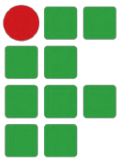


INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DA PARAÍBA - CAMPUS CAMPINA GRANDE
CURSO SUPERIOR BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

LUIZ MEDEIROS NETO

FUTS3: UMA PLATAFORMA PARA ANÁLISE DE DADOS DE JOGADORES DE FUTEBOL

Campina Grande

2024

Luiz Medeiros Neto

FUTS3: Uma Plataforma para Análise de Dados de Jogadores de Futebol

Monografia apresentada ao Curso de Bacharelado em Engenharia de Computação, do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba – Campus Campina Grande, em cumprimento às exigências parciais para a obtenção do título de Bacharel em Engenharia de Computação

Instituto Federal da Paraíba

Orientador: Prof. Dr.Emanuel Dantas Filho

Campina Grande

2024

M488f Medeiros Neto, Luiz
FUTS3: Uma plataforma para análise de dados de jogadores de futebol / Luiz Medeiros Neto. - Campina Grande, 2024.
63f. : il.

Trabalho de Conclusão de Curso (Curso Superior de Bacharelado em Engenharia de Computação) - Instituto Federal da Paraíba, 2024.
Orientador: Prof. Dr. Emanuel Dantas Filho.

1. Desempenho esportivo - análises e métodos 2. Engenharia de software 3. Software open source 4.Dantas Filho, Emanuel I. Título.

CDU 796.01

Luiz Medeiros Neto

FUTS3: Uma Plataforma para Análise de Dados de Jogadores de Futebol

Monografia apresentada ao Curso de Bacharelado em Engenharia de Computação, do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba – Campus Campina Grande, em cumprimento às exigências parciais para a obtenção do título de Bacharel em Engenharia de Computação

Trabalho aprovado. Campina Grande, 19 de setembro de 2024:

Prof. Dr.Emanuel Dantas Filho
Orientador

Prof. Dr. Ígor Barbosa da Costa
Membro da Banca

Prof. Me. Victor André Pinho de Oliveira
Membro da Banca

Campina Grande
19 de setembro de 2024

Este trabalho é dedicado a todos aqueles que estiveram presentes na minha caminhada, prestando forças e me oferecendo o que há de melhor no ser humano.

Agradecimentos

A princípio, gostaria de expressar meus agradecimentos a Deus, por me abençoar, fortalecer minha mente e colocar as pessoas certas em meu caminho. A vida é uma jornada de escolhas, companheirismo e oportunidades. Portanto, agradeço a todos aqueles que estiveram ao meu lado, oferecendo conselhos e compartilhando suas visões de vida, as quais agregaram imenso valor. Meu respeito e gratidão são direcionados a:

A minha família, Lázaro Medeiros, Magna Medeiros, Esther Medeiros, tios e avós pela educação, disciplina e ensinamentos valiosos que me forneceram, sendo meu alicerce nesta caminhada.

Ao Professor Victor Oliveira, pelo apoio durante meu primeiro projeto de pesquisa, pelos valiosos aprendizados e contribuições para o meu crescimento pessoal e profissional.

Ao meu orientador, Emanuel Dantas, que confiou em meu trabalho e proporcionou oportunidades para explorar grandes linhas de pesquisa, contribuindo significativamente para minha trajetória.

Aos professores Fagner Araújo, Gilson Lucena, Mary Karlla Guimarães e Moacy Pereira, que compartilharam experiências preciosas na área de eletrônica e ofereceram assistência em momentos cruciais.

A todos os Professores do IFPB - Campus Campina Grande, que desempenharam suas funções com maestria e foram de grande importância durante a minha graduação.

Aos amigos Jonathan Pontes, João Gabriel e Rennan Cavalcanti, pelo apoio ao longo do curso e pelos momentos memoráveis compartilhados.

Aos amigos Antônio Farias, Edriel José, João Victor Negreiros, Lucas Bivar, Luís Henrique Lima, Pedro Macêdo e Renysson Cavalcante pelo apoio fundamental durante minha jornada acadêmica e pelo partilhamento de conhecimento.

A todos os amigos que fiz no curso e que participaram de vários momentos: Alyson Vale, Arlan Santos, Ayrtton Lucas Silva, Daniel Barbosa, Diego Alex Maia, Diogo da Silva Santos, Eduardo Nogueira, Eduardo Reis, Elisson Barbosa, Estevao Holanda, Fabrício Domingos, Gabriel da Silva Nascimento, Ícaro Mendes, João Pedro de Lima, Josehylton Ricardo Neves, Lohan Oliveira, Lucas Alves, Lucas Andrade, Lucas Cordeiro, Mateus Albuquerque Pierre, Milena Lins Aguiar e Robson Luan.

"Queira-se ou não, acredite-se ou não, o futebol continua sendo uma das mais importantes expressões de identidade cultural coletiva, dessas que em plena era de globalização obrigatória nos recordam que o melhor do mundo está na quantidade de mundos que o mundo contém".
(Eduardo Galeano – Fechado por Motivo de Futebol)

Resumo

A análise de desempenho no futebol tem evoluído significativamente ao longo dos anos, combinando técnicas manuais com tecnologias avançadas. Comparar parâmetros e estabelecer medidas de desempenho é essencial para selecionar os jogadores ideais para um time de futebol, considerando suas condições e necessidades específicas. Nesse contexto, este trabalho apresenta a plataforma Futs3, desenvolvida para facilitar a análise de dados e o desempenho dos jogadores, além de auxiliar comissões técnicas na tomada de decisões. Assim, a ferramenta é uma solução *open source* que busca atender à crescente demanda por *softwares* de qualidade, acessíveis e eficientes para análise de desempenho, especialmente em clubes com recursos financeiros limitados. Ademais, com base em plataformas existentes, foram estabelecidos requisitos e melhorias, culminando na construção de um aplicativo web com alta disponibilidade. A plataforma oferece confiabilidade, personalização para atender às necessidades específicas dos clubes e mecanismos para associar os dados disponíveis ao modelo de jogo do clube. Dessa forma, o objetivo principal foi desenvolver um *software* robusto que permita uma análise detalhada e personalizada do desempenho dos jogadores, contribuindo para a otimização das estratégias táticas e o aprimoramento das habilidades individuais dos atletas.

Palavras-chave: Engenharia de Software; Desenvolvimento de sistemas; Sistemas web; Sistemas de análise tática; Análise de dados.

Abstract

The analysis of football performance has significantly evolved over the years, combining manual techniques with advanced technologies. Comparing parameters and establishing performance measures is essential to select the ideal players for a football team, considering their specific conditions and needs. In this context, this work presents the Futs3 platform, developed to facilitate data analysis and player performance evaluation, as well as to assist technical committees in decision-making. Therefore, the tool is an open-source solution that aims to meet the growing demand for quality, accessible, and efficient performance analysis software, especially in clubs with limited financial resources. Furthermore, based on existing platforms, requirements and improvements were established, culminating in the construction of a highly available web application. The platform offers a reliable cloud database, customization to meet the specific needs of clubs, and mechanisms to associate the available data with the club's playing model. Thus, the main objective was to develop robust software that allows for detailed and personalized analysis of player performance, contributing to the optimization of tactical strategies and the improvement of athletes' individual skills.

Keywords: Software Engineering; Systems Development; Web Systems; Tactical Analysis Systems; Data Analysis.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

AWS	Amazon Web Services
API	Application Programming Interface
CD	Continuous Development
CI	Continuous Integration
CSS	Cascading Style Sheets
DEV	Development
EC2	Elastic Compute Cloud
ECR	Elastic Container Registry
ECS	Elastic Container Service
GUI	Graphical User Interface
HATEOAS	Hypermedia As the Engine Of Application State
HTML	Hypertext Markup Language
IAM	Identity and Access Management
JWT	JSON Web Token
JPA	Java Persistence API
JVM	Java Virtual Machine
JSON	JavaScript Object Notation
MVC	Model-View-Controller
RDBMS	Relational Database Management System
RDS	Relational Database Service
REST	Representational State Transfer
S3	Simple Storage Service
SPA	Single-Page Applications

SSAS	Syntactically Awesome Style Sheets
SSH	Secure Shell
SQL	Structured Query Language
URL	Uniform Resource Locator
VPC	Virtual Private Cloud
XML	Extensible Markup Language
YAML	YAML Ain't Markup Language

Sumário

1 Introdução

No início da análise de desempenho no futebol, os processos de seleção eram realizados de maneira mais rudimentar e manual, assim como as decisões no jogo eram baseadas exclusivamente na experiência dos treinadores e membros da equipe técnica. Eles confiavam em seu conhecimento do jogo para tomar decisões sobre táticas, formação e substituições. Os clubes pioneiros em análise de desempenho conseguiam coletar apenas estatísticas simples e de forma manual, como chutes a gol, passes completos, desarmes, entre outras.

O uso de tecnologias começou a fazer parte da vida pessoal e profissional das pessoas, principalmente após a chegada dos computadores pessoais e Internet (PRESSMAN; MAXIM, 2021). Com o passar do tempo, o poder de processamento e armazenamento de sistemas computacionais mudaram a forma das pessoas realizarem suas atividades no dia a dia. E no futebol não seria diferente, muitas tecnologias foram inseridas no contexto esportivo para auxiliar os profissionais da área. Em específico para realizar análise de desempenho, podemos citar alguns marcos tecnológicos: Gravação de Vídeo Avançada, Análise de Vídeo e Software de Edição, Sistemas de Rastreamento e Sensores em Campo, Sistemas de Câmeras Múltiplas e *Replay* Instantâneo. O uso desses recursos mudaram a forma dos profissionais enxergarem o futebol. Assim, os departamentos de análise de desempenho ganharam mais destaque, e atualmente é consenso que os times que investem em tecnologia possuem mais subsídios para tomar decisões na montagem de elenco, nos treinamentos e na análise dos pontos fortes e fracos dos adversários.

As tecnologias trouxeram diferentes dados para serem usados pelos profissionais do futebol. Assim, a extração desses dados e a correta análise para transformar em informações úteis tornou-se uma área de grande importância nos clubes. A análise de dados desempenha um papel fundamental na otimização do desempenho dos jogadores de futebol. Ao analisar dados relacionados ao desempenho individual, como número de gols marcados, assistências, precisão nos passes, entre outros, é possível identificar pontos fortes e fracos de cada jogador. Com base nesses *insights*, é possível desenvolver treinamentos personalizados que visam aprimorar as habilidades específicas de cada atleta (AWARI, 2023).

Os analistas desempenham um papel fundamental no desenvolvimento e sucesso das equipes, dos jogadores e dos treinadores (SHAMAH, 2021). A área está em constante crescimento, com uso de tecnologias como ferramentas para impulsionar o desempenho de jogadores. Esse olhar científico para o desempenho no esporte, tem proporcionado um grande avanço no entendimento dos jogos, trazendo contribuições relevantes para a evolução do futebol (VOLOSSOVITCH; FERREIRA, 2013). Desse modo, surge a necessidade de

ferramentas que facilitem a personalização das formações com base nos dados coletados por essas ferramentas.

O enfoque científico no desempenho esportivo tem desempenhado um papel fundamental no entendimento do jogo, promovendo avanços significativos nas últimas décadas e contribuindo substancialmente para a evolução do futebol. Com o suporte do desenvolvimento tecnológico e o contínuo aprimoramento profissional, as equipes têm se preparado de maneira cada vez mais abrangente, influenciando de forma marcante o contexto competitivo e buscando constantemente a excelência no meio futebolístico. Em razão disso, clubes de alto nível passaram a investir em departamentos altamente estruturados para análise do jogo (GÓMEZ-RUANO, 2017; PEDREÑO, 2018; VENTURA, 2013).

Nesse sentido, é evidente o aumento do investimento por parte de clubes de alto nível na construção de setores para análise de dados (estatística) dos jogadores. Esses setores destacam a importância atribuída à análise de dados e estratégias no futebol moderno, bem como mostram a necessidade constante de adaptação e inovação dos times para se manterem competitivos em um ambiente esportivo que evolui freneticamente.

Portanto, temos um cenário em que é notório a análise de desempenho dos jogadores para otimizar o desempenho das equipes, refinando estratégias táticas, melhorando o desempenho da equipe e posteriormente obtendo os resultados desejados. É nesse contexto de ferramentas de análise de dados que insere-se esse trabalho, no qual foi desenvolvida a plataforma Futs3 com intuito de facilitar a análise de dados de jogadores e a extração de informação útil para as comissões técnicas.

1.1 Justificativa e Relevância do Trabalho

Com uso de diferentes tecnologias no ambiente futebolístico, o volume de dados para análise tem tamanho significativo. Usar esses dados de forma correta e extrair informação realmente útil tornou-se um desafio. O alto volume dos dados dificulta a análise como forma de buscar o sucesso esportivo das equipes (CARLING; WILLIAMS; REILLY, 2005; GLAZIER, 2010; VOLOSSOVITCH; FERREIRA, 2013).

Portanto, os clubes se veem com uma enormidade de ferramentas, cada uma com diferentes dados. Vale destacar que muitas dessas ferramentas são pagas, com valores que podem ser impeditivos para uso em contextos onde o orçamento é limitado. O alto número de ferramentas e dados obtidos por meio de plataformas de análise de desempenho, e a falta de profissionais capacitados para operar esses recursos faz com que a grande maioria dos clubes ainda realizem a análise de desempenho de forma manual.

Diante do exposto, surge a necessidade de criar uma plataforma que possa suprir as lacunas mencionadas. A plataforma Futs3 tem como foco simplificar a análise dos dados,

com um mecanismo de personificação onde cada clube poderá selecionar os dados que achar importantes. Além disso, a plataforma oferece mecanismos para associar o modelo de jogo do clube com os dados disponíveis. Por fim, destacar que a plataforma Futs3 é *open source*, facilitando o uso por clubes com poucos recursos financeiros.

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é desenvolver uma plataforma denominada Futs3 para análise de dados de jogadores de futebol, através de tecnologias modernas e com alta performance.

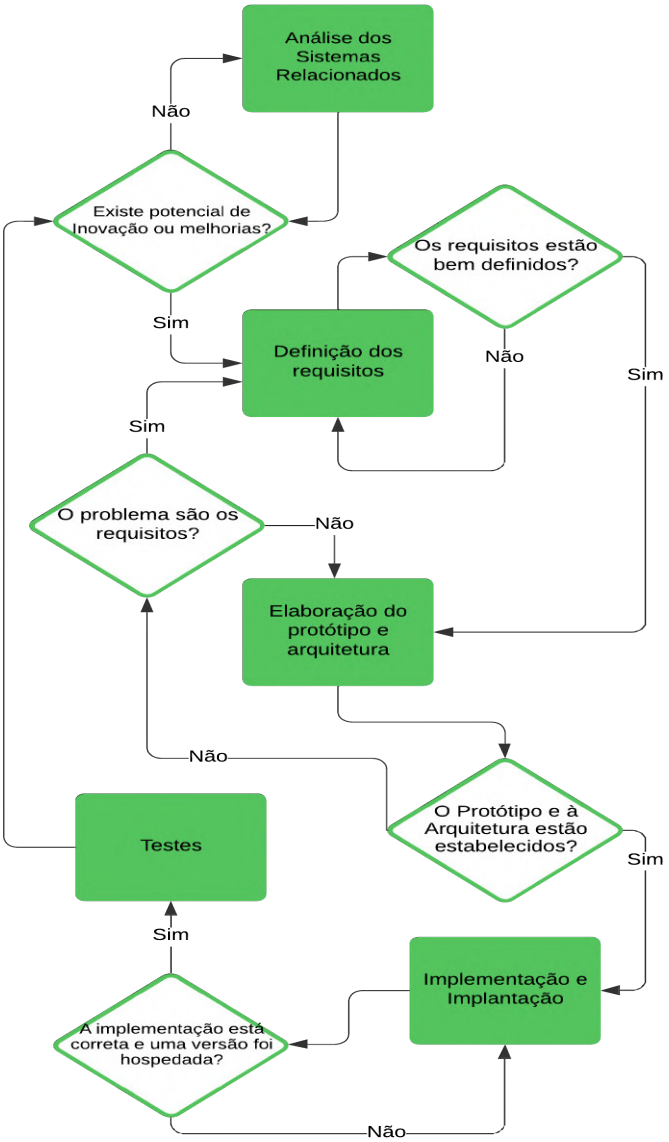
1.2.2 Objetivos Específicos

- Alta Disponibilidade: Desenvolver um software robusto capaz de armazenar e gerenciar informações críticas para análise de jogadores de futebol, garantindo sua disponibilidade contínua *(uptime)* de 90% por mês através de um banco de dados altamente confiável hospedado na nuvem.
- Análise de Similaridade com IA: Implementar funcionalidades que permitam realizar comparações detalhadas entre jogadores utilizando inteligência artificial, levando em consideração parâmetros de desempenho, bem como características técnicas e de extra-campo, oferecendo *insights* avançados e precisos para a tomada de decisões e *scouting*.
- Personalização e Flexibilidade: Oferecer uma plataforma altamente personalizável que permita aos clubes adaptar as análises de acordo com suas próprias necessidades e objetivos específicos, levando em consideração o estilo de jogo, as estratégias táticas e as preferências individuais dos treinadores e comissões técnicas, reduzindo o tempo de análise em mais de 60% em relação a outras plataformas.
- Redução de Custos: Fornecer uma solução acessível e de baixo custo para análise de desempenho de jogadores, especialmente para clubes com orçamentos limitados, através da disponibilização da plataforma como um software *open-source*.

2 Metodologia

A metodologia para o desenvolvimento da plataforma Futs3 foi dividida em várias etapas, por meio do desenvolvimento contínuo e incremental abrangendo desde a concepção e design até a implementação e implantação em nuvem. Convém lembrar ainda que, devido ao desenvolvimento incremental, todas as etapas se relacionam diretamente, como mostra o fluxograma a seguir:

Figura 1 – Fluxograma de Desenvolvimento do Projeto



Fonte: Elaborado pelo autor, 2023

2.1 Busca por Sistemas Relacionados

Na fase inicial do desenvolvimento, foi realizada uma pesquisa sobre os sistemas de análise de *scouts* já existentes no mercado. Essa análise teve como objetivo identificar as principais funcionalidades oferecidas por essas plataformas, bem como avaliar suas limitações e potenciais áreas para inovação.

Os sistemas analisados incluem Wyscout, Sportsbase e SofaScore, que são amplamente utilizados na análise de desempenho esportivo, eles estão entre os 13 melhores sites de estatísticas de futebol (ABSEITS, 2024). Cada uma dessas plataformas possui características distintas que atendem a diferentes usuários e suas necessidades. Todavia, durante uma análise mais profunda, foi possível identificar lacunas que permitem o desenvolvimento de um sistema mais personalizável e dinâmico com formações técnicas, especialmente em termos de *rankeamento* de jogadores conforme posições parametrizadas pelo usuário, análise de similaridade com IA e o custo elevado.

O Wyscout é uma plataforma italiana de análise de futebol, conhecida como "a maior biblioteca de vídeos e dados de futebol do mundo". Fundada em 2004, a Wyscout oferece ferramentas de análise de vídeo e uma base de dados abrangente que apoia clubes, agentes, jogadores e outros profissionais do futebol em processos de *scouting*, análise de partidas e transferências de jogadores. A plataforma coleta dados e vídeos de mais de 600 competições globais, permitindo que clubes identifiquem e avaliem talentos potenciais de maneira eficiente antes de enviar olheiros para observações presenciais. Além disso, a Wyscout fornece relatórios detalhados sobre competições, partidas, equipes e jogadores, integrando suas funcionalidades com outras ferramentas de análise, como o Hudl, para otimizar o fluxo de trabalho das equipes esportivas (WYSCOUT, 2024).

O SportsBase é uma plataforma digital de análise e vídeo que oferece uma solução completa e personalizável para profissionais do futebol, como técnicos, analistas e jornalistas. A plataforma fornece uma ampla gama de serviços, incluindo relatórios pós-jogo detalhados, coleta de dados em tempo real, rastreamento de condicionamento físico de jogadores e análise estatística avançada. Com mais de 2000 métricas disponíveis, os usuários podem filtrar jogadores por estatísticas específicas e vincular esses dados diretamente a vídeos para uma análise aprofundada. A SportsBase também oferece recursos como monitoramento de condicionamento físico, relatórios analíticos detalhados sobre o desempenho dos jogadores e das equipes, além de ferramentas de análise de vídeo e estatísticas, sendo uma solução robusta para clubes que buscam otimizar seus processos de *scouting* e gestão de dados esportivos (SPORTSBASE, 2024).

Por último, SofaScore é um aplicativo de esportes que oferece resultados ao vivo e estatísticas detalhadas para mais de 20 esportes e cerca de 11.000 ligas e torneios. O aplicativo, desenvolvido pela SofaIT na Croácia, é amplamente utilizado por milhões

de usuários em todo o mundo, sendo particularmente popular entre os fãs de futebol. SofaScore se diferencia por transformar dados brutos em gráficos e visualizações intuitivas, ajudando os usuários a entender rapidamente o que está acontecendo em uma partida. Além disso, o aplicativo oferece um sistema de avaliação de desempenho de atletas que permite acompanhar e avaliar jogadores de ligas inferiores, tornando-o uma ferramenta valiosa tanto para fãs quanto para analistas (SOFASCORE, 2024).

A tabela a seguir resume as principais funcionalidades definidas no Futs3 em relação a essas plataformas e se elas foram atendidas:

Requisitos	**Wyscout**	**Sportsbase**	**SofaScore**	**Futs3**	**Atendidos**
Análise de Similaridade		x		x	
Baixo Custo			x	x	x
Personalização de Formações	x			x	x
Rankeamento com Base em Formações	x	x	x	x	x

Tabela 1 – Requisitos atendidos pelas plataformas e o Futs3

- Wyscout: Oferece uma vasta gama de dados estatísticos, vídeos de partidas e marcações de jogadores em vídeo, sendo amplamente utilizado por clubes e olheiros. Além disso, apresenta jogadores *rankeados* por posição e permite a personalização de formações, algo semelhante ao Futs3, embora sem o uso de inteligência artificial para análise de similaridade.

- Sportsbase: Similar ao Wyscout, o Sportsbase fornece análises detalhadas, rankings de jogadores e recursos de análise de similaridade. No entanto, ele não permite a personalização de formações para parametrizar os *rankings*, algo que é possível no Futs3.

- SofaScore: É focado na visualização de dados ao vivo e tem um custo mais baixo. Sua funcionalidade de análise é limitada, e ele não oferece personalização de formações nem análise de similaridade com o uso de inteligência artificial, sendo menos adequado para fins de treinamento ou *scouting* avançado.

A partir dessa análise, concluiu-se que o desenvolvimento do Futs3 é altamente viável, tendo em vista personalização, baixo custo e utilização de inteligência artificial para análises de similaridade de jogadores, o que deve atender um mercado variado, considerando clubes de diferentes tamanhos, treinadores e analistas de desempenho que necessitam de ferramentas mais adaptáveis e acessíveis.

2.2 Definição dos Requisitos

Nessa etapa, foram realizadas pesquisas de mercado e conversas com profissionais da área para obter informações que orientem o desenvolvimento contínuo e a manutenção da plataforma, de forma a atender às necessidades reais dos usuários finais.

2.2.1 Histórias de Usuário

Durante a fase inicial de definição dos requisitos, foram elaboradas histórias de usuário para capturar os diferentes cenários de uso do sistema para a primeira versão. Exemplos de histórias de usuário incluem:

- Como treinador, quero personalizar diferentes formações táticas para minha equipe, com o objetivo de testar diferentes estratégias e definir os melhores jogadores para um jogo.
- Como analista de desempenho, quero comparar as estatísticas dos meus jogadores com jogadores similares, para identificar pontos que precisam de atenção e melhorias.
- Como diretor de clube, quero visualizar relatórios detalhados de desempenho com base nas formações táticas, por um custo acessível, e assim tomar decisões que levem em consideração dados estatísticos.

Essas histórias de usuário foram priorizadas e organizadas em um *backlog*, que serviu para nortear o desenvolvimento de uma versão inicial das funcionalidades da plataforma, assim como novas versões.

2.2.2 Requisitos Funcionais

Os requisitos funcionais especificam as funcionalidades que a plataforma Futs3 deve ter para atender às histórias de usuário. Entre os principais requisitos funcionais, destacam-se:

- Cadastro, Edição e Listagem de Usuários: Permitir que novos usuários se registrem na plataforma e que eles tenham acesso a seus dados pessoais, podendo editá-los quando necessário.
- Cadastro, Edição e Listagem de Jogadores: O sistema deve possibilitar operações de *backoffice* de jogadores, permitindo o cadastro, edição e listagem de informações detalhadas sobre cada jogador.

- Cadastro, Edição e Listagem de Parâmetros: Proporcionar a definição e gestão de parâmetros que serão utilizados na análise e comparação de jogadores, como métricas de desempenho, atributos físicos e técnicos.
- Cadastro, Edição e Listagem de Posições: Possibilitar o cadastro e a organização das diferentes posições táticas, permitindo que essas sejam parametrizadas de acordo com as necessidades específicas de cada equipe.
- Cadastro, Edição e Listagem de Modos de Jogo: Oferecer opções para o cadastro e gestão de diferentes modos de jogo, como formações táticas específicos, partidas amistosas e competições, com a finalidade de prover uma análise contextualizada do desempenho.
- Personalização de Formações: Permitir que os usuários customizem as formações das equipes de acordo com suas necessidades táticas, facilitando a criação de estratégias personalizadas.
- Diversificação de Jogadores: Oferecer suporte para diferentes perfis de jogadores, permitindo uma análise detalhada e personalizada, que leve em conta as características únicas de cada jogador.
- *Rankeamento* de Jogadores: Implementar um sistema de *ranking* que posicione os jogadores de acordo com seu desempenho em diferentes métricas, considerando a formação e posição configuradas pelo treinador para estabelecê-los.
- Análise de Similaridade com IA: Integrar serviços de inteligência artificial à aplicação para comparar jogadores e equipes, sugerindo alternativas ou melhorias com base em dados estatísticos.
- Relatórios Customizáveis: Gerar relatórios detalhados que possam ser customizados pelos usuários para atender diferentes objetivos e públicos.

2.2.3 Requisitos Não Funcionais

Além dos requisitos funcionais, foram definidos requisitos não funcionais que garantem a qualidade e o desempenho da plataforma:

- Usabilidade: A interface da plataforma deve ser intuitiva e objetiva, visando aumentar a acessibilidade e diminuir a curva de aprendizado dos usuários.
- Desempenho: A plataforma deve ser capaz de processar grandes volumes de dados sem comprometer a performance, utilizando *schedule jobs* em horários de menor atividade dos usuários.

- Escalabilidade: A arquitetura da plataforma deve permitir o escalonamento, suportando um número crescente de usuários e dados, mantendo uma comunicação fluente entre servidor e cliente.
- Segurança: A plataforma deve garantir a segurança dos dados dos usuários, implementando práticas como criptografia de dados sensíveis com algoritmos como o *Argon2* para *hash* de senhas, estratégias de *token* e *refresh token*, além do controle de acesso por permissionamento com *roles*.
- Custo: O desenvolvimento e manutenção da plataforma devem ser realizados de forma a minimizar os custos, tornando-a acessível para pequenas e médias organizações.

2.3 Definição do Protótipo e Arquitetura

A fase de definição do protótipo e arquitetura foi crucial para garantir que a plataforma Futs3 atendesse de forma eficiente os requisitos funcionais identificados. Assim como, a arquitetura da solução foi projetada para ser modular e escalável, permitindo que novas funcionalidades sejam adicionadas ao sistema com facilidade, mantendo a compatibilidade com versões anteriores e sem comprometer o desempenho ou a estabilidade da plataforma.

2.3.1 Protótipo

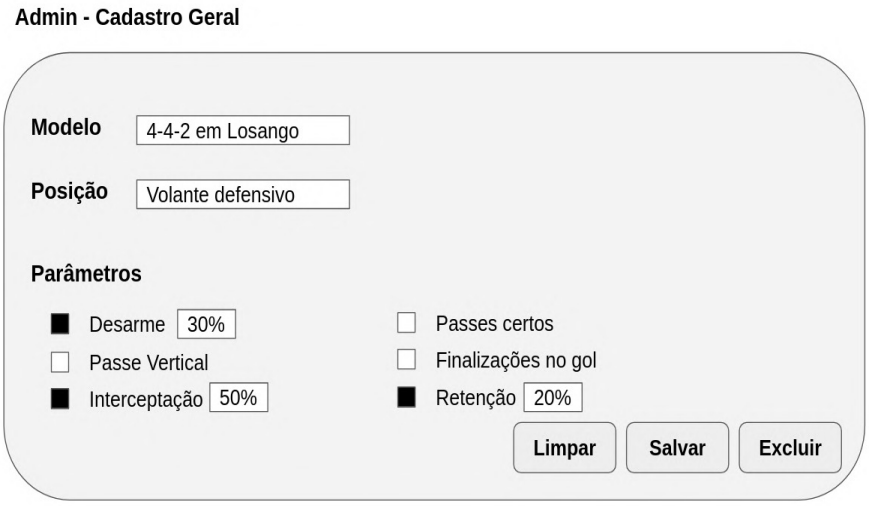
A princípio, foi desenvolvido um protótipo simples e funcional utilizando ferramentas de design como Figma e Canva. Esse protótipo serviu como base para entender alguns dos principais fluxos do aplicativo e garantir a concepção inicial da usabilidade do sistema. As interfaces foram projetadas com foco na simplicidade e acessibilidade, assegurando que usuários com diferentes níveis de experiência possam utilizar a plataforma sem maiores dificuldades.

A seguir, encontram-se duas modelagens de telas principais do sistema:

A Figura 2 apresenta a tela de Seleção de Parâmetros e Posição, na qual o usuário pode selecionar e atribuir pesos a critérios específicos para a análise dos jogadores de uma determinada posição, como a de volante defensivo.

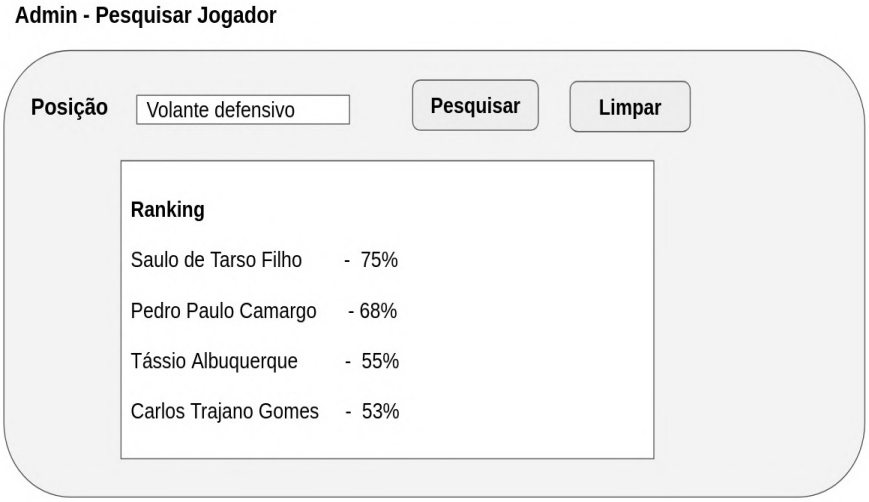
A Figura 3 ilustra a tela de Ranking de Jogadores com Base na Posição. Logo, é possível visualizar os melhores jogadores cadastrados de acordo com os critérios definidos anteriormente. Essa seção facilita a análise comparativa destacando aqueles que melhor se adaptam às exigências técnicas estabelecidas pelo usuário.

Figura 2 – Tela de Seleção de Parâmetros e Posição



Fonte: Elaborado pelo autor, 2023

Figura 3 – Tela de Ranking de Jogadores com Base na Posição



Fonte: Elaborado pelo autor, 2023

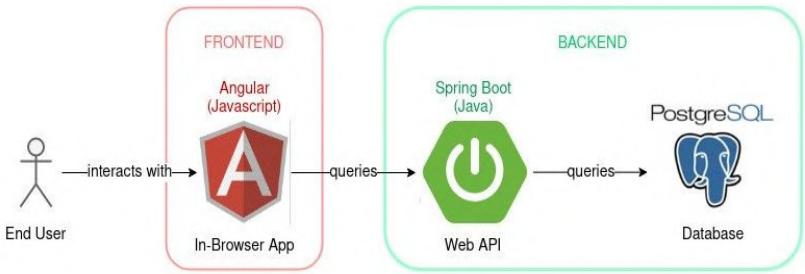
2.3.2 Arquitetura do Sistema

A arquitetura da plataforma foi baseada em uma abordagem de cliente-servidor, tendo em vista que o cliente é uma SPA feita em Angular e do lado do servidor identifica-se uma API principal construída em Spring, que receberá as requisições da SPA e de APIs de terceiros ou microsserviços nos quais estão delegadas outras funcionalidades. Esses serviços são implementados de maneira independente, o que permite uma maior flexibilidade e facilita o escalonamento da aplicação.

A Figura 4 ilustra a arquitetura geral da plataforma, bem com destaca a comunicação entre os principais componentes que fazem parte da aplicação junto as tecnologias

utilizadas para desenvolver cada parte.

Figura 4 – Arquitetura Geral do Sistema



Fonte: (TALKS, 2024)

Desse modo, vale ressaltar os seguintes pontos:

- *Frontend*: Trata-se de uma SPA desenvolvida em Angular, com o objetivo de garantir alta manutenibilidade, reatividade, responsividade e disponibilidade para o usuário, o que garante uma experiência fluida.
- *Backend*: Foi implementado em Java utilizando o ecossistema do Spring. É delegado ao *backend* a responsabilidade de processar dados inseridos pelo usuário, integrar-se a serviços de IA e gerenciar a comunicação entre os diferentes serviços.
- Banco de Dados: Utilizou-se um banco de dados relacional (PostgreSQL) para armazenar os dados, que são estruturados e relacionados, o que garante organização e uma modelagem apropriada.
- Infraestrutura em Nuvem: A plataforma foi implantada em uma infraestrutura de nuvem escalável na AWS, garantindo alta disponibilidade, segurança e desempenho.

2.4 Implementação e Implantação

Após as definições da arquitetura e do protótipo, inicia-se a fase de implementação. Nesta etapa, as funcionalidades identificadas são codificadas, testadas e integradas ao software, de maneira que a compatibilidade com versões anteriores seja mantida. Dentre as tecnologias escolhidas para a implementação e implantação da plataforma incluem-se, *frameworks* robustos e práticas de *DevOps* para garantir uma entrega contínua e eficiente.

2.4.1 Implementação

Cada módulo da plataforma foi implementado conforme as especificações definidas nos requisitos. A implementação segue metodologias ágeis e o desenvolvimento contínuo, com *sprints* quinzenais, nas quais o progresso é revisado e os ajustes são feitos conforme as observações. Alguns dos principais módulos são:

- Coleta de Dados: Esse módulo é responsável pela coleta de dados inseridos manualmente pelo usuário por meio do cliente, utilizando o *dashboard* da aplicação ou o uso de planilhas, assim como APIs de terceiros para a coleta automatizada de dados.
- Armazenamento de Dados: Nesse módulo da aplicação, os dados coletados são organizados para serem armazenados em uma instância do RDS, que é um sistema de banco de dados em nuvem compatível com o PostgreSQL e focado na eficiência de consultas e integridade dos dados.
- Análise Estatística: O módulo de estatísticas abrange toda a parte de *scouting* desde a geração de *rankings* a análises gráficas, Para isso, implementou-se serviços com algoritmos responsável por trazer esses *insights*. Ferramentas como chart.js foram utilizadas para a criação de gráficos oportunos que ajudam os usuários a visualizarem os dados de forma clara e intuitiva, notando as diferenças entre os jogadores que estão sendo analisados.
- Personalização: Os componentes do módulo de personalização permitem aos usuários personalizarem as análises técnicas, formações e relatórios, além disso o desenvolvido foi realizado priorizando flexibilidade e usabilidade.

2.4.2 Implantação

Após a implementação de uma versão, ela deve ser implantada em nuvem, utilizando práticas de *Continuous Integration/Continuous Deployment (CI/CD)* para automatizar o processo de *deploy*. Com isso, destaca-se os seguintes pontos:

- *Deploy* Automatizado: Essa configuração acontece por meio da utilização de ferramentas como GitHub Actions e Docker para automatizar o processo de *build* e *deploy* da aplicação junto aos serviços da AWS.
- Monitoramento: Para acompanhamento da aplicação em nuvem, foram usadas as ferramentas de monitoramento da AWS.
- Escalabilidade: Para garantir o fator de escalabilidade no sistema, é necessária a configuração de *auto scaling* através dos recursos da AWS para assegurar que a plataforma possa lidar com picos de demanda sem comprometer a experiência do usuário.

2.5 Testes

Os testes desempenham um papel fundamental no desenvolvimento e manutenção software abordado nesse trabalho, sendo uma das etapas mais críticas para assegurar a qualidade, confiabilidade e segurança de uma aplicação antes do lançamento de uma nova versão em um ambiente de produção. No contexto do Futs3, no qual a aplicação envolve funcionalidades sofisticadas como personalização de formações, *rankeamento* de jogadores e análise de similaridade utilizando Inteligência Artificial (IA), é evidente que os testes devem ter uma alta cobertura nos diferentes componentes do sistema e para uma análise detalhada, os testes dividiram-se em testes de unidade, integração e desempenho.

2.5.1 Testes de Unidade

Os testes de unidade são a base de uma estratégia de testes eficaz. No Futs3, cada módulo foi submetido a uma série de testes de unidade para validar o comportamento individual de suas funcionalidades. Do lado do *backend* implementou-se testes com a utilização do JUnit em combinação com o Mockito, que é uma poderosa ferramenta de *mock* para simular comportamentos de objetos e facilitar a validação de funções isoladas.

2.5.2 Testes de Integração

Os testes de integração são a etapa seguinte e têm como objetivo verificar se os diferentes módulos do sistema Futs3 interagem corretamente entre si. Esses testes são essenciais para detectar problemas que podem não ser aparentes nos testes de unidade, tais como falhas de comunicação entre serviços ou inconsistências nos fluxos de dados e até mesmo na modelagem do banco de dados. Os testes de integração são voltados a verificar operações entre o módulo de gestão de usuários, que inclui autenticação e autorização, e o módulo de geração de *insights*, no qual as análises de dados e IA são processadas. Esses testes garantem que, por exemplo, as informações de um jogador sejam corretamente processadas e que os *insights* gerados sejam baseados em dados atualizados e precisos.

2.5.3 Testes de Desempenho

Por último, os testes de desempenho são essenciais para garantir que a aplicação está lidando de forma eficiente com a carga de trabalho esperada, especialmente ao considerar cenários com múltiplos usuários acessando o sistema simultaneamente. Para isso, com o uso do JMeter é possível simular esses cenários de forma que o sistema receba inúmeras requisições por segundo, permitindo medir métricas críticas como tempo de resposta, *throughput*, e uso de recursos (CPU, memória, etc.). Esses testes são cruciais não apenas para validar a performance sob carga, mas também para identificar possíveis gargalos em serviços já desenvolvidos e pontos que podem comprometer a experiência do usuário

final. Além disso, o JMeter permite a execução de testes em diferentes configurações de infraestrutura, ajudando a determinar a escalabilidade do sistema e sua capacidade de manter um desempenho aceitável conforme o número de acesso simultâneos cresce.

Portanto, o conjunto de testes abordado, quando executados de forma abrangente e sistemática, fornecem mais confiança na hora de disponibilizar uma versão em produção garantindo ao usuário final uma experiência agradável, fluida e segura.

3 Arquitetura

Esta seção tem como objetivo apresentar as definições da arquitetura do sistema e as ferramentas essenciais para o entendimento desse trabalho. As escolhas dessas tecnologias foram cruciais no resultado final do processo de desenvolvimento, uma vez que elas influenciaram diretamente a qualidade e a eficiência do software desenvolvido.

3.1 Arquitetura de Software

A arquitetura de software define a organização, a comunicação e a disponibilidade dos componentes de um sistema, o que determina a sua longevidade, implicando na manutenção ao longo do tempo e impacta diretamente a experiência do usuário final. A arquitetura de software não é apenas sobre o que o sistema faz, mas como ele faz o que faz (FOWLER, 2003). Logo, ela envolve decisões altamente relevantes que impactam diretamente na escalabilidade, performance e segurança do sistema.

A arquitetura de software, portanto, deve ser estabelecida com um olhar crítico sobre a aplicação, de forma que sejam consideradas tanto as suas necessidades atuais quanto as futuras. Uma arquitetura eficiente permite gerenciar a complexidade inerente ao desenvolvimento de software, de maneira que a compreensão, a comunicação e a evolução do sistema sejam facilitadas (BASS L.; CLEMENTS, 2003). Dessa forma, a arquitetura não só organiza um sistema como um todo, mas também possibilita a sua adesão a novas funcionalidades e demandas tecnológicas.

3.1.1 API REST

O modelo de uma API REST (*Representational State Transfer*) é uma das abordagens mais populares e eficientes para a construção de servidores web devido à sua flexibilidade para tráfego de requisições e o seu contexto de usabilidade. Essa arquitetura de API é baseada nos princípios de comunicação *stateless*, ou seja, o REST permite a integração entre diferentes sistemas utilizando métodos HTTP padrão. As APIs REST permitem que serviços sejam expostos de forma clara e acessível, facilitando a interoperabilidade entre diferentes plataformas e dispositivos (RICHARDSON LEONARD; RUBY, 2020). Desse modo, essa interoperabilidade e a capacidade de manipular dados estruturados em formato JSON, XML ou YAML, torna a REST ideal para o desenvolvimento de aplicações web escaláveis e interconectadas.

3.1.2 API RESTful

A API RESTful tem seus princípios fundamentais baseados no padrão REST, porém ela permite que diferentes sistemas interajam de maneira consistente e previsível. Dessa forma, essa padrão arquitetural é orientado a recursos, nos quais cada um deles é identificado por uma URL de acesso única, assim como as operações realizadas sobre esses recursos seguem as convenções do protocolo HTTP e HATEOAS.

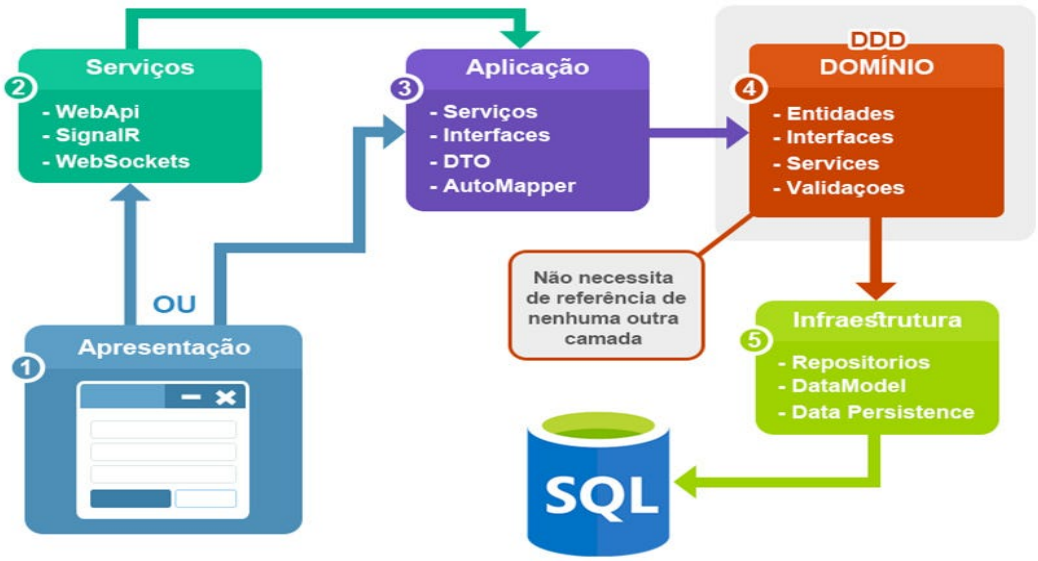
O modelo RESTful desempenha um papel crucial na construção de arquiteturas de microsserviços, tendo em vista que cada serviço é projetado para ser especializado e desacoplado dos demais. Assim como, a característica *stateless* das APIs RESTful contribui significativamente para a escalabilidade e a resiliência dos sistemas, uma vez que cada requisição é independente e não carrega informações de estado.

3.1.3 Arquitetura em três camadas

A arquitetura em três camadas é um padrão utilizado para organizar a estrutura de sistemas com alta eficácia e confiabilidade, dividindo-o em três partes principais com responsabilidades distintas. Essa organização facilita a manutenção e a escalabilidade do sistema, visto que permite que as diferentes camadas sejam implementadas e testadas de forma independente, o que contribui significativamente para a flexibilidade e a robustez do software (FOWLER, 2003).

Na Figura 5, destaca-se o que envolve cada camada, como entidades, interfaces e validações.

Figura 5 – Arquitetura em Três Camadas



Fonte: (ANDRADE, 2023)

- A Camada de Apresentação é destinada a parte de comunicação com o cliente, ou seja, é responsável por exibir dados ao usuário e disparar eventos para uma API. Nessa camada fica tudo que envolve a interação entre usuário e sistema, incluindo interfaces gráficas e a apresentação visual das informações. Com isso, é possível garantir uma experiência de usuário intuitiva e eficiente.

- A Camada de Serviços é a parte central da arquitetura, uma vez que é responsável por conter as regras de negócio da aplicação, bem como a lógica pela qual os dados são processados pela aplicação. Essa camada mantém um contrato, que faz a intermediação entre a Camada de Apresentação e a Camada de Acesso a Dados, aplicando as regras que definem o funcionamento do sistema. Além disso, ela assegura que os dados sejam processados corretamente, mantendo a integridade dos dados conforme os requisitos do software.

- A Camada de Aplicação é responsável por realizar a comunicação com o banco de dados. Logo, ela estabelece operações de leitura e escrita de dados, desacoplando essas funcionalidades das demais camadas. Essa separação permite que a lógica de acesso a dados seja modificada sem impactar as camadas de apresentação ou lógica de negócio, o que facilita a manutenção e escalabilidade do sistema.

Entretanto, Fowler destaca que essa estrutura pode sofrer alterações e deve ser adaptada às necessidades específicas de cada aplicação, como é o caso do servidor da plataforma, no qual é possível identificar as três camadas principais e as camadas secundárias de domínio e infraestrutura que pertencem à camada de aplicação.

3.2 Versionamento de software

O versionamento de software é uma prática essencial no desenvolvimento de sistemas web, permitindo que os desenvolvedores gerenciem versões diferentes do código-fonte ao longo do tempo de forma que seja possível navegar, adicionar e excluir versões. O controle de versão é altamente relevante para a integração contínua, na qual todas as mudanças são integradas de forma regular ao repositório principal para evitar conflitos complexos no futuro (FOWLER, 2018). Essa prática é fundamental para garantir a integridade e a rastreabilidade do software, permitindo reverter alterações, auditar modificações e colaborar de forma eficaz em equipes distribuídas.

3.2.1 Git

O Git é um sistema de controle de versão distribuído que permite aos desenvolvedores gerenciar as alterações no código de maneira eficiente elaborando um versionamento

consistente. A ferramenta Git oferece uma abordagem descentralizada, no qual cada desenvolvedor possui uma cópia completa do repositório, incluindo todo o histórico de mudanças (LOELIGER JON; MCGUIRE, 2012). Logo, isso permite o desenvolvimento contínuo do trabalho sem a necessidade de conexão com a internet, o que facilita a junção de diferentes versões do código assim que elas forem expostas a um repositório remoto, como o GitHub.

3.2.2 GitHub

O GitHub é uma plataforma que utiliza o Git como sistema de controle de versão, oferecendo uma conjunto de funcionalidades que facilitam o desenvolvimento colaborativo de software, como um ponto de acesso global em nuvem e funcionalidades para revisão de novas versões do sistema. O GitHub não só facilita a integração de contribuições entre vários desenvolvedores, mas também melhora a qualidade do software através de revisões e *feedbacks* contínuos, com suas funcionalidades de *pull requests* e revisões de código (FOWLER, 2018).

Além disso, o GitHub oferece um conjunto de ferramentas para automação e gestão de projetos, como a sua extensão GitHub Actions, que permite a configuração de pipelines de CI/CD diretamente nos repositórios. Isso possibilita a automação de testes, *builds* e *deploys* de aplicações web.

3.3 Bancos de dados

Os bancos de dados são ferramentas imprescindíveis em sistemas de software, uma vez que são responsáveis pelo armazenamento, organização e recuperação de informações de maneira eficiente e segura. Eles são projetados para lidar com grandes volumes de registros e suportar operações complexas, garantindo a integridade e a disponibilidade das informações. Existem vários modelos de bancos de dados, contudo, para este trabalho, foi escolhido o SGBD PostgreSQL, devido às suas vantagens e recursos, que serão abordados a seguir.

3.3.1 SQL

A linguagem SQL (*Structured Query Language*) oferece uma interface robusta para a manipulação de registros, sendo fundamental para a administração de bancos de dados relacionais (ELMASRI; NAVATHE, 2017). Nesse contexto, ela permite que um sistema realize uma ampla variedade de operações, como inserir, consultar, atualizar e excluir dados, além de definir e modificar a estrutura das tabelas. O SQL também é conhecido por sua capacidade de expressar consultas complexas que podem unir instâncias de múltiplas tabelas, aplicar filtros e realizar cálculos agregados, mantendo, ao mesmo tempo, a performance e a integridade das informações.

3.3.2 ACID

ACID é um acrônimo que define as propriedades essenciais para garantir a confiabilidade das transações em bancos de dados relacionais: Atomicidade, Consistência, Isolamento e Durabilidade. A atomicidade assegura que todas as operações de uma transação sejam concluídas com sucesso ou revertidas em casos de falhas, preservando a integridade dos registros. A consistência garante que uma transação leve o banco de um estado válido a outro, mantendo as regras de integridade. O isolamento impede que transações concorrentes interfiram entre si, assegurando que cada transação seja executada como se fosse a única em execução no contexto das consultas. Por último, a durabilidade garante que os resultados de uma transação sejam permanentemente aplicados, mesmo em caso de falhas no sistema.

As propriedades ACID são cruciais para garantir que os sistemas de banco de dados operem de forma confiável, mesmo sob condições adversas, protegendo as informações contra inconsistências e perdas (GRAY; REUTER, 1993). Esses fatores são fundamentais para o controle dos registros em sistemas de banco de dados, especialmente em ambientes com múltiplas transações simultâneas.

3.3.3 Transações

As transações em bancos de dados são conjuntos de operações que devem ser executadas como uma única unidade. Elas são essenciais para manter a integridade dos registros em consultas nas quais múltiplos processos podem acessar e modificar as informações. Uma transação deve cumprir todas as propriedades ACID para garantir que o banco permaneça em um estado consistente.

3.3.4 PostgreSQL

O sistema de gerenciamento de banco de dados objeto relacional (ORDBMS) PostgreSQL é um software de código aberto que se destaca por sua robustez e alta confiabilidade. Essa ferramenta é amplamente utilizada em sistemas onde a integridade das informações e a aderência às propriedades ACID são de grande importância, devido ao seu suporte a uma ampla variedade de tipos de dados e funcionalidades avançadas, como a criação de tipos de dados personalizados, operadores e funções.

O PostgreSQL combina a rigidez necessária para aplicações críticas com a flexibilidade de um sistema altamente extensível, o que o torna ideal para uma ampla gama de aplicações, desde pequenas empresas até grandes corporações (MOMJIAN, 2015). Dessa forma, a escolha do PostgreSQL é uma excelente opção para projetos em que os desenvolvedores necessitam de um banco de dados capaz de se adaptar a diferentes cenários e necessidades.

3.4 Back-end

Em termos de aplicações web, o *back-end* é a parte do sistema responsável por gerenciar a lógica de negócios, o acesso aos dados e a segurança da aplicação. Esse serviço geralmente define uma API que interage com bancos de dados, clientes externos e regras que determinam o funcionamento da aplicação. No ecossistema Java, várias ferramentas e *frameworks* foram desenvolvidos para garantir a agilidade e confiabilidade na criação de servidores robustos e escaláveis. Assim, podemos citar as ferramentas do ecossistema do Spring Boot como sendo as mais utilizadas em aplicações modernas, algumas delas são o Spring Data JPA, Flyway e Spring Security.

3.4.1 Spring Boot

O Spring Boot é uma ferramenta quem tem como objetivo fornecer uma plataforma de base pronta com infraestrutura para produção que pode ser utilizada para criar uma nova aplicação Spring ou microsserviço (SPRING, 2024a). Logo, o Spring Boot aprimora o desenvolvimento de aplicações Java de grande porte baseadas no Spring Framework, permitindo que os desenvolvedores criem soluções com configurações mínimas de infraestrutura de aplicação, ganhando velocidade, eficiência e a confiabilidade da aplicação no processo de desenvolvimento.

3.4.2 Spring Data JPA

O Spring Data JPA disponibiliza métricas para a criação de repositórios de dados baseados no JPA (*Java Persistence API*), eliminando a necessidade de grande parte do *boilerplate code* que seria necessário para implementar um repositório completo (SPRING, 2024b). O Spring Data JPA oferece abstrações para o acesso a dados de forma rápida e bem estruturada, utilizando o padrão JPA. Nesse sentido, o benefício se aplica ao desenvolvimento de repositórios de dados de maneira eficiente, segura e flexível, o que permite consultas seguras.

3.4.3 Flyway

O Flyway é uma dependência de software que permite a migração e o gerenciamento das versões de banco de dados de forma automática e controlada. Essa ferramenta realiza migrações incrementais, aplicando *scripts* de migração de forma lógica e sequencial para garantir que o banco de dados esteja sempre em um estado conhecido. O Flyway permite que os desenvolvedores rastreiem, versionem e apliquem mudanças ao esquema do banco de dados de maneira automatizada e repetível (FLYWAY, 2024).

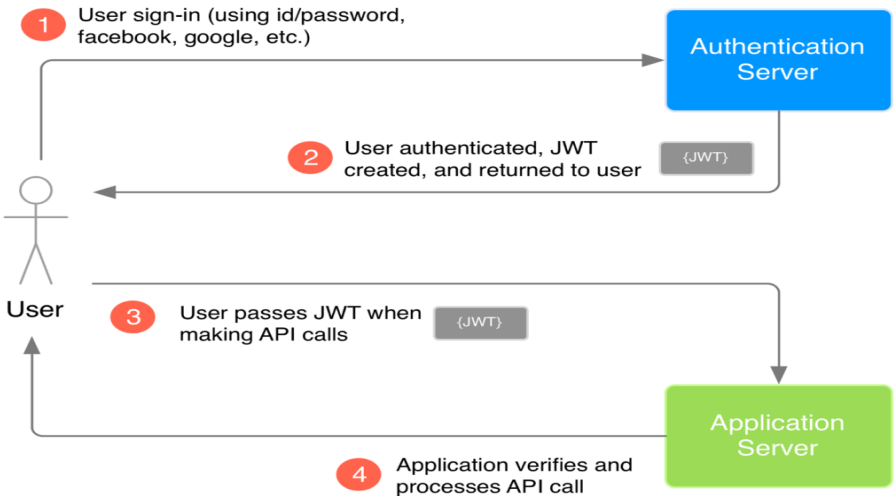
3.4.4 Spring Security

O Spring Security é um framework de segurança poderoso e altamente personalizável para aplicações Java, uma vez que oferece autenticação, autorização e proteção contra inúmeras ameaças de segurança na web. Essa dependência se destaca pelo seu nível de adaptabilidade ao projeto, bem como fornece uma série de funcionalidades altamente personalizáveis para uso, como a integração com OAuth2, JWT e a configuração de roles e permissões baseadas em anotações Java (SPRING, 2024c). Assim, o Spring Security é amplamente utilizado para prover segurança a APIs REST e implementar práticas modernas e eficientes de segurança.

3.4.5 JWT

O fluxo de autenticação e autorização com JWT (JSON Web Token) como mostrado na Figura 6 é uma excelente abordagem para garantir a segurança em aplicações web. Nesse processo de autenticação, o usuário envia suas credenciais (como nome de usuário, senha e permissões) para o servidor, que as valida conforme os dados persistidos e as políticas de acesso. Após essa etapa, o servidor gera um *token* JWT, caso as credenciais sejam válidas. Esse *token*, que geralmente possui um prazo de validade de algumas horas, é um objeto JSON assinado digitalmente com uma chave secreta, contendo informações do usuário e outros dados relevantes. No próximo fluxo, o *token* é enviado de volta ao cliente, que o armazena localmente, geralmente em *cookies* ou no armazenamento local do navegador, possibilitando realizar novas requisições sem trafegar dados sensíveis pela rede, por meio do uso do *token* JWT (NAAR, 2021). Tanto a parte de autenticação quanto a de autorização foram implementadas diretamente no servidor, sem a necessidade de dependências externas de APIs de terceiros.

Figura 6 – Diagrama de Fluxo de Autenticação e Autorização com JWT



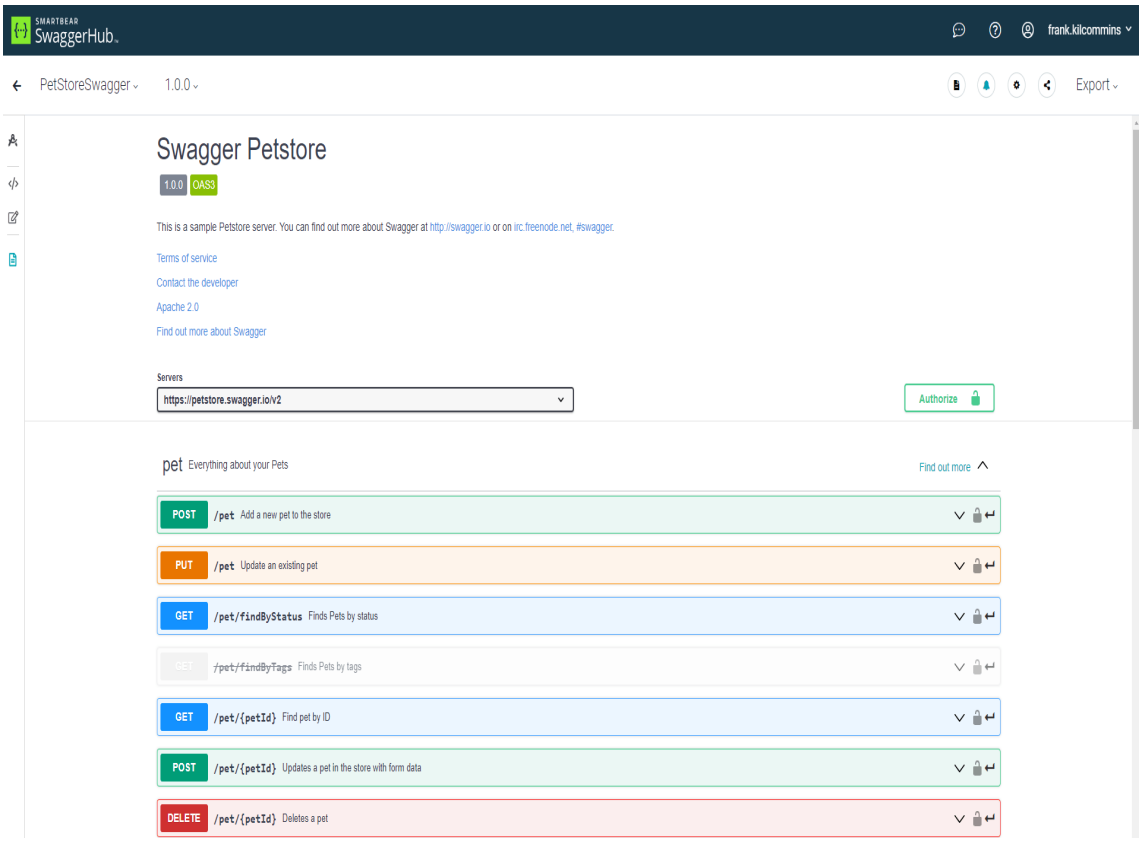
Fonte: (LUIZTOOLS, 2023)

3.4.6 Swagger

O Swagger é um conjunto de ferramentas definidas conforme a especificação do Open API, que visa facilitar a criação, visualização e testes de APIs. A interface gráfica do Swagger, como mostrado na Figura 7, permite que desenvolvedores e *stakeholders* interajam diretamente com a API, através de sua documentação prática e objetiva (SWAGGER, 2024). Com ele, é possível gerar uma interface gráfica que permite a exploração de uma API diretamente no navegador.

Além de beneficiar o fluxo de desenvolvimento do projeto, a documentação das APIs também desempenha um papel essencial na comunicação com *stakeholders*, funcionando como um contrato claro entre cliente e servidor e estabelecendo expectativas transparentes sobre o comportamento e as funcionalidades da API.

Figura 7 – Exemplo de UI do Swagger



Fonte: (SWAGGER, 2023)

3.5 Front-end

O *front-end* é responsável pela criação da interfaces de usuário e da experiência visual de um sistema. Essa camada lida diretamente com a apresentação de dados ao usuário final e a interação com os elementos da interface para efetivar ações nesses dados.

Ferramentas e *frameworks* modernos permitem que os desenvolvedores criem interfaces ricas e interativas, com uma ênfase crescente em desempenho e usabilidade. A adoção de tecnologias como Single Page Applications (SPA), *frameworks* como Angular, bibliotecas de componentes como PrimeNG, e ícones escaláveis como Font Awesome, tem transformado o desenvolvimento *front-end*, padronizando a criação de interfaces dinâmicas, responsivas e eficientes.

3.5.1 SPA

Single Page Applications (SPAs) são aplicações web que carregam uma única página HTML e atualizam dinamicamente o conteúdo conforme o usuário interage com a aplicação, sem a necessidade de recarregar a página inteira. Essa abordagem melhora a experiência do usuário ao proporcionar uma navegação mais rápida e fluida, semelhante à de aplicativos *desktop*. SPAs utilizam *frameworks* como o Angular para gerenciar o estado da aplicação e manipular o DOM de forma eficiente, garantindo que apenas as partes relevantes da página sejam atualizadas, e assim reduzindo a latência, bem como melhorando a usabilidade final.

Uma das principais vantagens das SPAs é a capacidade de oferecer uma experiência mais rica ao usuário, minimizando o tempo de carregamento e a latência percebida por meio de práticas de *lazy loading*. No entanto, SPAs também apresentam desafios, como a necessidade de gerenciar rotas e estados complexos do cliente, além de garantir a usabilidade e o desempenho em dispositivos móveis. Essas questões são tratadas com o uso de técnicas avançadas, como a renderização do lado do servidor (SSR) e a pré-renderização, que ajudam a mitigar os problemas associados à arquitetura SPA (ANGULAR, 2024).

3.5.2 Angular

O Angular é um *framework* de desenvolvimento *front-end* mantido pela Google, que permite a criação de aplicações web dinâmicas e robustas. Ele é amplamente utilizado para desenvolver SPAs, oferecendo um conjunto completo de ferramentas para a construção de interfaces ricas e escaláveis. O Angular adota uma arquitetura baseada em componentes, na qual a interface do usuário é dividida em pequenas partes reutilizáveis, cada uma com sua própria lógica e apresentação.

Além disso, esse *framework* inclui funcionalidades como injeção de dependência, roteamento, e comunicação com APIs RESTful, o que garante eficiência no desenvolvimento de aplicações complexas. Outra característica importante do Angular é o seu ecossistema robusto, que inclui uma vasta coleção de bibliotecas e ferramentas para testes, animações e traduções para o sistema.

3.5.3 PrimeNg

O PrimeNG é uma biblioteca de componentes de interface de usuário para Angular, que fornece uma ampla gama de componentes prontos para uso, como tabelas, formulários, gráficos e menus. A biblioteca segue os princípios de design responsivo e acessível, garantindo que as interfaces sejam adaptáveis e flexíveis. O PrimeNG é uma das bibliotecas de componentes UI mais completas para Angular, com uma vasta coleção de componentes que aceleram o desenvolvimento de aplicações *front-end* (PRIMETEK, 2024).

Outras vantagens do uso dessa dependência incluem componentes prontos para uso, com lógica individual que permite personalização conforme as preferências do desenvolvedor, o que é essencial para atender às necessidades específicas do design do sistema. Além disso, o PrimeNG é compatível com vários temas de interface do usuário, facilitando a aplicação de diferentes estilos visuais e a manutenção dos componentes de maneira consistente em toda a aplicação. A facilidade de integração e a vasta gama de funcionalidades fazem do PrimeNG uma escolha ideal para desenvolvedores que buscam eficiência e qualidade em seus projetos de Angular (PRIMETEK, 2024).

3.6 Testes de software

Os testes de software garantem que o código funcione conforme o esperado mediante mudanças relevantes no sistemas, assim como os requisitos principais e a qualidade geral do sistemas seja mantida. Existem diversas ferramentas e bibliotecas para testar diferentes segmentos do software, desde a interface de usuário até os serviços *backend*. Entre essas ferramentas, destacam-se o Testcontainers, para testes com contêineres, o REST Assured para testes de APIs RESTful; e Jest para testes de front-end com Javascript. Essas ferramentas ajudam a automatizar o processo de testes, aumentando a eficiência durante o seu desenvolvimento, por meio de cobertura de código e detecção de problemas regulares antes que uma versão seja lançada em produção e durante a elaboração de novas versões.

3.6.1 Testcontainers

O Testcontainers é uma biblioteca Java que permite a criação e gestão de contêineres Docker em testes automatizados. Com Testcontainers, é possível configurar ambientes de teste isolados do servidor principal, que replicam a infraestrutura de produção, como bancos de dados e filas de mensagens, encapsulando tudo em contêineres. Isso garante que os testes sejam executados em condições consistentes e que simulam o mundo real, o que aumenta a confiabilidade final do produto.

Além disso, Testcontainers facilita a execução de testes que envolvem múltiplos serviços, permitindo a orquestração de vários contêineres simultaneamente. Logo, para

simular cenários com vários fluxos paralelos isso é muito importante, tendo em vista que a infraestrutura real pode não estar disponível. Testcontainers também se integra bem com outras ferramentas de teste, como JUnit e Spring Boot, o que simplifica a configuração e a execução de testes automatizados em projetos Java (TESTCONTAINERS, 2024).

3.6.2 REST Assured

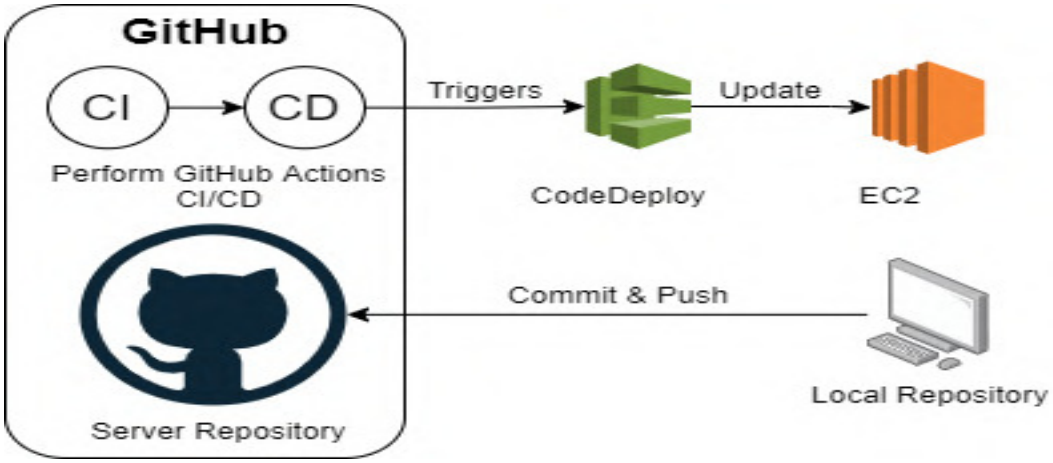
O REST Assured é uma biblioteca Java para testes de integração de APIs REST, que permitem validar os fluxos dessas APIs, incluindo respostas HTTP, verificação de cabeçalhos e validação de payloads JSON, XML e YAML. O REST Assured oferece uma API fluente e intuitiva para realizar testes de serviços REST, facilitando a escrita de testes automatizados que verificam o comportamento e a funcionalidade de APIs RESTful (ASSURED, 2024). Assim sendo, essa ferramenta é altamente viável para um projeto escalável, no qual é necessário testar autenticação, manipulação de cabeçalhos de requisição, respostas e configuração de parâmetros de consulta.

3.7 Devops

DevOps é um conjunto de práticas que permite integrar o desenvolvimento de software a operações de TI para melhorar a colaboração e a automação de processos ao longo do ciclo de vida do sistema (SMITH, 2024). Nessa seção, serão abordadas as principais ferramentas utilizadas para prover o sistema desenvolvido na web.

Na Figura 8, é possível observar a integração entre ferramentas como o GitHub e os serviços da AWS (Amazon Web Services), que são fundamentais para a implementação de práticas DevOps e para a escalabilidade da aplicação, permitindo, assim, o provisionamento de infraestrutura em nuvem e a disponibilidade da aplicação de forma confiável e eficiente.

Figura 8 – Diagrama Representacional de uma Pipeline com GitHub Actions



Fonte: (DATA, 2023)

3.7.1 Contêineres

Contêineres são uma forma padronizada de empacotar código, configurações e dependências para que o software possa ser executado de forma rápida e confiável entre diferentes ambientes de computação (DOCKER, 2024). Logo, isso resolve o problema de inconsistências entre ambientes de desenvolvimento, teste e produção, assegurando que o sistema não possua grandes variações em seu comportamento conforme as mudanças de ambiente. Nesse sentido, eles proporcionam que a aplicação seja separada corretamente do sistema operacional e que eles se comuniquem da maneira correta, possibilitando que o software seja executado em qualquer máquina virtual ou física que tenha o Docker instalado, por exemplo. Isso é muito utilizado para hospedagem de servidores devido a segurança, confiabilidade e portabilidade.

3.7.2 Docker

O Docker permite que você construa, teste e implante aplicações rapidamente, em qualquer ambiente, usando contêineres (DOCKER, 2024). Assim, o Docker é uma plataforma que automatiza o processo de criação, implantação e gerenciamento de contêineres, por meio de imagens que podem ser compartilhadas e executadas em qualquer sistema que suporte Docker, eliminando as diferenças entre os ambientes e garantindo que a aplicação funcione da forma esperada em máquinas distintas.

Além disso, o Docker também facilita a gestão de ambientes complexos, permitindo que múltiplos contêineres sejam orquestrados e gerenciados em escala conforme a quantidade de acessos. Além disso, com ferramentas como o Docker Compose, é possível definir e executar aplicações multi-contêineres de forma eficiente, enquanto ferramentas como Docker Swarm e Kubernetes viabilizam a orquestração e escalabilidade de contêineres. Portanto, o Docker se tornou uma ferramenta essencial para a implementação de pipelines de CI/CD, permitindo a construção e o teste automatizados de software em ambientes isolados e replicáveis (DOCKER, 2024).

3.7.3 AWS

A Amazon Web Services (AWS) é uma plataforma completa de serviços de computação em nuvem nos quais permite trabalhar com práticas de monitoramento de aplicações, instâncias de servidores e bancos de dados em nuvem com um alto padrão de segurança. A AWS possibilita que os sistemas escalem com planejamento e eficiência e que os seus usuários possuam custos apenas pelo serviços alocados utilizando infraestrutura global de *datacenters*. A integração com os serviços como Amazon VPC, S3, RDS, ECR, ECS, IAM e EC2 permite que os sistemas sejam hospedados em um ambiente excelente quanto a práticas de segurança de redes e disponibilidade de servidores.

3.7.3.1 VPC

A Amazon Virtual Private Cloud (VPC) permite definir um ambiente de rede virtual que simula uma rede tradicional no data center, mas com os benefícios da escalabilidade e flexibilidade da infraestrutura AWS (AWS, 2024). Uma vez configurada, a VPC oferece uma gama de recursos para o ambiente de rede, incluindo a seleção de intervalos de Protocolo de Internet (IP), a criação de sub-redes, configuração de tabelas de roteamento e *gateways* de rede.

3.7.3.2 S3

A Amazon Simple Storage Service (S3) é um serviço de armazenamento de objetos que oferece escalabilidade, disponibilidade e segurança de dados. O S3 viabiliza o armazenamento e proteção de dados para uma variedade de casos de uso, como backup e restauração, arquivamento, aplicativos móveis e web. Com o S3 é possível ter uma infraestrutura durável, segura e altamente escalável para armazenar e recuperar qualquer quantidade de dados, a qualquer momento, de qualquer lugar na web (AWS, 2024).

3.7.3.3 RDS

O Amazon Relational Database Service (RDS) facilita a configuração, conexão, operação e escalabilidade de bancos de dados relacionais na nuvem, com custos previsíveis e flexibilidade de escalabilidade (AWS, 2024). Com isso, é possível ter acesso a hardware, configuração de banco de dados, aplicação de *patches* e backups. O RDS suporta vários motores de banco de dados, incluindo o PostgreSQL, no qual foi utilizado no projeto.

3.7.3.4 IAM

Um AWS Identity and Access Management (IAM) permite acessar e gerenciar de forma segura os serviços e recursos da AWS. Dessa maneira, é possível criar e gerenciar usuários e grupos, definir permissões para esses usuários e assim intermediar o acesso a infraestrutura configurada para hospedagem do servidor.

3.7.3.5 EC2

O Amazon Elastic Compute Cloud (EC2) é um serviço de que oferece capacidade escalável para sistemas web, permitindo que os usuários criem máquinas virtuais altamente personalizáveis. Essas instâncias são essenciais dentro da Amazon Web Services (AWS), uma vez que elas proporcionam aos desenvolvedores a liberdade de configurar e gerenciar seus ambientes de produção.

O EC2 destaca-se pela sua eficiência ao lidar com variações na carga de trabalho de acesso no servidor por meio de escalabilidade horizontal, em que instâncias adicionais

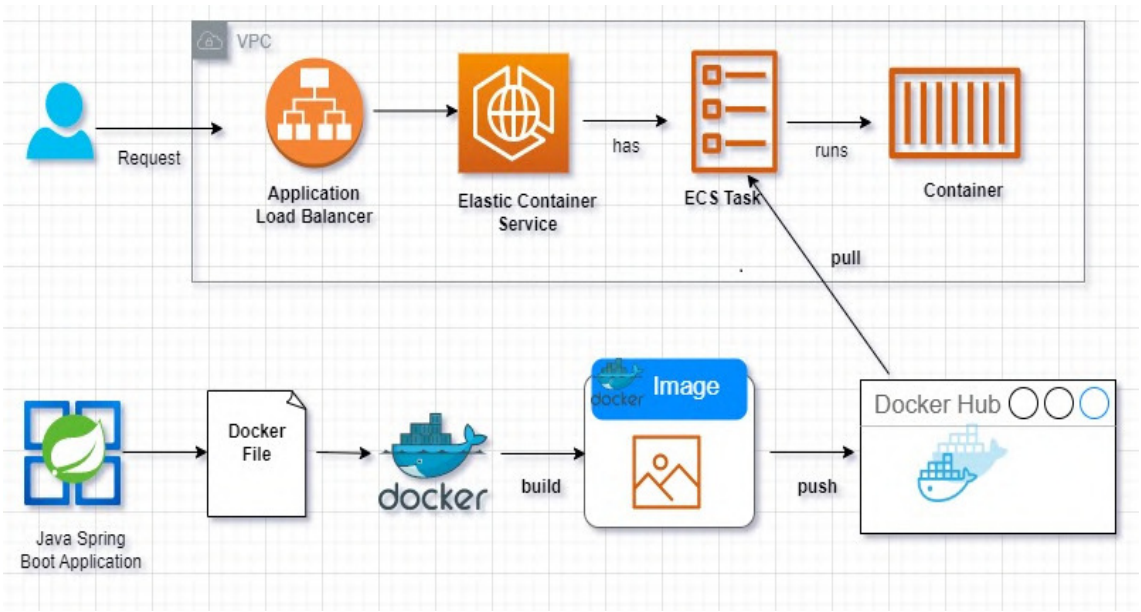
podem ser facilmente ativadas ou desativadas conforme o aumento ou diminuição da demanda (AWS, 2024). Isso é indubitável para contextos nos quais existem grandes picos de acesso, bem como esse recurso garante eficiência no controle de custos, haja vista que o preço de utilização é calculado com base no tempo de utilização. Além disso, o EC2 se integra a outros serviços da AWS, oferecendo funcionalidades como balanceamento de carga, escalonamento automático e opções robustas de segurança.

O EC2 inclui recursos de grande importância em aplicações corporativas, como:

- **Instâncias sob demanda**: Pagamento baseado no uso, que é ideal para cargas de trabalho com demandas imprevisíveis.
- **Instâncias reservadas**: Descontos vantajosos para usuários que fazem uso de planos de uma instância específica por períodos pré-determinados.
- **Instâncias *spot***: Alternativas de baixo custo para aplicações que precisam ser interrompidas sem impacto significativo.

Com essa combinação de recursos, o EC2 se posiciona como uma ótima opção para serviços que sempre ficaram ativos.

Figura 9 – Arquitetura Completa com hospedagem na AWS



Fonte: (TIP, 2023)

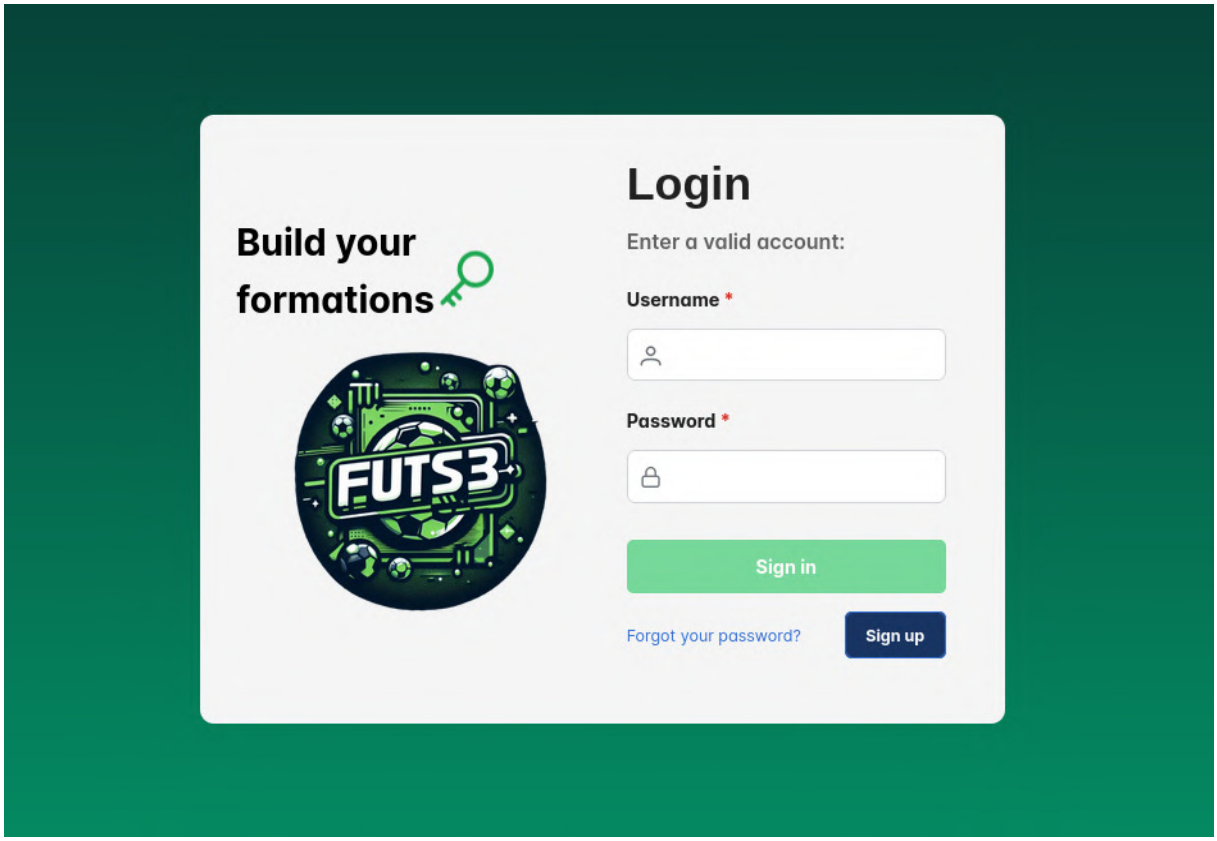
4 Resultados

Este capítulo apresenta os resultados obtidos com o desenvolvimento da aplicação. Nele serão apresentadas as principais funcionalidades oferecidas, assim como a experiência na perspectiva do usuário ao interagir com o sistema. A seguir, serão detalhadas as etapas do fluxo de uso, com base nas interfaces visuais e nos componentes desenvolvidos no *front-end* e integrados ao *back-end* da plataforma. Ao longo do texto, serão apresentados dados meramente ilustrativos, que não se equivalem à realidade, mas têm o objetivo de exemplificar alguns dos principais fluxos do Futs3.

4.1 Tela de Login da Aplicação

A tela de login da aplicação Futs3, ilustrada na Figura 10, é a primeira parte do sistema acessada pelos usuários. Nesta interface, o usuário pode inserir suas credenciais para acessar a plataforma, que, caso estejam corretas, o redireciona para a página de modos de jogo.

Figura 10 – Tela de Login da Aplicação



Fonte: Elaborado pelo autor, 2024

A princípio, é possível identificar algumas características principais conforme ilustrado na Figura 10, como:

- ***Logo e Slogan* da Plataforma:** O logo do Futs3, acompanhado pelo slogan "Build your formations", reforça o objetivo principal da plataforma, conforme os objetivos especificados no projeto, que incluem a personalização de formações e flexibilidade.
- ***UI/UX*:** Durante a elaboração do *design*, optou-se por utilizar cores que contrastam bem, com o objetivo de tornar a interface intuitiva e visualmente simples, para que mesmo usuários com pouca experiência possam navegar e utilizar o sistema sem maiores dificuldades.
- **Campos de *Login*:** Esta interface solicita as informações de nome de usuário e senha, bem como possui ícones associados que indicam claramente a função de cada campo, ajudando na orientação visual do usuário. Além disso, possuem asteriscos indicando a obrigatoriedade do campo.
- **Ações Disponíveis:**
  - **Entrar:** Ao inserir suas credenciais, o usuário pode acionar o botão de "Sign in"(Entrar). Caso estejam corretas, ele será direcionado à seção de jogadores.
  - **Cadastre-se:** A opção "Sign up"(Cadastrar-se) é oferecida para novos usuários. Através dessa opção, eles podem criar uma conta na plataforma e começar a se beneficiar dos recursos disponibilizados.
  - **Esqueci minha senha:** Um link "Forgot your password?"está disponível para os usuários que possam ter esquecido sua senha, permitindo uma recuperação rápida via e-mail. Para essa etapa, é muito importante que o usuário tenha preenchido corretamente o campo de *e-mail*, utilizando um endereço individual e ativo, uma vez que o processo de recuperação de senha ocorrerá por meio desse *e-mail*.

De acordo com as ações apresentadas, é possível identificar os eventos que são disparados ao *Back-End* à medida que elas ocorrem.

4.1.1 O Processo de Autenticação das Credenciais durante o *Login*

O fluxo de autenticação e autorização é fundamental para garantir o acesso com segurança às funcionalidades da plataforma. Esse processo envolve mecanismos para identificar um usuário válido no sistema, proteger dados sensíveis e prevenir acessos não autorizados.

- **Entrada de Credenciais:** Quando um usuário insere suas informações de login, essas credenciais são enviadas de forma criptografada para o servidor. Se os dados forem válidos, o servidor gera e retorna um token JWT (JSON Web Token). Esse *token* será utilizado em todas as próximas requisições ao servidor, permitindo que esse acesse informações como nome de usuário, nível de autorização no sistema, entre outros dados relevantes.

- **Verificação de Credenciais:** Ao receber as credenciais do usuário, o servidor realiza a verificação das informações recebidas. Primeiro, ele consulta o banco de dados utilizando o *username* fornecido. Se o usuário existir, o servidor obtém o *hash* da senha armazenada e compara com o *hash* da senha enviada pelo *front-end* durante o login do usuário. Essa comparação é feita utilizando o algoritmo de *hashing* Argon2, conhecido por seu alto padrão de segurança e eficiência. Se as senhas coincidirem, o servidor emite tanto o *access token* quanto um *refresh token*, e o cliente armazenará as novas informações referentes ao usuário, como ilustrado na Figura 11.

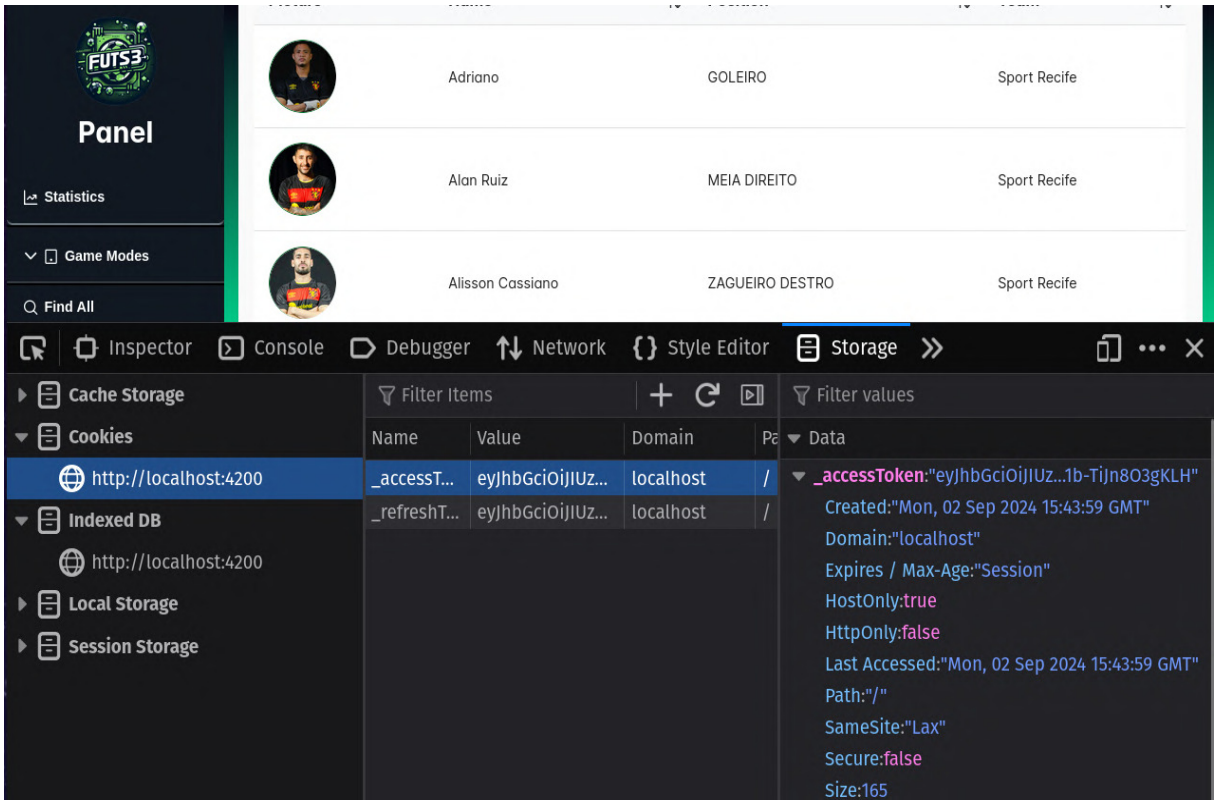


Figura 11 – Tokens Gerados pelo Servidor

Fonte: Elaborado pelo autor, 2024

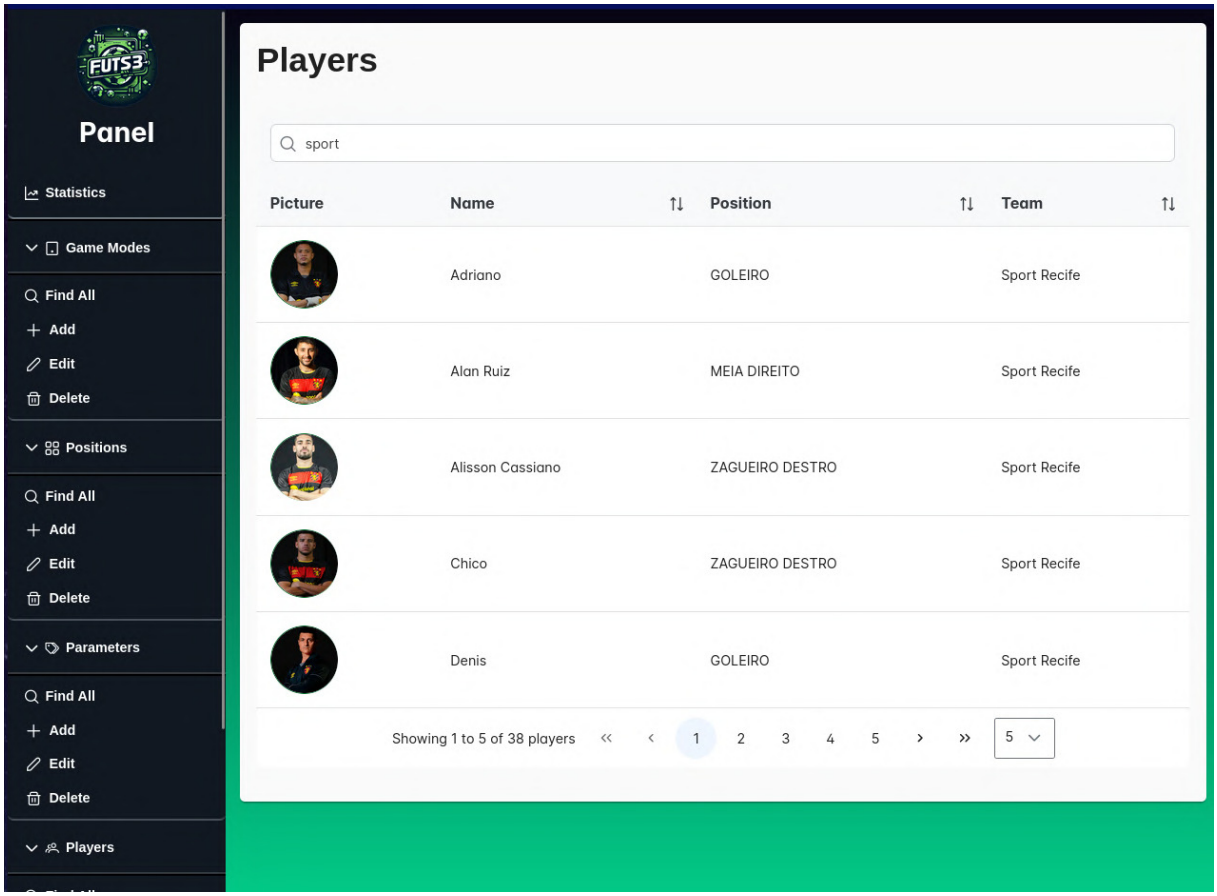
- **Gerenciamento de Sessões:** O *token* JWT possui um tempo de validade limitado, garantindo que o acesso à plataforma seja temporário e seguro. Quando o *token* expira, o cliente deve utilizar o *refresh token* para solicitar um novo sem precisar

realizar o login novamente. Caso o *refresh token* também tenha expirado, o usuário precisará efetuar o login novamente para obter novos *tokens*. Essas medidas são fundamentais para manter a segurança das sessões e proteger os dados durante o tráfego.

4.2 Navegação na Plataforma

Após realizar o *login*, o usuário é redirecionado para a tela de jogadores, conforme ilustrado na Figura 12. A navegação acontece por meio de uma *sidebar*, que provê acesso global às seções e recursos disponibilizados.

Figura 12 – *Sidebar* da Aplicação



Fonte: Elaborado pelo autor, 2024

Assim, é possível identificar os seguintes aspectos:

- **Seções Principais**: As seções principais na *sidebar* incluem "Statistics", "Game Modes", "Positions", "Parameters"e "Players". Nota-se que cada uma dessas seções está associada a conjuntos de funcionalidades conforme sua indexação. Logo, os usuários podem visualizar estatísticas, gerenciar modos de jogo, configurar posições

e parâmetros, além de observar e manipular dados de jogadores cadastrados. Esta categorização facilita a navegação, de forma que os usuários identifiquem rapidamente o módulo vigente e tenham uma navegação fluida.

- **Subitens de Menu**: Dentro de algumas das seções principais, destacam-se subitens como "Find All", "Add", "Edit"e "Delete", que oferecem um nível adicional de granularidade, uma vez que permitem aos usuários executar operações de *CRUD* (Create, Read, Update, Delete) sobre os dados.

4.2.1 Aspectos Funcionais da Interface de Navegação

A seguir, destacam-se os principais aspectos que tornam a *sidebar* um elemento central na usabilidade da plataforma:

- **Acesso Rápido**: A disposição dos itens na barra lateral possibilita o acesso rápido às funcionalidades mais frequentemente utilizadas, como a gestão de jogadores e posições. Desse modo, a inclusão de opções como "Find All", "Add", "Edit"e "Delete"diretamente na *sidebar* evita a necessidade de navegação excessiva dentro da interface, economizando tempo e tornando o processo de gerenciamento de dados mais eficiente. Assim, um usuário pode configurar parâmetros de uma posição mesmo estando na seção de jogadores, por exemplo.

- **Escalabilidade e Personalização**: A *sidebar* foi estruturada em termos de escalabilidade, ou seja, novos módulos e subitens podem ser facilmente adicionados sem maiores preocupações, como responsividade e usabilidade do sistema.

- **Seção de Estatísticas**: Esse módulo fica na parte superior do menu de navegação, tendo em vista que é o ponto principal da navegação e deve ser facilmente acessado. Assim, o usuário poderá obter informações relacionadas à análise de dados e estatísticas de desempenho dos jogadores conforme posições previamente configuradas.

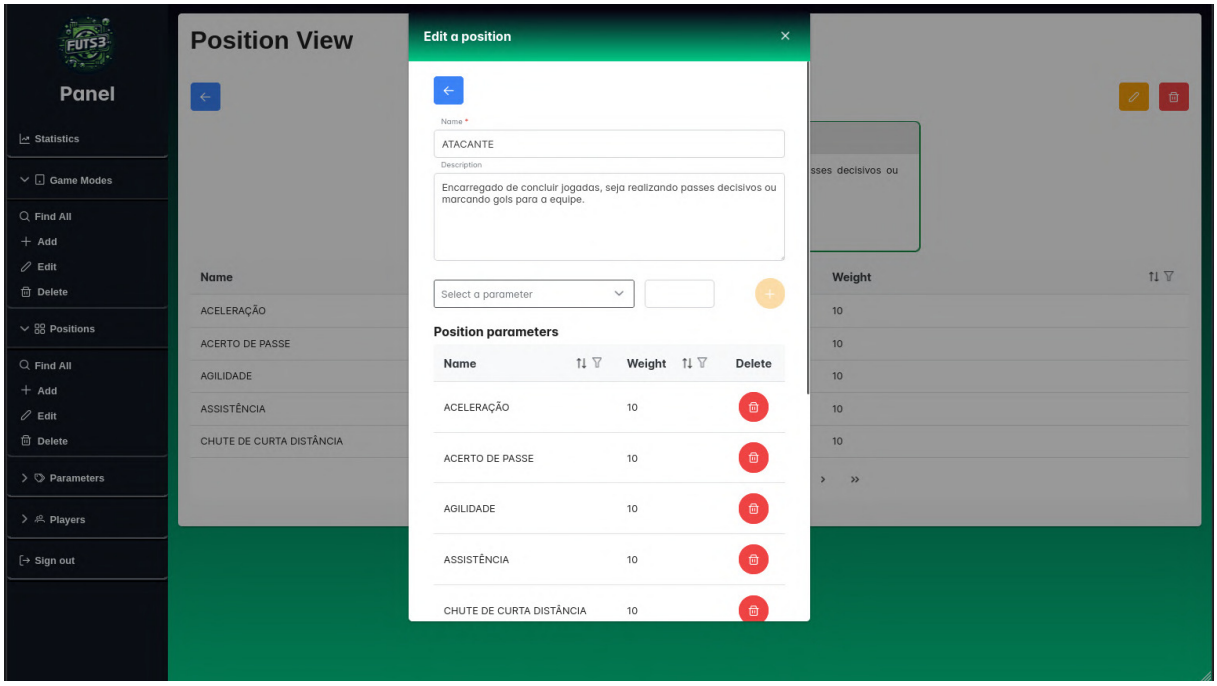
4.3 Cadastro e Análise de Posições

Mediante as funcionalidades da plataforma, destaca-se o cadastro e a análise de posições. Em cada posição cadastrada, são incluídos parâmetros personalizados, que devem ser atribuídos a essa posição antes que o sistema faça o *ranking* os melhores jogadores. Métricas como "Acerto de Passe", "Assistência"e "Resistência"são bons exemplos de parâmetros. Assim, os usuários podem customizar essas métricas de acordo com o estilo desejado na posição.

4.3.1 Edição de Parâmetros por Posição

A Figura 13 ilustra a interface de edição dos parâmetros de uma posição. Nessa tela, o usuário pode adicionar novos parâmetros e ajustar valores de peso, que identificam a importância daquele *scouting* para essa posição. Consequentemente, isso definirá os critérios de avaliação que influenciam diretamente a análise de desempenho dos jogadores conforme o *ranking* na seção de estatísticas.

Figura 13 – Edição de Parâmetros por Posição



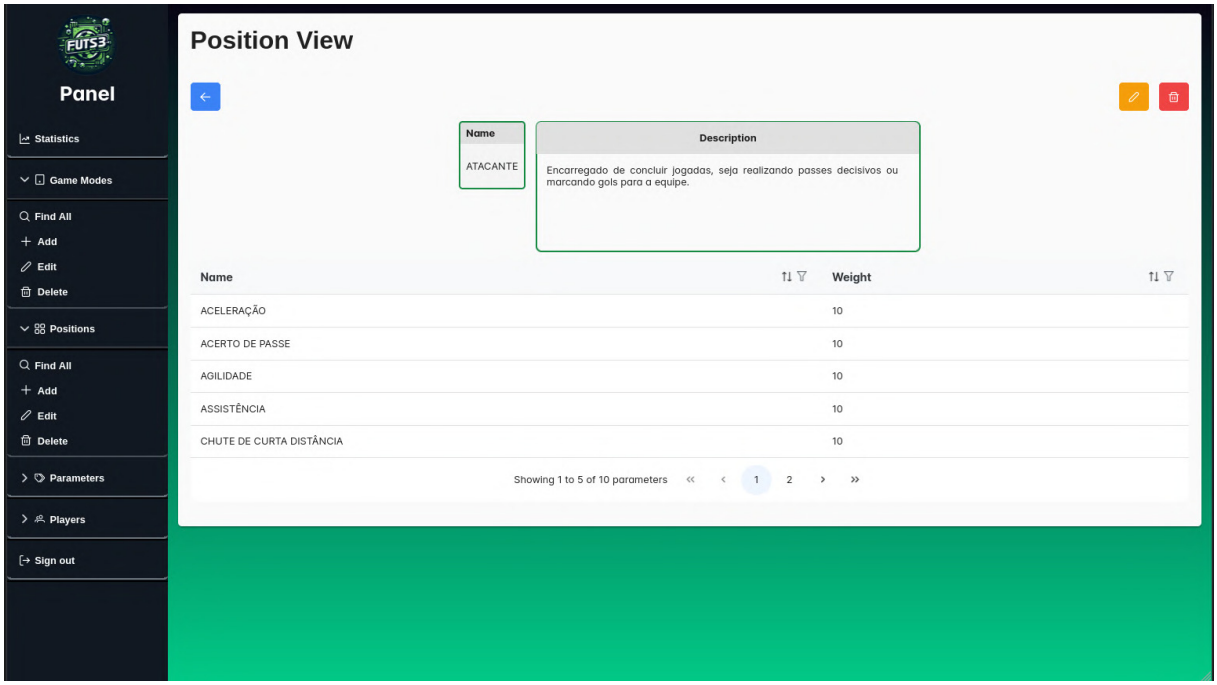
Fonte: Elaborado pelo autor, 2024

Como mostrado na Figura 13, esse nível de personalização é essencial para alinhar as métricas de desempenho com os objetivos táticos da equipe e configurados pelo usuário, oferecendo *insights* mais precisos sobre os jogadores em suas posições específicas.

4.3.2 Análise de Posições

Após o cadastro de uma posição na plataforma Futs3, os usuários podem visualizar detalhadamente todas as configurações realizadas durante o processo de criação. Conforme ilustrado na Figura 14, a interface de visualização de posição fornece uma visão clara e organizada das informações cadastradas, permitindo uma fácil verificação e, se necessário, edição dos dados.

Figura 14 – Tela de Visualização de uma Posição



Fonte: Elaborado pelo autor, 2024

4.4 Administração de Modos de Jogo

Outra funcionalidade central da plataforma é o gerenciamento de modos de jogo, acessível na seção "Game Modes"do menu, conforme mostrado na Figura 15.

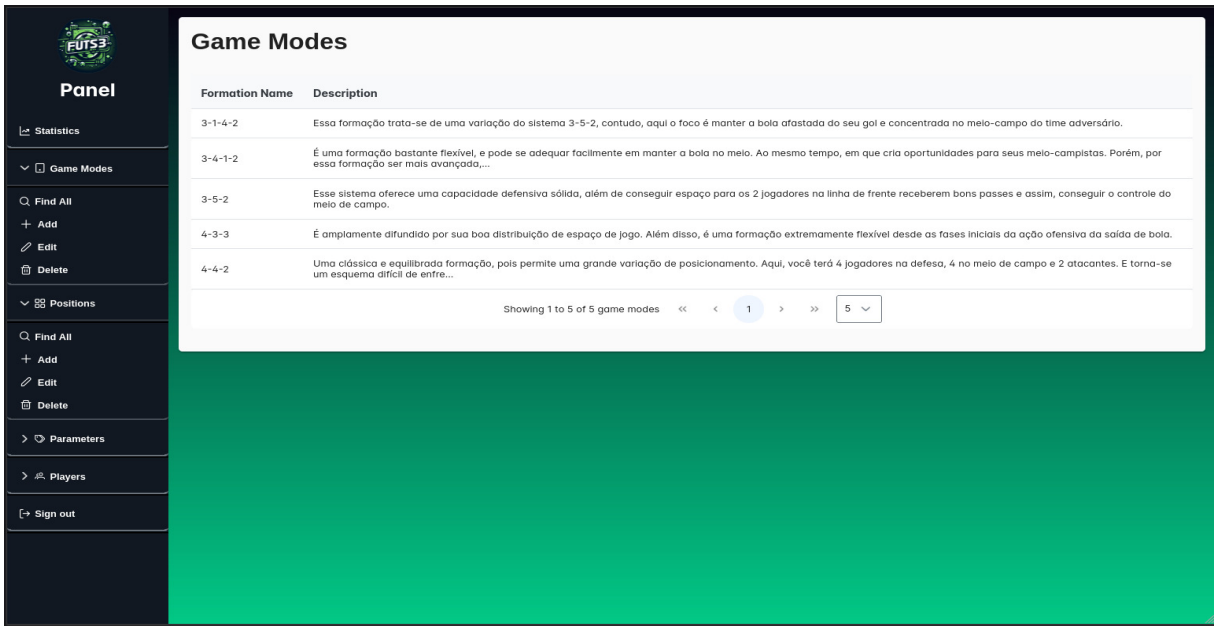


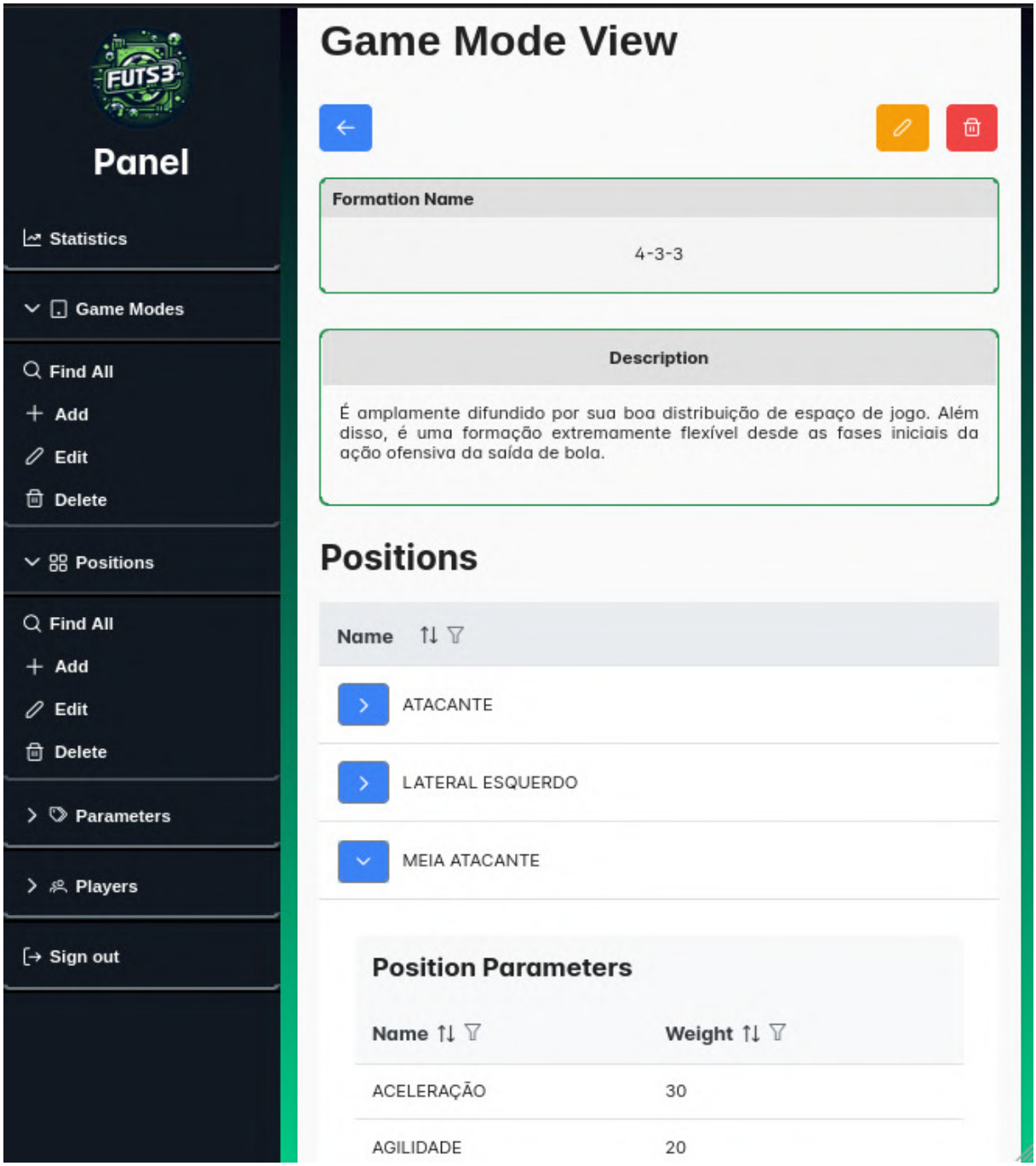
Figura 15 – Visualização de Modos de Jogo

Fonte: Elaborado pelo autor, 2024

A Figura 15 exibe a interface na qual os modos de jogo são listados, permitindo ao usuário realizar operações nas formações táticas conforme achar necessário.

Cada formação é descrita conforme a preferência do usuário, com informações como o esquema tático e as posições configuradas. Essa seção é essencial para técnicos e analistas que desejam documentar e ajustar suas táticas ao longo da temporada. A Figura 16 mostra informações acessíveis ao usuário por meio da seção de visualização.

Figura 16 – Visualização de um Modo de Jogo



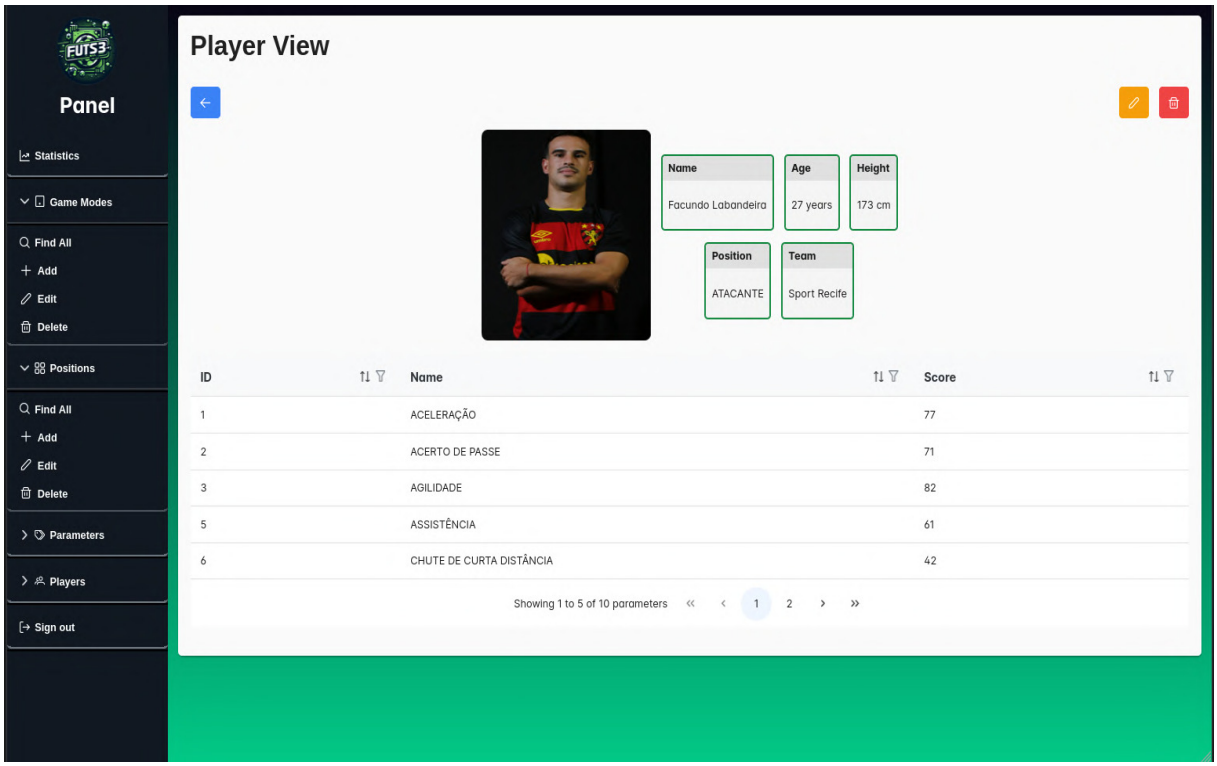
Fonte: Elaborado pelo autor, 2024

Uma vez montado o modo de jogo, o usuário tem acesso ao *ranking* disponibilizado na seção de estatísticas. Para que ele possa requisitar os melhores jogadores dado um modo de jogo e uma posição, é necessário ter os atletas devidamente cadastrados e com *scores* válidos. Caso contrário, isso resultará em alguns problemas, como o cálculo indevido por conta da falta de parâmetros.

4.5 Gerenciamento de Jogadores

Nesta seção, os usuários têm acesso a uma listagem detalhada dos jogadores, com filtros para pesquisa por nome, posição e time. O módulo de gerenciamento de jogadores permite aos usuários organizar, monitorar e avaliar o desempenho dos atletas de maneira eficiente e detalhada. Na Figura 17 é apresentada uma visão geral das principais características e funcionalidades da tela de visualização.

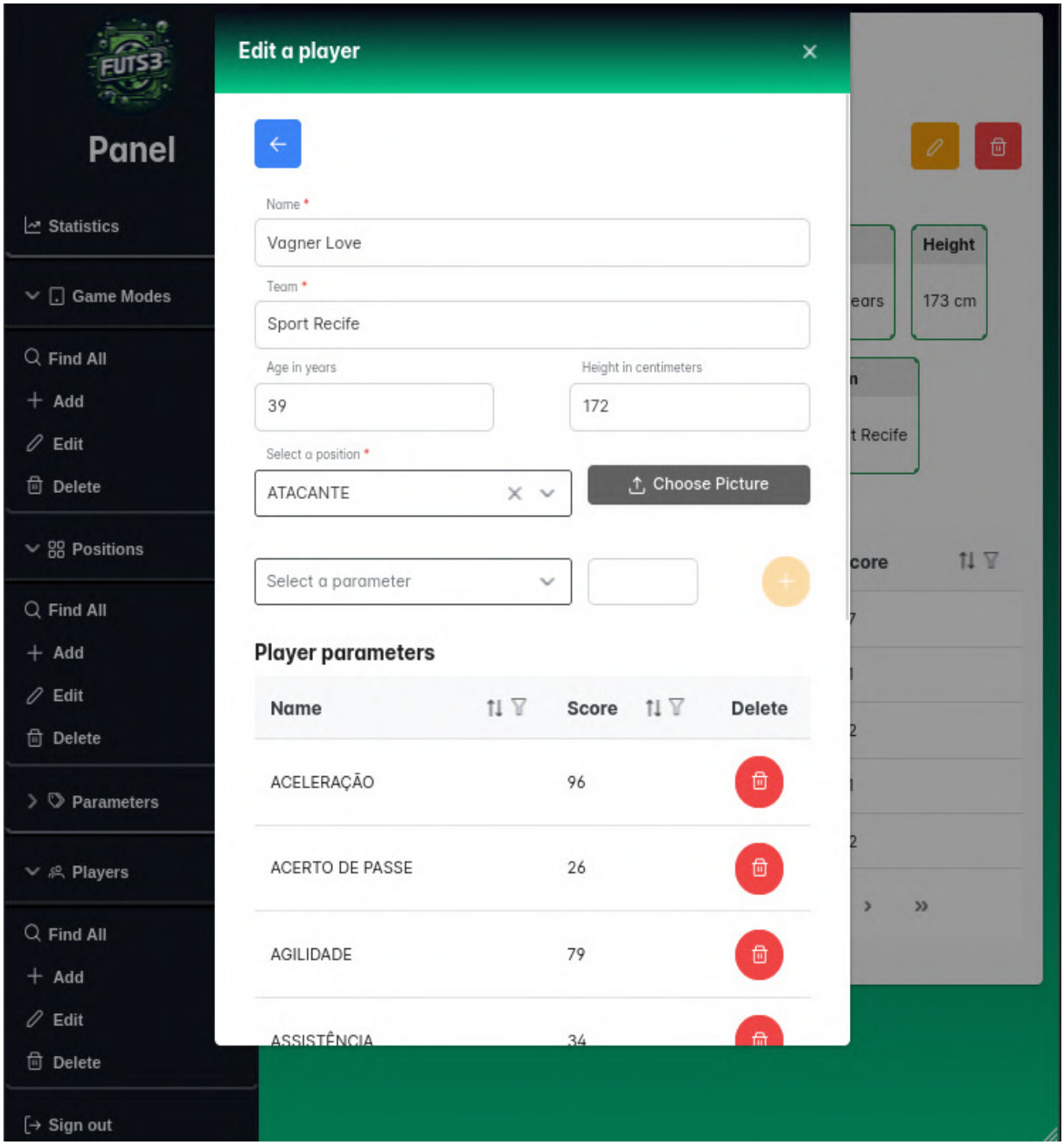
Figura 17 – Seção de Visualização de um Jogador



Fonte: Elaborado pelo autor, 2024

Além disso, o sistema provê um componente específico para cadastro de novos jogadores conforme o usuário achar oportuno para as suas estratégias. Nele, é possível inserir informações detalhadas como nome, posição, equipe e outros dados relevantes para busca e seleção. Nesse sentido, em concordância com a Figura 18, o sistema permite a edição dessas informações, possibilitando a atualização dos dados dos jogadores conforme necessário, seja para refletir mudanças na posição ou em outros aspectos relevantes.

Figura 18 – Modal de Edição dos Dados de um Jogador



Fonte: Elaborado pelo autor, 2024

4.6 Painel de Estatísticas

A seção de estatísticas é o principal módulo da aplicação. É a partir dela que os dados dos jogadores podem ser filtrados com base nas posições e parâmetros configurados pelo usuário, esses fatores são muito importantes para tomada de decisões estratégicas. Essa seção foi projetada para fornecer *insights* detalhados sobre os parâmetros de desempenho dos atletas, permitindo comparações entre jogadores e a identificação dos pontos de destaque, além de áreas de melhoria. A seguir, é apresentada uma análise detalhada e

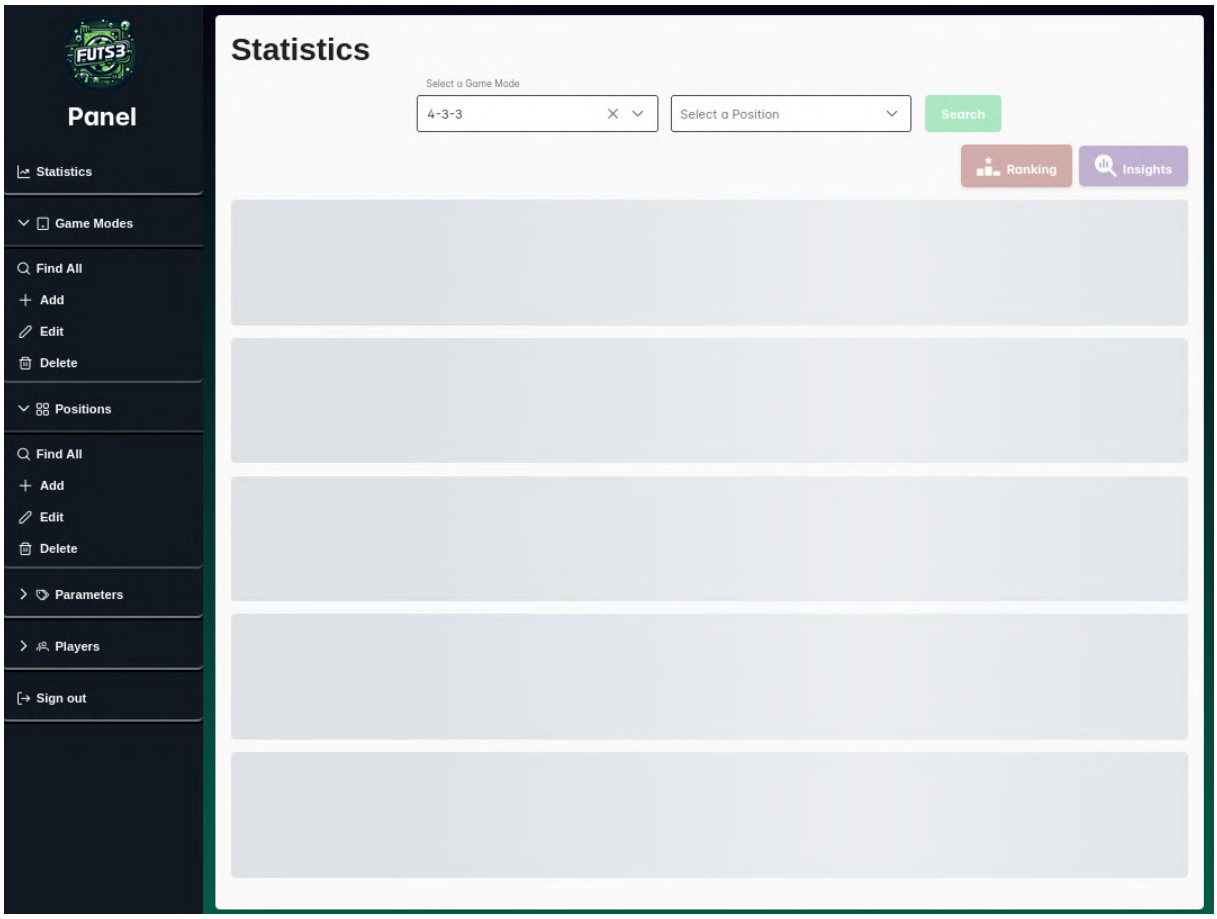
especificada desta seção, conforme as imagens fornecidas.

4.6.1 Seleção de Modo de Jogo e Posição

A seguir, será ilustrado um fluxo de seleção de um modo de jogo (4-3-3), que filtra os jogadores e parâmetros de acordo com a posição selecionada. Isso implica em uma análise específica do desempenho dentro daquele contexto tático, que é feita do lado do servidor. Logo, a capacidade de filtrar os jogadores por posição permite focar na análise de jogadores que desempenham funções semelhantes, facilitando comparações diretas e estratégias de melhoria para cada posição específica.

Conforme demonstrado na Figura 19, os usuários devem escolher o modo de jogo e a posição dos jogadores que serão analisados. Logo, os dados serão carregados e exibidos como ilustrados pela Figura 20, em função da formação tática e do *score* dos jogadores conforme os parâmetros da posição selecionada.

Figura 19 – Seleção e Carregamento do Ranking de Jogadores

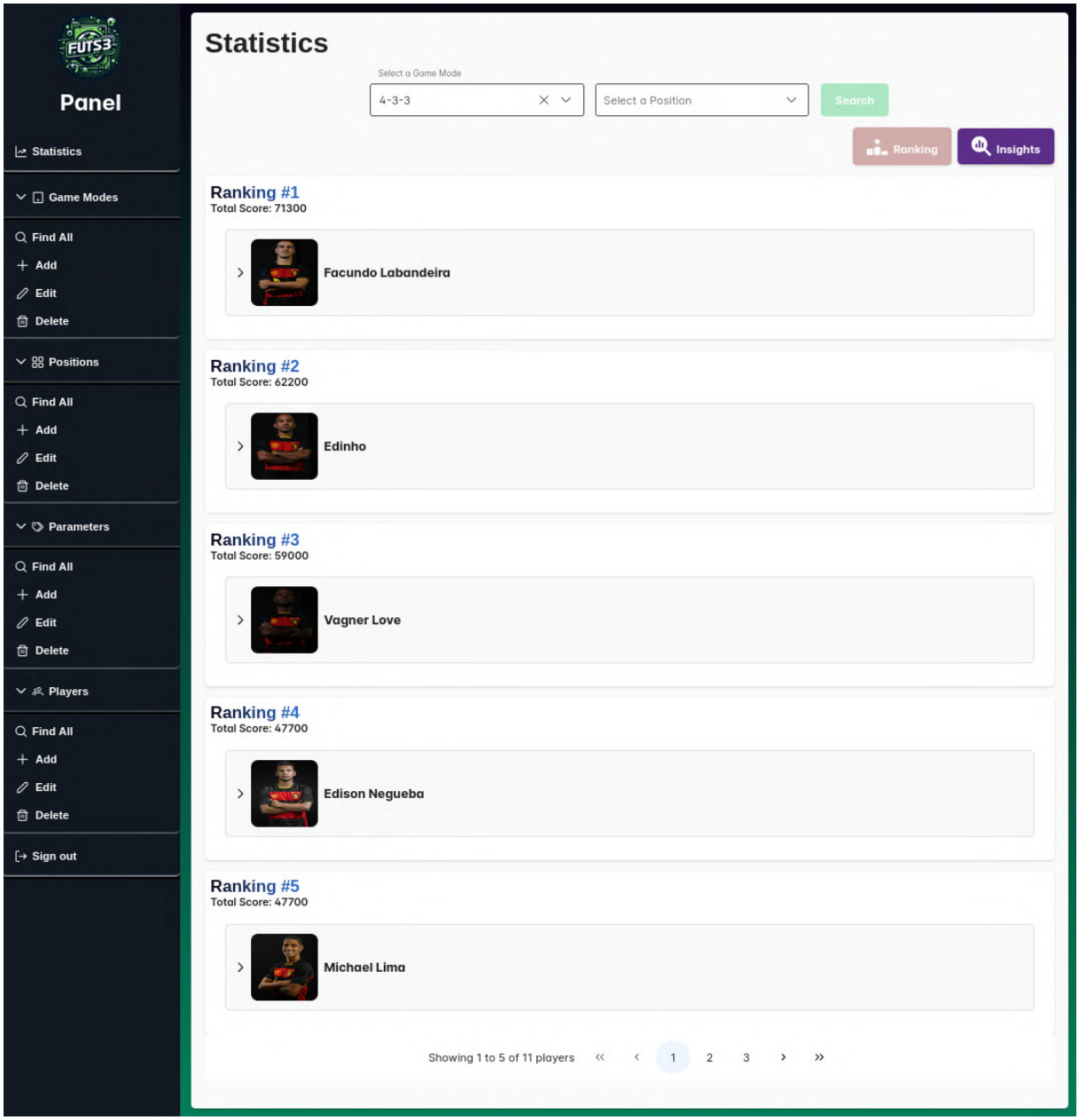


Fonte: Elaborado pelo autor, 2024

Dessa maneira, a aplicação foca na visualização geral do usuário, permitindo notar a pontuação total acumulada pelos jogadores de forma clara, ao proporcionar uma visão

comparativa direta entre eles. Caso seja necessário, a expansão do painel deve ser realizada e os *scouts* conferidos separadamente. Esse é um dos principais requisitos e objetivos estabelecidos com base nas histórias de usuário, visto que faz alusão ao potencial de inovação do projeto, bem como estabelece formalidade com o que foi proposto.

Figura 20 – Ranking de Jogadores

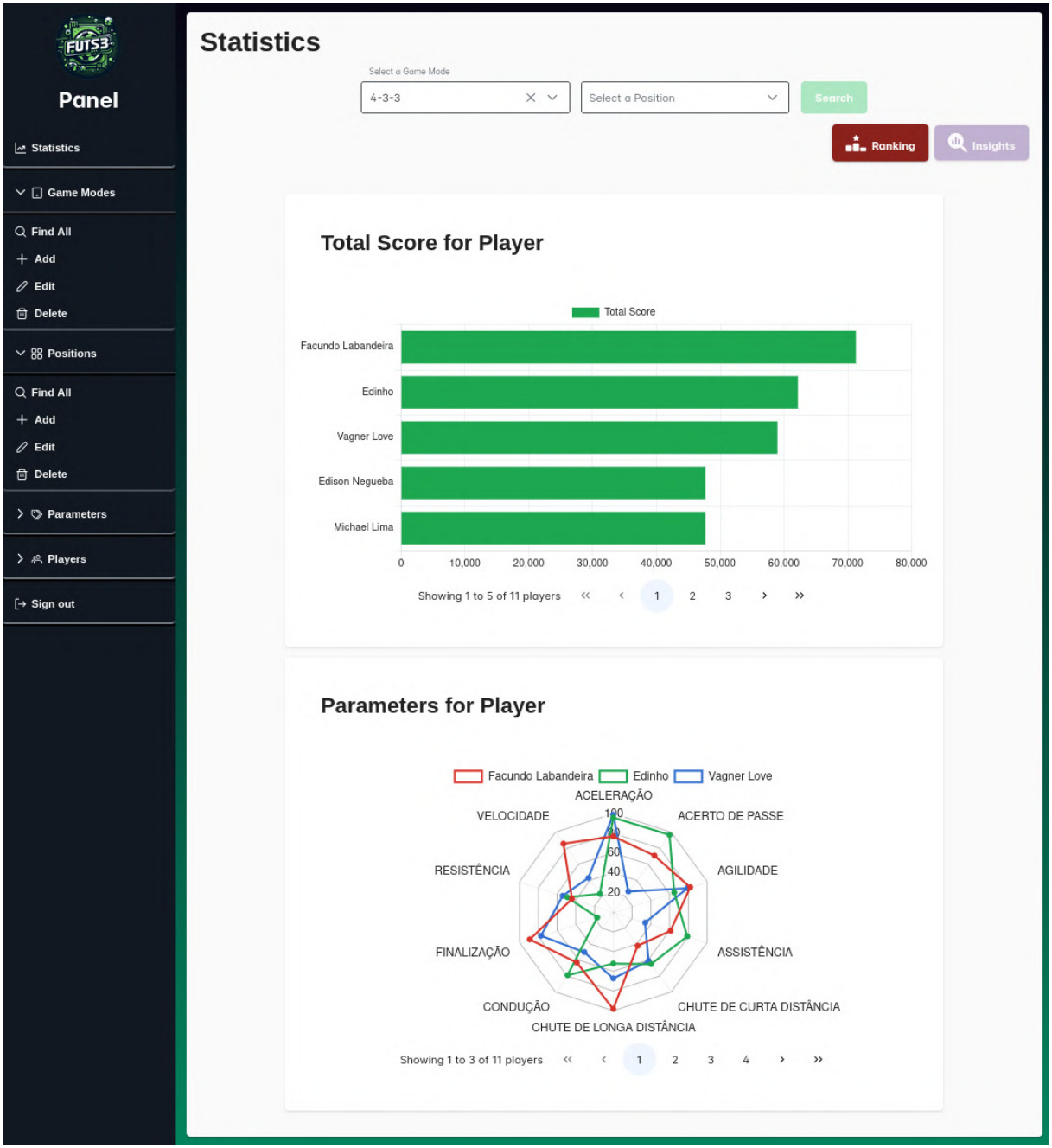


Fonte: Elaborado pelo autor, 2024

4.6.2 Análise Gráfica dos Jogadores

Após a seleção do modo de jogo e posição, a plataforma apresenta uma visualização gráfica da pontuação total dos jogadores, representada por um gráfico de barras e, para pontuações específicas, um gráfico de radar, conforme designado na Figura 21.

Figura 21 – Gráficos de *Scouts* dos Jogadores



Fonte: Elaborado pelo autor, 2024

Dadas essas observações, destacam-se os seguintes aspectos visuais e funcionais sobre os elementos:

- **Gráfico de Barras**: É nesse gráfico que a comparação direta entre os jogadores deve ser realizada, uma vez que é exibida a pontuação total acumulada por cada jogador em relação aos parâmetros definidos. Isso é essencial para identificar rapidamente os atletas com melhor desempenho.

- **Paginação**: A paginação permite que usuários visualizem mais jogadores caso a lista exceda o número padrão de exibição. Isso garante práticas modernas como *lazy loading*, fazendo com que a aplicação consuma menos recursos de *hardware* e que a análise não seja sobrecarregada visualmente, mantendo a clareza e a objetividade.
- **Gráfico de Radar**: Este gráfico é ideal para uma comparação detalhada entre os atletas e visualização do desempenho ponto a ponto em múltiplos parâmetros simultaneamente, como aceleração, condução e finalização. Ele permite uma análise multifacetada, na qual observa-se diferenciais e pontos de melhoria que cabem aos jogadores.

5 Considerações Finais e Trabalhos Futuros

Este trabalho concentrou-se na criação de uma plataforma que visa otimizar a análise de desempenho de jogadores de futebol por meio da coleta, tratamento e visualização de dados estatísticos. A solução foi projetada para atender à crescente demanda por ferramentas que auxiliem as comissões técnicas na tomada de decisões estratégicas, promovendo uma abordagem mais científica e baseada em evidências estatísticas.

O Futs3 se insere como uma ferramenta para coleta e processamento de dados de jogadores, permitindo uma compreensão mais profunda do jogo e uma otimização mais eficaz de formações com base no desempenho dos atletas. A plataforma combina tecnologias de monitoramento e computação em nuvem com métodos de análise de dados, facilitando a personalização de estratégias e treinamentos. Além disso, a aplicação é intuitiva e acessível, destacando-se pelo seu baixo custo. Dentre as funcionalidades já implementadas, destacam-se:

- **Coleta e Armazenamento de Dados**: O Futs3 permite a coleta de métricas essenciais, como precisão de passes, número de finalizações, desarmes e outros indicadores de desempenho, centralizando essas informações em um banco de dados com alta disponibilidade e segurança.
- **Análise e *Ranking***: A ferramenta oferece recursos para a geração de gráficos e relatórios de *ranking* completos e detalhados, que facilitam a interpretação dos dados coletados, permitindo uma visão clara dos melhores jogadores para uma determinada estratégia de jogo.

Como trabalhos futuros, é possível destacar funcionalidades que visam ampliar o potencial da plataforma:

- **Personalização de Treinamentos**: Com base nos dados analisados, o sistema deve sugerir treinamentos específicos que visam corrigir falhas e aprimorar habilidades, alinhando-se às necessidades individuais dos jogadores.
- ***Upload* de Planilhas**: Permitir *upload* de planilhas relacionadas ao *scouting* de jogadores.
- **Utilização de Modelos Preditivos de Desempenho**: Uso de *machine learning* para prever tendências de desempenho e sugerir ajustes táticos, considerando o contexto de um jogo, o perfil do time e os adversários. Além disso, esses modelos

permitem a análise de jogadores por similaridade, de forma que as escolhas por substitutos possam ser feitas de maneira eficiente.

- **Validação Avançada de Dados**: Mecanismos de verificação de consistência e integridade dos dados serão adicionados para evitar distorções nos resultados das análises, garantindo maior confiabilidade nas decisões tomadas com base nas informações fornecidas pela plataforma.

Por fim, o projeto será disponibilizado como uma solução em nuvem, garantindo acessibilidade e escalabilidade a equipes de diferentes regiões e com recursos distintos. O software desenvolvido pode ser acessado no *GitHub*[1].

5.1 Lições Aprendidas

O desenvolvimento do Futs3 trouxe a oportunidade de obter aprendizados valiosos no que diz respeito ao desenvolvimento *full stack*, que abrange desde a modelagem de dados, a criação de *interfaces* intuitivas até a implementação de um servidor capaz de lidar com grande carga de acessos e dados. A necessidade de lidar com diferentes tecnologias e integrações, como bancos de dados, APIs, processamento em nuvem e visualização de dados, exigiu um entendimento abrangente das camadas de um sistema de *software*. Esse processo destacou a importância de uma arquitetura bem estruturada, permitindo que fossem superados os desafios de manutenção e escalabilidade com mais eficiência, além de garantir que as diversas funcionalidades interagissem corretamente.

Outro aspecto fundamental foi tornar o sistema capaz de lidar com um alto volume de dados. Durante o desenvolvimento, foi preciso criar soluções que permitissem a coleta, tratamento e análise de grandes quantidades de dados de desempenho dos jogadores, sem comprometer a precisão ou a velocidade da plataforma. Isso demandou o uso de técnicas de otimização de consultas em banco de dados, bem como o desenvolvimento de fluxos operacionais internos que fossem eficientes para processar as informações rapidamente. O projeto demonstrou que trabalhar com grandes volumes de dados exige não apenas conhecimento técnico, mas também a capacidade de tomar decisões rápidas e eficazes para garantir que o sistema funcione de maneira estável e eficiente.

[1] https://github.com/luizmedeirosn

Referências

ABSEITS. **Os 13 melhores sites de estatísticas de futebol**. 2024. Acesso em: 31 ago. 2024. Disponível em: https://abseits.at/os-13-melhores-sites-de-estatisticas-de-futebol/. Citado na página 20.

ANDRADE, E. **Parte 02 - Criando Arquitetura em Camadas com DDD, Injeção de Dependências e EF**. 2023. Acesso em: 01 set. 2024. Disponível em: https://medium.com/@ericandrade_24404/parte-02-criando-arquitetura-em-camadas-com-ddd-inje%C3%A7%C3%A3o-de-dep-ef-defac0005667. Citado na página 31.

ANGULAR. **Angular Documentation**. 2024. Acesso em: 10 ago. 2024. Disponível em: https://angular.io/docs. Citado na página 38.

ASSURED, R. **REST Assured Documentation**. 2024. Disponível em: https://rest-assured.io/. Acesso em: 10 ago. 2024. Citado na página 40.

AWARI. **A ciência de dados aplicada ao futebol: como analisar e otimizar o desempenho dos jogadores**. 2023. Disponível em: https://awari.com.br/a-ciencia-de-dados-aplicada-ao-futebol-como-analisar-e-otimizar-o-desempenho-dos-jogadores/. Acesso em: 01 Mar 2024. Citado na página 16.

AWS. **Amazon Web Services Documentation**. 2024. Disponível em: https://docs.aws.amazon.com/. Acesso em: 10 ago. 2024. Citado 2 vezes nas páginas 42 e 43.

BASS L.; CLEMENTS, P. K. **Software Architecture in Practice**. 2. ed. [S.l.]: Addison-Wesley, 2003. 23–35, 97–103 p. Citado na página 30.

CARLING, C.; WILLIAMS, A. M.; REILLY, T. **Handbook of soccer match analysis: A systematic approach to improving performance**. 1. ed. London and New York: Routledge, 2005. E-book. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=Lfqq6NdzA3QC&oi=fnd&pg=PT10&dq=the+science+of+match+analysis&ots=GnfKliKOqP&sig=167EtC-308-uXDXD53n82vLtn5M#v=onepage&q=the%20science%20of%20match%20analysis&f=false. Citado na página 17.

DATA, T. P. T. **Amazon EC2 Deployment Complete CI/CD Pipeline using GitHub Actions and AWS CodeDeploy**. 2023. Acesso em: 01 set. 2024. Disponível em: https://medium.com/thelorry-product-tech-data/amazon-ec2-deployment-complete-ci-cd-pipeline-using-github-actions-and-aws-codedeploy-8a477123ff Citado na página 40.

DOCKER. **Docker Documentation**. 2024. Disponível em: https://docs.docker.com/. Acesso em: 10 ago. 2024. Citado na página 41.

ELMASRI, R.; NAVATHE, S. B. **Fundamentals of Database Systems**. 7. ed. [S.l.]: Pearson, 2017. Citado na página 33.

FLYWAY. **Flyway Documentation**. 2024. Disponível em: https://flywaydb.org/documentation. Acesso em: 10 ago. 2024. Citado na página 35.

FOWLER, M. **Patterns of Enterprise Application Architecture**. [S.l.]: Addison-Wesley Professional, 2003. Citado 2 vezes nas páginas 30 e 31.

FOWLER, M. **Continuous Integration: Improving Software Quality and Reducing Risk**. [S.l.]: Addison-Wesley, 2018. 15–28, 101–115 p. ISBN 978-0321579478. Citado 2 vezes nas páginas 32 e 33.

GLAZIER, P. S. Game, set and match? substantive issues and future directions in performance analysis. **Sports Medicine**, v. 40, n. 8, p. 625–634, 2010. Disponível em: https://doi.org/10.2165/11534970-000000000-00000. Citado na página 17.

GRAY, J.; REUTER, A. **Transaction Processing: Concepts and Techniques**. 1. ed. [S.l.]: Morgan Kaufmann, 1993. Citado na página 34.

GÓMEZ-RUANO, M. A. La importancia del análisis notacional como tópico emergente en ciencias del deporte: [the importance of performance analysis as an emergent research topic in sport sciences]. **RICYDE: Revista Internacional de Ciencias del Deporte**, v. 8, n. 47, p. 1–4, 2017. Disponível em: https://doi.org/10.5232/ricyde2017.047ed. Citado na página 17.

LOELIGER JON; MCGUIRE, M. **Version Control with Git**. 2. ed. [S.l.]: O'Reilly Media, 2012. 25–42, 89–110 p. ISBN 978-1449316389. Citado na página 33.

LUIZTOOLS. **Autenticação JSON Web Token (JWT) em Node.js**. 2023. Acesso em: 01 set. 2024. Disponível em: https://www.luiztools.com.br/post/autenticacao-json-web-token-jwt-em-nodejs/. Citado na página 36.

MOMJIAN, B. **PostgreSQL: Introduction and Concepts**. [S.l.]: Addison-Wesley, 2015. Citado na página 34.

NAAR, J. **JWT Cookbook: JSON Web Tokens Implementations in Node.js, Java, Python, and Golang**. [S.l.]: Packt Publishing, 2021. 22–45, 112–130 p. ISBN 978-1801074405. Citado na página 36.

PEDREÑO, J. M. **Scouting en Fútbol. Del fútbol base al alto rendimiento**. 2. ed. Vigo: MC Sports, 2018. Citado na página 17.

PRESSMAN, R. S.; MAXIM, B. R. **Engenharia de Software**. 9ª. ed. [S.l.]: McGraw Hill Brasil, 2021. ISBN 978-65-5804-010-1. Citado na página 16.

PRIMETEK. **PrimeNG Documentation**. 2024. Disponível em: https://www.primefaces.org/primeng/showcase/. Acesso em: 10 ago. 2024. Citado na página 39.

RICHARDSON LEONARD; RUBY, S. **RESTful Web Services**. 2. ed. [S.l.]: O'Reilly Media, 2020. 45–76, 123–145 p. ISBN 978-1491929502. Citado na página 30.

SHAMAH, M. E. do P. **Análise de desempenho no futebol: a prática do analista de desempenho nas categorias de base dos clubes brasileiros da Série A**. 2021. Disponível em: https://lume.ufrgs.br/bitstream/handle/10183/233037/001134830.pdf?sequence=1. Acesso em: 01 Mar 2024. Citado na página 16.

SMITH, J. Introduction to devops. **Tech Journal**, v. 15, n. 3, p. 45–50, 2024. Citado na página 40.

SOFASCORE. **SofaScore Platform**. 2024. Acesso em: 31 ago. 2024. Disponível em: https://www.sofascore.com/. Citado na página 21.

SPORTSBASE. **Sportsbase Platform**. 2024. Acesso em: 31 ago. 2024. Disponível em: https://football.sportsbase.world/. Citado na página 20.

SPRING. **Spring Boot Documentation**. 2024. Disponível em: https://spring.io/projects/spring-boot. Acesso em: 10 ago. 2024. Citado na página 35.

SPRING. **Spring Data JPA Documentation**. 2024. Disponível em: https://spring.io/projects/spring-data-jpa. Acesso em: 10 ago. 2024. Citado na página 35.

SPRING. **Spring Security Documentation**. 2024. Disponível em: https://spring.io/projects/spring-security. Acesso em: 10 ago. 2024. Citado na página 36.

SWAGGER. **Swagger UI**. 2023. Acesso em: 01 set. 2024. Disponível em: https://swagger.io/tools/swagger-ui/. Citado na página 37.

SWAGGER. **Swagger Documentation**. 2024. Disponível em: https://swagger.io/docs/. Acesso em: 10 ago. 2024. Citado na página 37.

TALKS, Q. T. **Designing a Public Asynchronous API**. 2024. Disponível em: https://techtalks.qima.com/designing-a-public-asynchronous-api/. Acesso em: 1 set. 2024. Citado na página 26.

TESTCONTAINERS. **Testcontainers Documentation**. 2024. Disponível em: https://www.testcontainers.org/. Acesso em: 10 ago. 2024. Citado na página 40.

TIP, A. **Containerizing a Java Spring Boot Application and Deploying to ECS**. 2023. Acesso em: 01 set. 2024. Disponível em: https://awstip.com/containerizing-a-java-spring-boot-application-and-deploying-to-ecs-3c7cc73f7c6. Citado na página 43.

VENTURA, N. **Observar para ganhar. O Scouting como Ferramenta do Treinador**. 2. ed. Portugal: Prime Books, 2013. Citado na página 17.

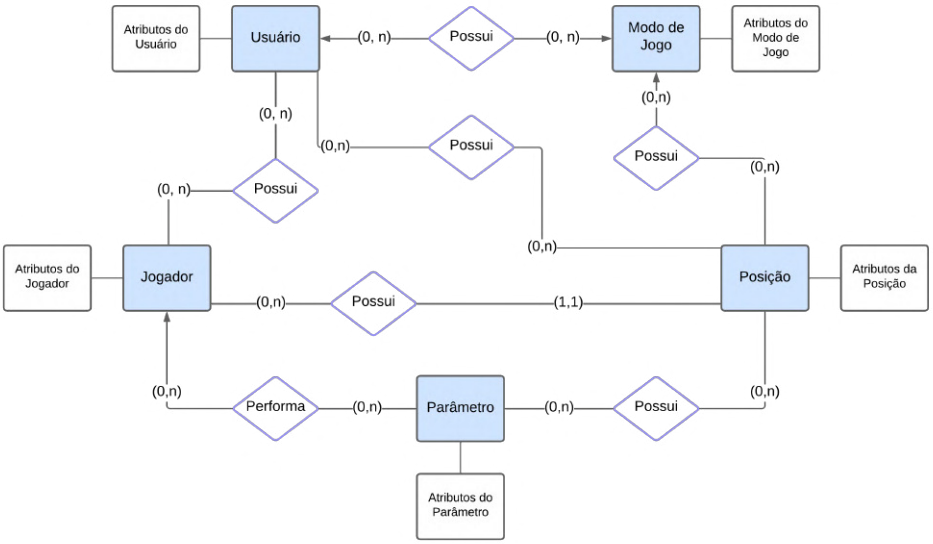
VOLOSSOVITCH, A.; FERREIRA, A. P. **Da descrição estática à predição dinâmica. A evolução das perspetivas de análise da performance nos jogos desportivos coletivos**. Lisboa: Edições FMH, 2013. 1-34 p. Citado 2 vezes nas páginas 16 e 17.

WYSCOUT. **Wyscout Platform**. 2024. Acesso em: 31 ago. 2024. Disponível em: https://wyscout.hudl.com/. Citado na página 20.

Apêndices

APÊNDICE A – MER do FUTS3

Figura 22 – Modelo Entidade Relacionamento da Aplicação.



Fonte: Elaborado pelo autor, 2024

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Campina Grande - Código INEP: 25137409
R. Tranqüilino Coelho Lemos, 671, Dinamérica, CEP 58432-300, Campina Grande (PB)
CNPJ: 10.783.898/0003-37 - Telefone: (83) 2102.6200

Documento Digitalizado Restrito

Versão Final do TCC

Assunto:	Versão Final do TCC
Assinado por:	Luiz Medeiros
Tipo do Documento:	Comprovante
Situação:	Finalizado
Nível de Acesso:	Restrito
Hipótese Legal:	Controle Interno (Art. 26, $ 3o, da Lei no 10.180/2001)
Tipo do Conferência:	Cópia Simples

Documento assinado eletronicamente por:

- **Luiz Medeiros Neto, ALUNO (201921250001) DE BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO - CAMPINA GRANDE**, em 06/10/2024 00:45:46.

Este documento foi armazenado no SUAP em 06/10/2024. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 1268709
Código de Autenticação: c7f5d9533b