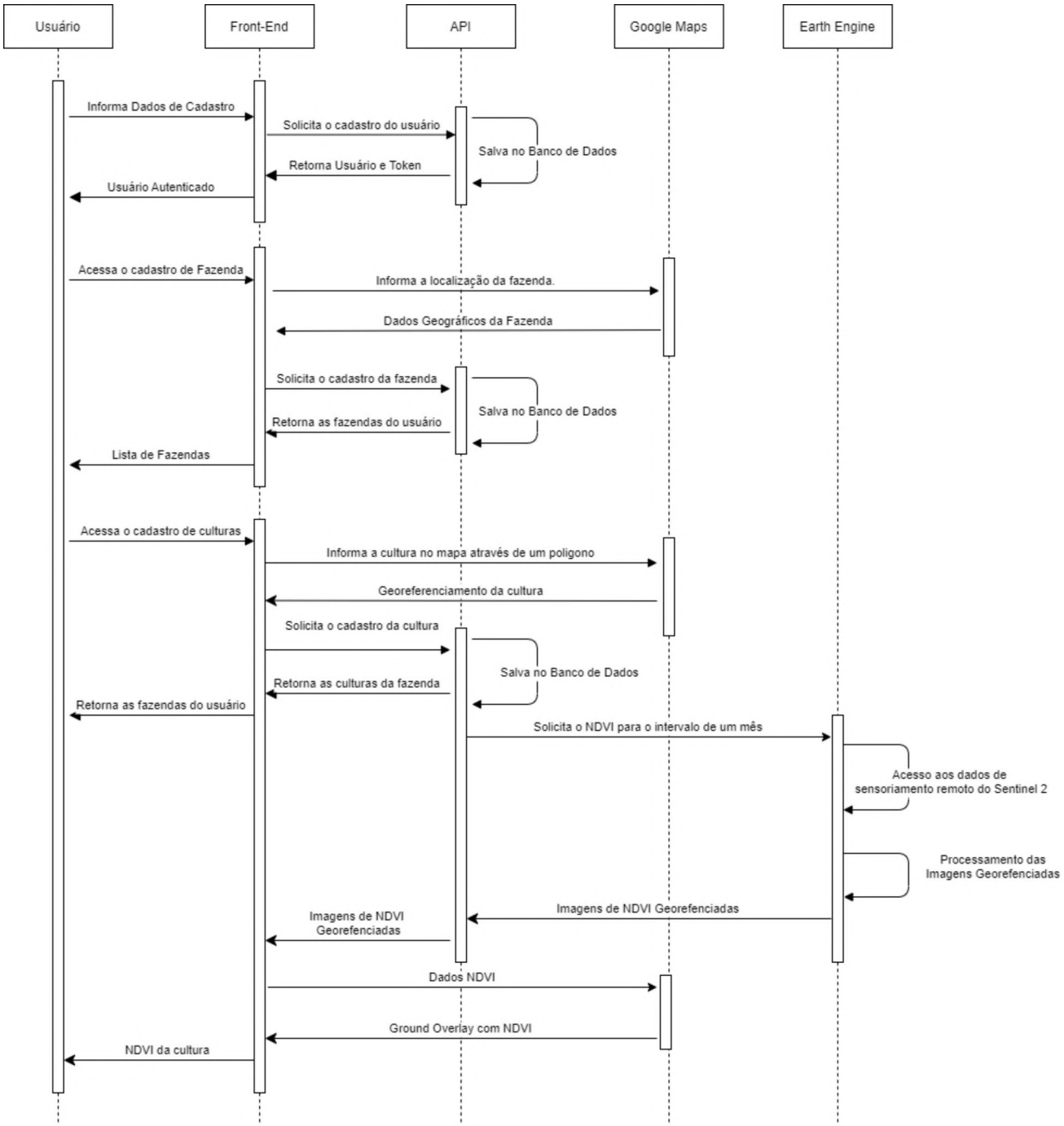
3.4 DISTRIBUIÇÃO EM AMBIENTE DE PRODUÇÃO

Para possibilitar que o sistema seja acessível através de um domínio público, a plataforma AWS (Amazon Web Services) de computação em nuvem da Amazon foi utilizada para instanciar os servidores da aplicação. Dentre os diversos serviços disponibilizados pela AWS, o Amazon Elastic Compute Cloud - EC2 foi empregado, possibilitando a alocação de capacidade computacional na nuvem de forma segura e redimensionável.

Utilizando o protocolo SSH, foi estabelecido um canal de comunicação com uma instância do EC2, onde um servidor Linux foi configurado para disponibilizar a aplicação. Dentro do servidor, o *back-end* foi configurado para atendar à requisições na porta 5050 e rodar como um serviço Linux. Já o *front-end* da ferramenta foi disponibilizado através de um servidor NGINX configurado para responder à requisições na porta 3000 dessa mesma instância do EC2. Embora seja aconselhável desacoplar o *front* e o *back* em servidores distintos para escalar horizontalmente o sistema, ambos foram disponibilizadas na mesma instância do EC2 para, inicialmente, reduzir custos de operação.

Ao final das configurações, a aplicação e todas suas funcionalidades são acessíveis na web através de uma URL pública. Na Figura 25 é apresentado o diagrama de sequência final da plataforma proposta, onde o ciclo de vida de interação das diferentes partes do sistema é descrito.

Figura 25 – Diagrama Sequêncial do Sistema.



Fonte: O Autor

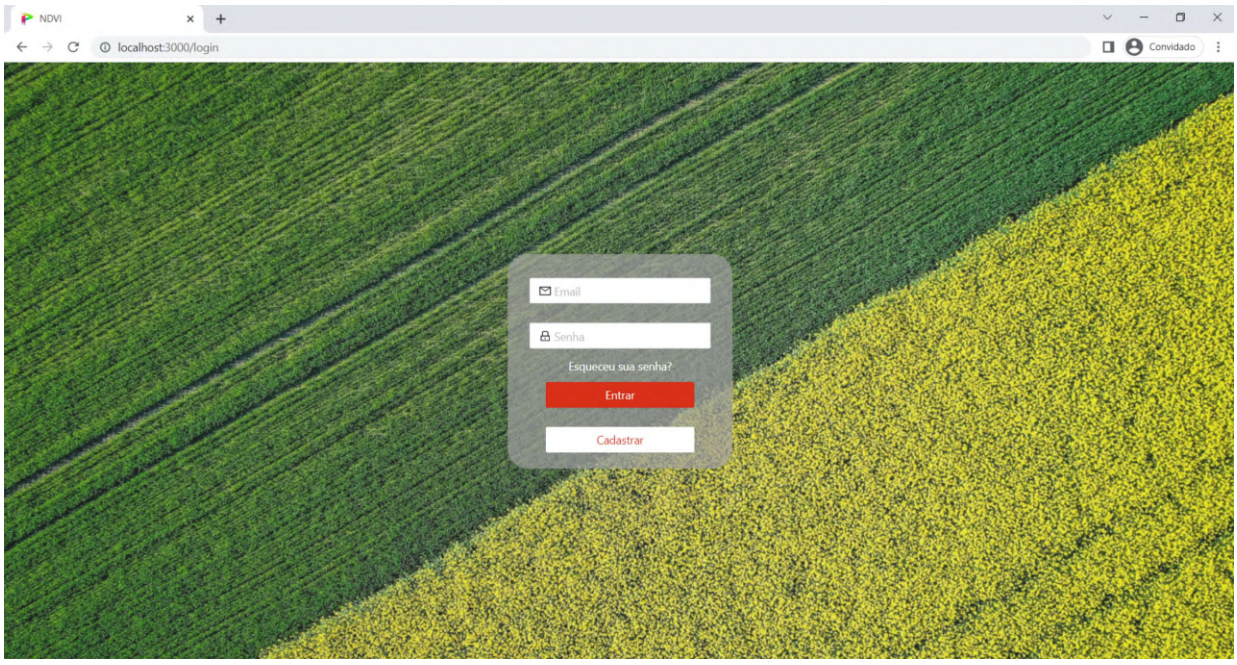
4 RESULTADOS

Nesse capítulo, o MVP desenvolvido para análise e monitoramento de NDVI é apresentado. As funcionalidades e o fluxo de utilização do sistema são demonstrados e discutidos.

4.1 LOGIN E CADASTRO DE USUÁRIO

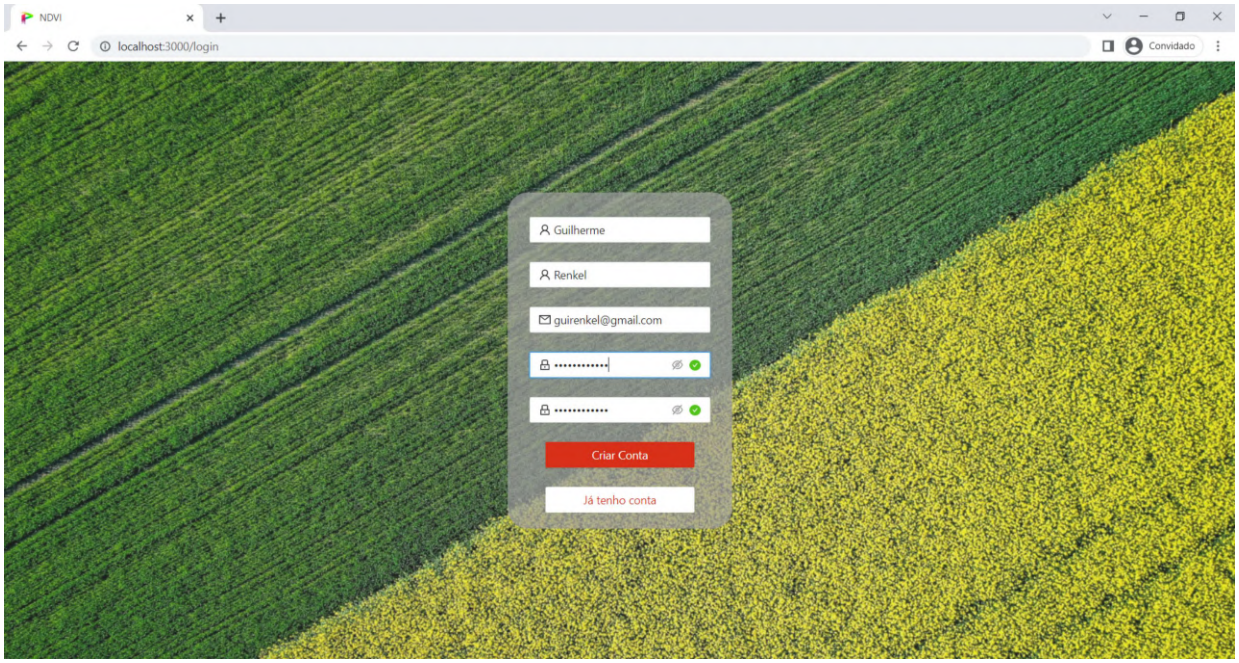
Acessando o sistema a partir de sua URL pública, o usuário é direcionado para a tela de Login, onde poderá criar uma nova conta ou utilizar uma já existente. Ao se autenticar, o token do usuário que é fornecido pela API é salvo na memória do navegador para ser utilizado nas requisições ao servidor. Nas Figuras 26 e 27 são apresentadas, respectivamente, as telas de login e criação de usuários.

Figura 26 – Tela de Login



Fonte: O Autor

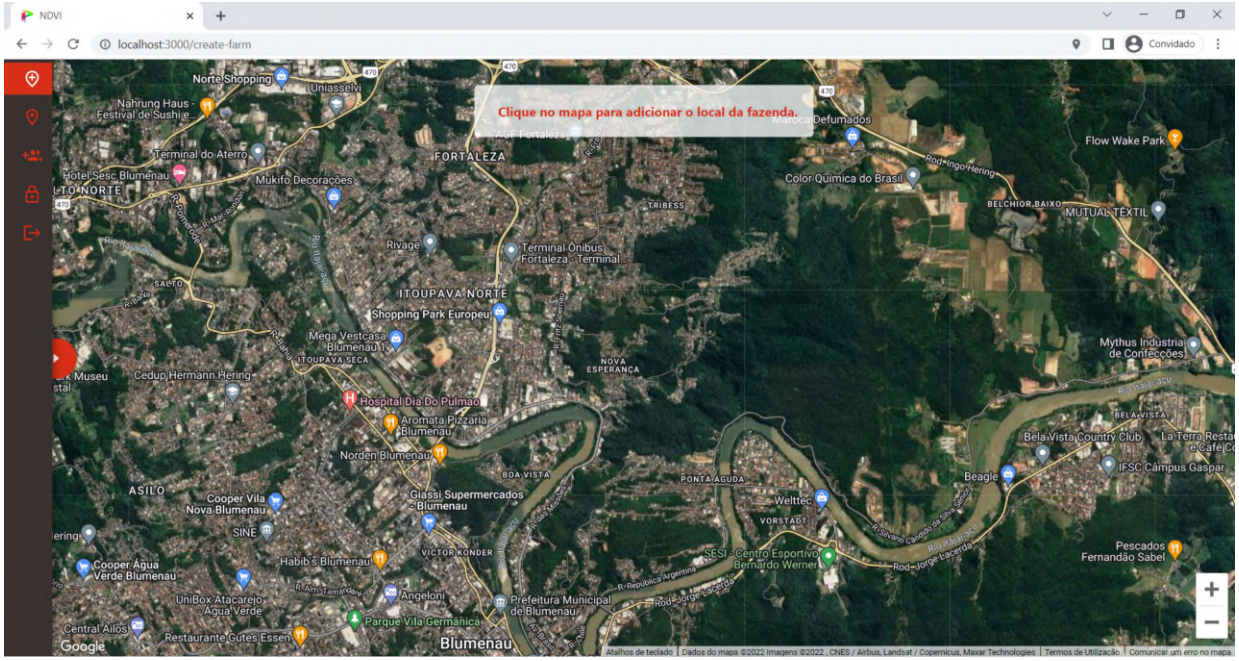
Figura 27 – Tela de Criação de Usuário



Fonte: O Autor

Uma vez autenticado, o usuário obtém acesso à tela inicial do sistema. A Figura 28 apresenta a estrutura interna da aplicação, onde se destaca a presença de um mapa interativo e um menu lateral para navegação. Ao abrir essa tela, o sistema utiliza a geolocalização do navegador para centralizar o mapa em torno da localização atual do usuário.

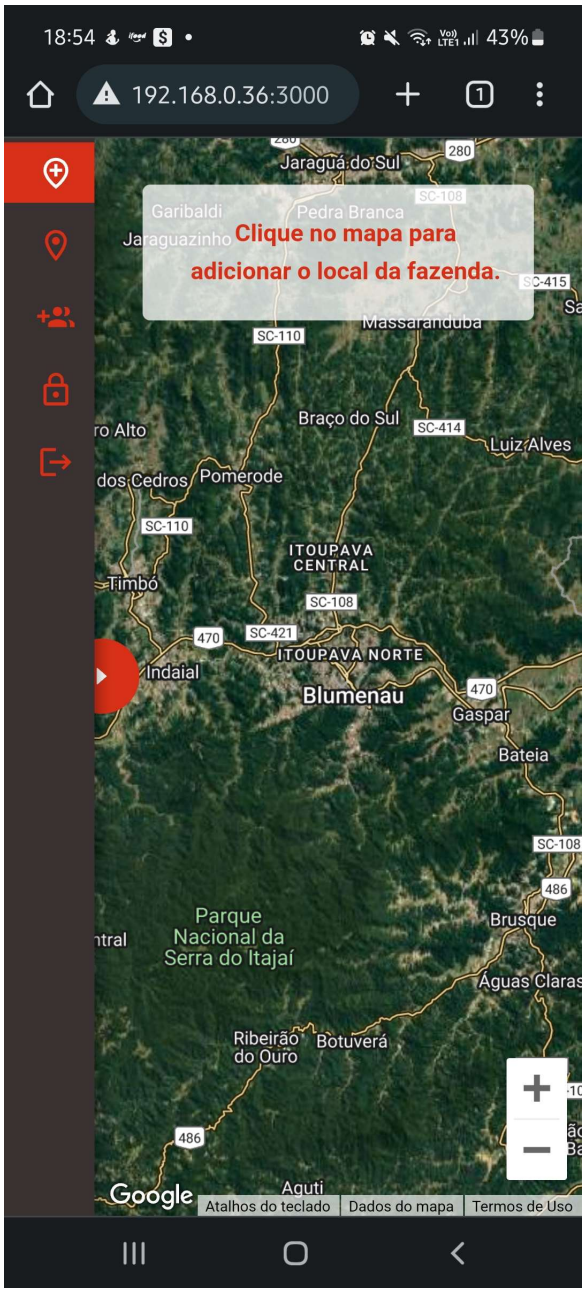
Figura 28 – Tela Inicial do Sistema



Fonte: O Autor

A interface do sistema foi construída de modo a ser responsiva a diferentes dispositivos, o que permite que a aplicação seja também facilmente utilizada em celulares e tablets. Na Figura 29 é possível observar a tela em um dispositivo móvel.

Figura 29 – Sistema acessado a partir de um dispositivo móvel



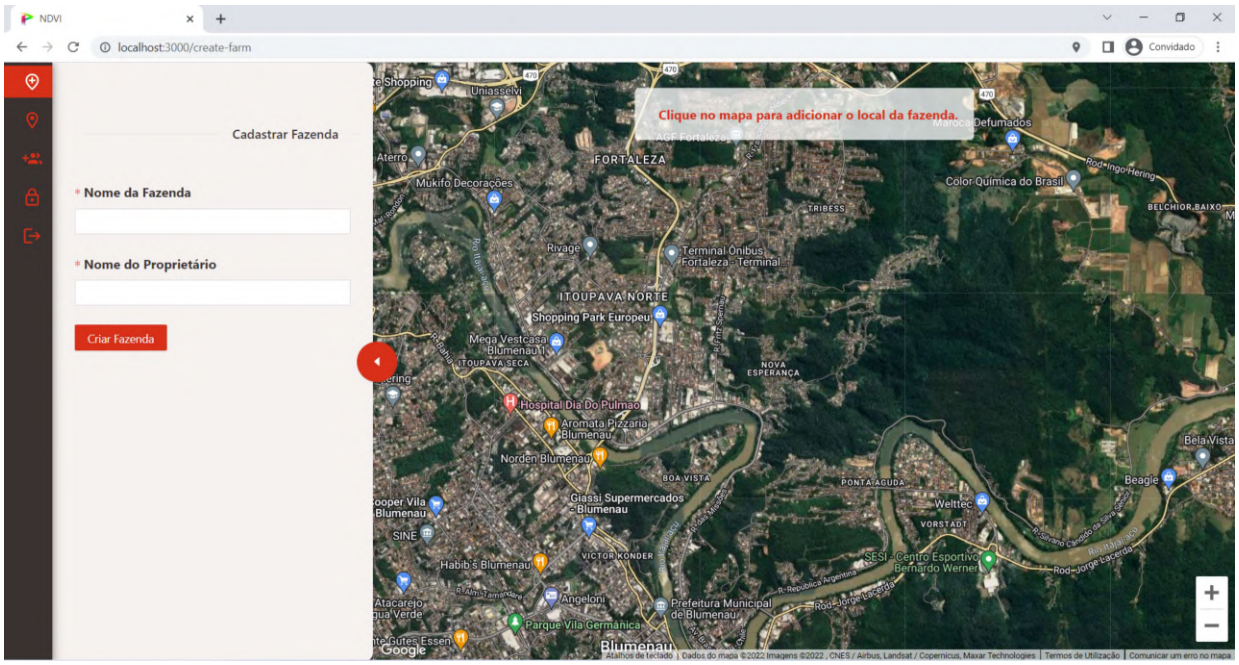
Fonte: O Autor

4.2 CADASTRO DE FAZENDAS

O cadastro de fazendas é realizado a partir da opção do menu **Cadastrar Fazenda**, onde é apresentado ao usuário um formulário que contém os campos **Nome da Fazenda** e **Nome do Proprietário**, conforme demostrado na Figura 30. Após inserir as informações

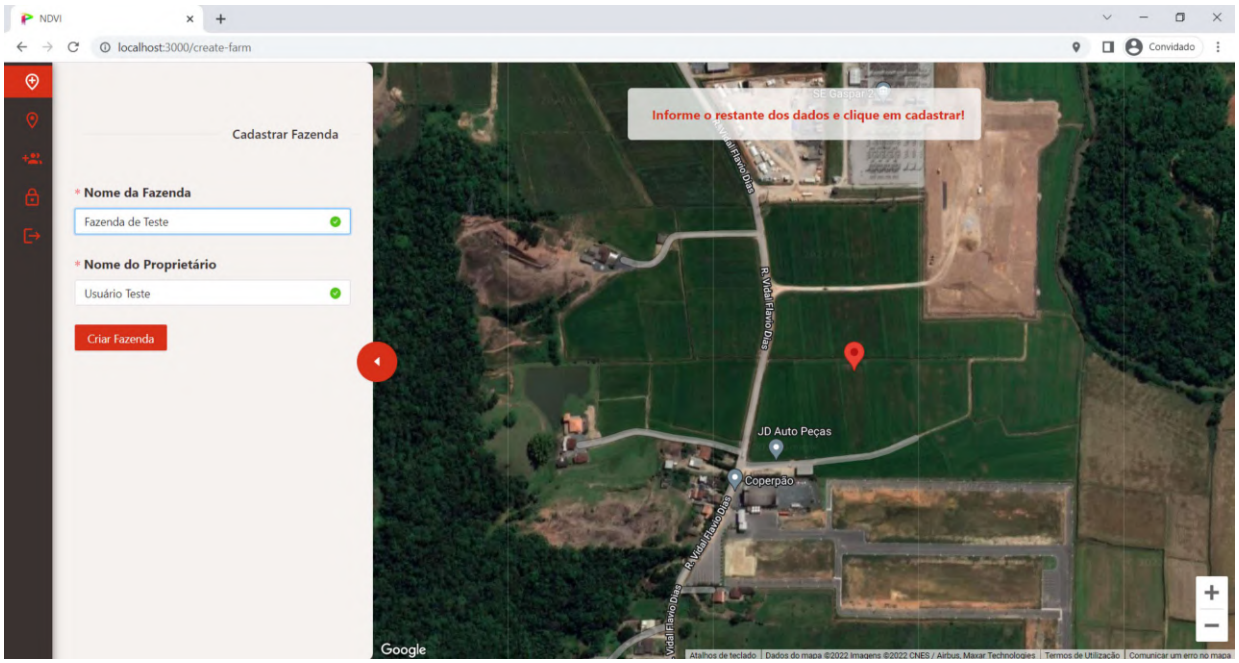
básicas de sua fazenda, o usuário deve também informar no mapa a localização da sua propriedade. Na Figura 31, é apresentado um exemplo de cadastro pré-finalizado.

Figura 30 – Cadastro de Fazenda



Fonte: O Autor

Figura 31 – Local da fazenda informado pelo usuário



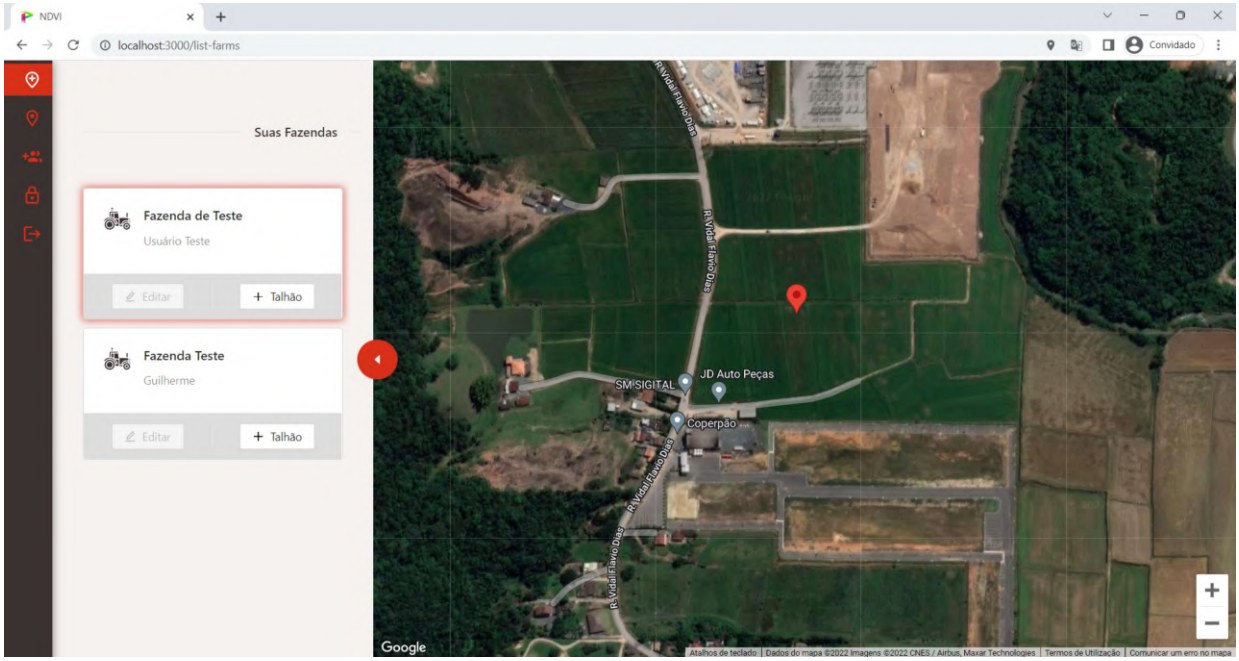
Fonte: O Autor

4.3 LISTAGEM DE FAZENDAS E CADASTRO DE TALHÕES

Finalizado o cadastro de fazendas, o usuário pode acessar uma listagem de suas fazendas cadastradas através da opção **Minhas Fazendas** presente no menu. Nessa tela, suas fazendas são listadas, possibilitando a visualização no mapa e o cadastro de plantações, conforme demostrado na Figura 32.

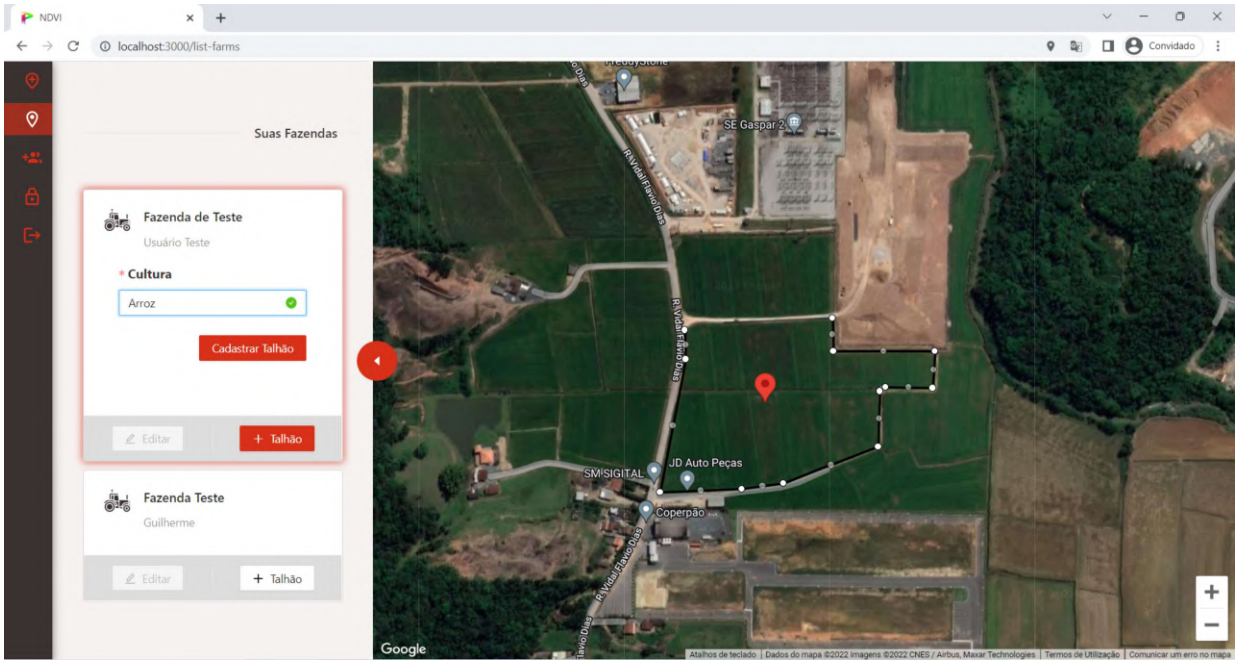
O cadastro de áreas de cultivo é feito através da botão **+Talhão** presente em cada uma das fazendas cadastradas. Ao selecionar essa opção, um pequeno formulário com o campo **Cultura** é apresentado ao usuário e o modo de inserção de polígonos é habilitado no mapa para permitir que o usuário informe o contorno que representa a sua área de cultivo. Na Figura 33 o processo de inserção de um novo polígono é apresentado.

Figura 32 – Listagem de Fazendas



Fonte: O Autor

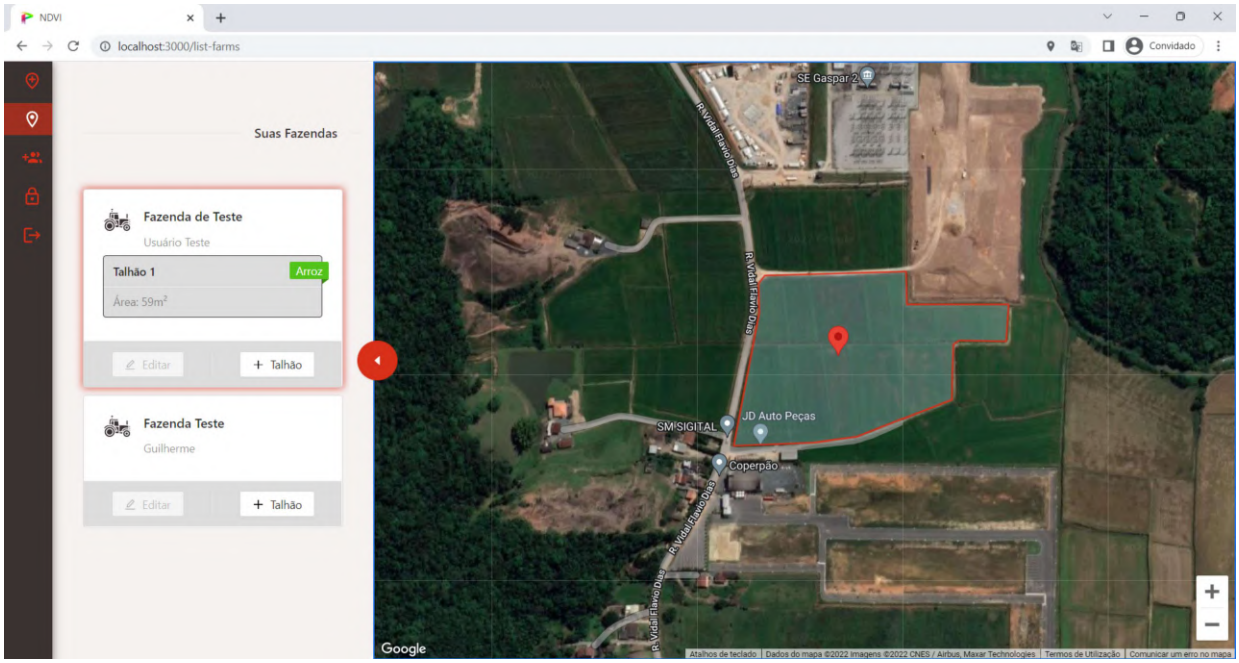
Figura 33 – Cadastro de uma nova cultura



Fonte: O Autor

Cada fazenda pode conter n talhões cadastrados, representando n áreas de cultivo distintas. Ao ser selecionado um dos talhões, uma sobreposição é feita no mapa para indicar o seu perímetro e localização, conforme demonstrado na Figura 34.

Figura 34 – Cultura cadastrada



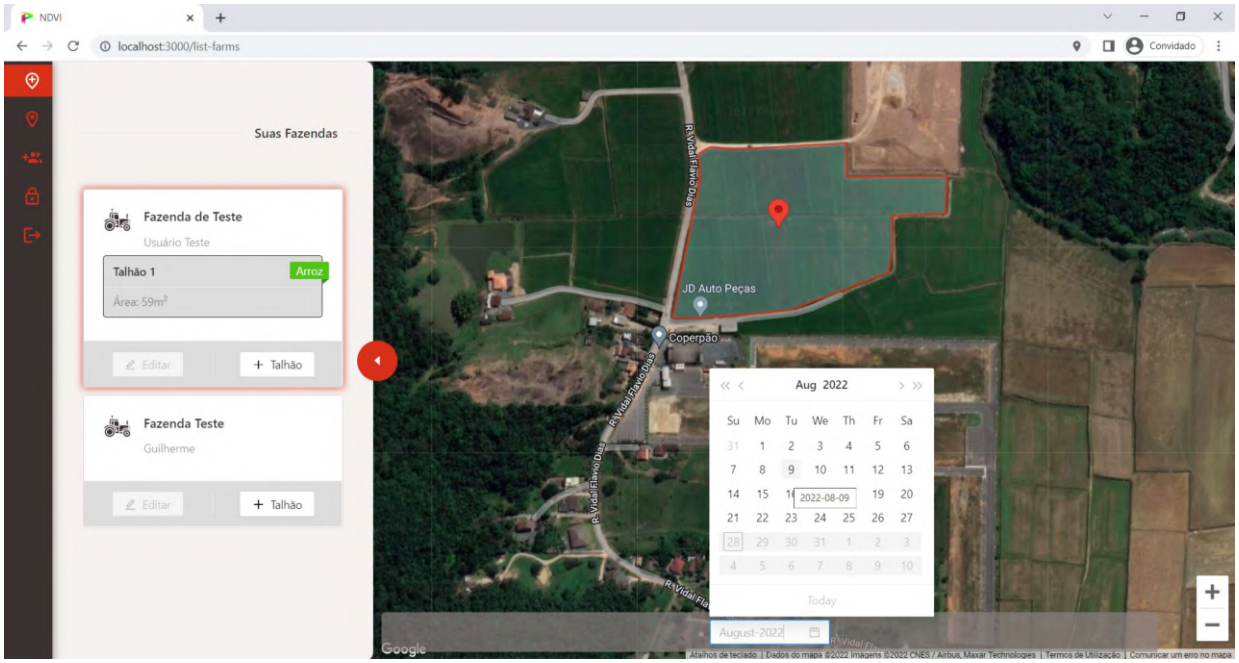
Fonte: O Autor

4.4 GERAÇÃO DO NDVI

Após selecionar um dos talhões, inicialmente um campo de entrada de data seria habilitado na parte inferior da tela para permitir ao usuário informar a data desejada para a geração do índice de vegetação por diferença normalizada, onde a partir dessa informação seriam buscadas imagens dentro de um intervalo de ±7 dias e um único índice seria computado, conforme apresentado na Figura 35. Entretanto, essa abordagem apresentou um grande problema na experiência de uso para casos onde a região estivesse com sobreposição de nuvens no período informado, pois nenhum resultado seria retornado caso não fossem encontradas imagens válidas para a geração do NDVI.

Buscando melhorar a experiência do usuário, optou-se por gerar automaticamente o índice de vegetação para várias datas - espaçadas por um intervalo de 7 dias - a partir da data da consulta. Dessa modo, ao selecionar um talhão o sistema automaticamente irá apresentar o resultado para 5 possíveis períodos computados a partir da data selecionada. Caso o usuário deseje visualizar mais resultados, um botão lateral permitirá essa funcionalidade, conforme demonstra a Figura 36. Na ocorrência de um período não possuir imagens possíveis de serem processadas, essa informação é também apresentada ao usuário, como representado na Figura 37

Figura 35 – Escolha da data para geração de NDVI



Fonte: O Autor

Figura 36 – Geração automática para vários períodos

Fonte: O Autor

Figura 37 – Mensagem para o usuário caso a sobreposição por nuvens tenha impossibilitado o cálculo do NDVI

Fonte: O Autor

Ao clicar em um dos resultados listados, o sistema irá gerar uma nova sobreposição no mapa, mas dessa vez contendo o NDVI desejado, conforme exemplificado pela Figura 38.

Figura 38 – Visualização do NDVI no mapa

Fonte: O Autor

5 CONCLUSÕES

O trabalho desenvolvido apresentou a construção de uma aplicação web voltada à geração e análise do índice de vegetação por diferença normalizada através do processamento de imagens de satélite. Dessa forma, através do processo de arquitetura e integração de diferentes sistemas, foram aplicados os conhecimentos obtidos nas disciplinas voltadas à área de sistemas computacionais.

A arquitetura cliente-servidor foi utilizada na construção da aplicação, onde diferentes tecnologias baseadas em Javascript foram empregadas no desenvolvimento do *front-end* e *back-end*. O *front-end* da aplicação foi construído utilizando o *framework* React, permitindo a criação de uma interface responsiva desenvolvida através de componentes reutilizáveis e uma alta performance devido ao seu controle de renderização. O *back-end* do sistema teve como base de sua arquitetura o Node.js, garantindo alto desempenho ao mesmo passo que possibilitou o uso de uma linguagem extremamente dinâmica como o Javascript.

Para gerar os indices de vetação por diferença normalizada, as regiões de interesse foram obtidas através da interface gráfica que foi integrada ao Google Maps. Empregando as informações georreferenciadas cadastradas pelo usuário por meio de um mapa, imagens do satélite Sentinel 2 foram obtidas a partir do Earth Engine para um dado intervalo de tempo e processadas para fornecer o NDVI da lavoura presente na fazenda.

A aplicação foi disponibilizada através de uma URL pública utilizando um servidor Linux configurado em uma instância do EC2 na AWS, o que a possibilita ser facilmente escalável através dos recursos oferecidos por esse provedor de serviços de nuvem.

Dessa forma, constata-se que os principais objetivos, descritos na Seção 1.1.2, deste trabalho foram integralmente alcançados. A solução desenvolvida é composta por diferentes tecnologias e integrada a diferentes sistemas, o que permitiu elucidar diversos conceitos relacionados a construção, comunicação e disponibilização de sistemas computacionais através da internet. A utilização da arquitetura cliente-servidor dentro do provedor de soluções de nuvem AWS possibilitou entender profundamente o poder de escalabilidade que a separação de responsabilidades proporciona ao permitir que diferentes partes da aplicação sejam escaladas isoladamente devido ao alto desacoplamento que a ausência de um monólito promove. Além disso, o desenvolvimento de uma interface responsiva permitiu expandir a usabilidade da aplicação para dispositivos móveis sem a necessidade de arcar com os custos relacionados ao desenvolvimento de uma nova aplicação exclusivamente voltada para esse fim.

O MVP resultante desse trabalho agora seguirá para sua fase de validação, onde será entregue agora ao usuário final para que sejam coletados *feedbacks* relacionados a usabilidade da aplicação e levantadas novas funcionalidades que tornem a solução ainda mais assertiva.

5.1 SUGESTÕES PARA TRABALHOS FUTUROS

Em relação as funcionalidades já existentes na aplicação, a possibilidade de comparar simultaneamente o NDVI de dois talhões distintos pode ser elencada como uma melhoria de grande relevância, trazendo uma nova perspectiva para as análises que o sistema já viabiliza. Para trazer mais detalhes sobre o contexto que originou o índice de vegetação computado, a inclusão de dados climáticos de umidade, temperatura e incidência solar para o período e região de interesse poderá contribuir para tornar as análises ainda mais ricas e assertivas. Além disso, porém agora voltado ao seu posicionamento no mercado como ferramenta gestão do agronegócio, o desenvolvimento de funcionalidades que possibilitem a gestão das fazendas e culturas aumentariam expressivamente o potencial competitivo da ferramenta.

REFERÊNCIAS

AGUIAR, Gustavo Stor de. Node.js Estudo Tecnológico e Desenvolvimento Full-Stack Javascript de Plataforma de Competições em Problemas Algorítmicos Trabalho de Graduação. **Universidade Federal de Pernambuco**, 2015.

AKANKSHA; CHATURVEDI, Akshay. **Comparison of Different Authentication Techniques and Steps to Implement Robust JWT Authentication**. [*S.l.*: *s.n.*], 2022. P. 772–779.

ALAM, Sawood; CARTLEDGE, Charles L.; NELSON, Michael L. Support for Various HTTP Methods on the Web. **Cornell University**, 2014.

ARAUJO, Massilon J. **Fundamentos do Agronegócio**. 6. ed. [*S.l.*]: Atlas, 2022.

BERTOLIN, Natalia de Oliveira; FILGUEIRAS, Roberto; VENANCIO, Luan Peroni; MANTOVANI, Everardo Chartuni. Predição da Produtividade De Milho Irrigado Com Auxílio De Imagens De Satélite. **Revista Brasileira de Agricultura Irrigada**, 2017.

BLOSFELD, Letycia Hass. Correlação linear entre os índices de vegetação NDVI e NDRE com a produtividade do milho segunda safra. **Universidade Federal de Mato Grosso**, 2014.

CHEN, Jiajia; CHENG, Weiqing. Analysis of web traffic based on HTTP protocol. *In*: 2016 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM). [*S.l.*: *s.n.*], 2016. P. 1–5.

DESHMUKH, Smita; MANE, Deepak; RETAWADE, Abhijeet. Building a Single Page Application Web Front-end for E-Learning site. **International Conference on Computing Methodologies and Communication (ICCMC)**, 2019.

DUARTE, Otto Carlos Muniz Bandeira. **Redes P2P**. [*S.l.*: *s.n.*], 2022. https://www.gta.ufrj.br/grad/12_1/p2p/introducao.html.

EMBRAPA. **A Agricultura Brasileira**. [*S.l.*: *s.n.*], 2017. https://www.embrapa.br/vii-plano-diretor/a-agricultura-brasileira.

EMBRAPA. **Visão 2030: o Futuro da Agriculutra Brasileira**. [*S.l.*: *s.n.*], 2018. https://www.embrapa.br/documents/10180/9543845/Vis%C3%A3o+2030++o+futuro+da+agricultura+brasileira/2a9a0f27-0ead-991a-8cbf-af8e89d62829.

JAVEED, Arshad. Performance Optimization Techniques for ReactJS. **IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)**, 2019.

JIAQI, Tian; ZHU, Xiaolin; WIU, Jin; SHEN, Miaogen; CHEN, Jin. Coarse-Resolution Satellite Images Overestimate Urbanization Effects on Vegetation Spring Phenology. **Remote Sensing**, v. 12, n. 1, p. 117, 2020.

MOZILLA.ORG. **Introdução ao DOM - APIs da Web**. [*S.l.*: *s.n.*], 2022. https://developer.mozilla.org/pt-BR/docs/Web/API/Document_Object_Model/Introduction.

NPM-STAT. **Statistics for Packages**. [*S.l.*: *s.n.*]. https://npm-stat.com/.

OLIVEIRA JÚNIOR, Laércio Germano de. ATLOM.JS: um Framework NODE.JS para aplicações Web baseado em componentes. **Universidade Federal do Ceará**, 2017.

PIX4D. **Optimizing the ROI of fungicides with NDVI imagery**. [*S.l.*: *s.n.*], 2018. https://www.pix4d.com/blog/pix4dmapper-optimizing-the-ROI-of-fungicides-with-NDVI.

PULUCENO, Thiago Vieira. Estudo de caso sobre uma API REST em Node.js. **Universidade Federal de Santa Catarina**, 2012.

QUARTAROLI, Carlos; VICENTE, Luiz Eduardo; ARAUJO, Luciana Spinelli. **Sensoriamento remoto**. [*S.l.*]: Embrapa, 2014.

REZNICK, João Paulo Kruger. Produtividade, qualidade industrial e nutricional na cultura do trigo. **Universidade Federal do Paraná**, 2017.

RISSINI, Adriano Luiz Lodi; KAWAKAMI, Jackson; GENÚ, Aline Marques. Índice de Vegetação por Diferença Normalizada e Produtividade de Cultivares de Trigo Submetidas a Doses de Nitrogênio. **Revista Brasileira de Ciência do Solo**, v. 39, n. 6, p. 1703–1713, 2015.

RIZK, Hashem; HABIB, Maki K. Robotized Early Plant Health Monitoring System. **IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society**, 2018.

SHIRATSUCHI, Luciano Shozo *et al.* **Agricultura de Precisão: Resultados de um Novo Olhar**: Sensoriamento Remoto: conceitos básicos e aplicações na Agricultura de Precisão. [*S.l.*]: Embrapa, 2014.

SILVA, Sezar Augusto Abadi. Manejo da população de plantas na cultura do milho a partir do NDVI obtido por imagens de satélite. **Universidade Federal de Santa Maria**, 2018.

SILVA NUNES, José Luis da. A Agricultura de Precisão como Ferramenta para o Produtor Rural, 2010.

STACK, Rising. **Node.js Under the Hood**. [*S.l.*: *s.n.*]. https://resources.risingstack.com/Node.js+at+Scale+Vol.+2+-+Node.js+Under+the+Hood.pdf.

STACKOVERFLOW. **Survey**. [*S.l.*: *s.n.*]. https://insights.stackoverflow.com/survey.

SULYMAN, Shakirat Haroon. Client-Server Model. **IOSR Journal of Computer Engineering**, 2014.

SUN, Xia; ZHAO, Yiming; YAO, Chang; YANG, Runping. Research and implementation of an efficient data persistence model. *In*: 2010 2nd International Conference on Future Computer and Communication. [*S.l.*: *s.n.*], 2010. v. 1, p. v1-361-v1–364.

SYSTEM, EARTH OBSERVING. **All you need to know about NDVI**. [*S.l.*: *s.n.*], 2019. https://eos.com/blog/ndvi-faq-all-you-need-to-know-about-ndvi.

TILKOV, Stefan; VINOSKI, Steve. Node.js: Using JavaScript to Build High-Performance Network Programs. **IEEE Internet Computing**, v. 14, n. 6, p. 80–83, 2010.

VELOG.IO. **DOM**. [*S.l.*: *s.n.*], 2021. https://velog.io/@lucylou/DOM.