UNIVERSIDADE FEDERAL DA PARAÍBA

CENTRO DE INFORMÁTICA

COORDENAÇÃO DE BACHARELADO EM CIÊNCIAS DA COMPUTAÇÃO

TRABALHO DE CONCLUSÃO DE CURSO

YURI FELIX RAMALHO DE OLIVEIRA

DESENVOLVIMENTO E IMPLANTAÇÃO DE UMA PLATAFORMA WEB PARA PREDIÇÃO DE INTERAÇÕES ATÔMICAS DE ESTRUTURAS TRIDIMENSIONAIS DE PROTEÍNAS COM O USO DO YSERA

João Pessoa - PB

Junho/2023

YURI FELIX RAMALHO DE OLIVEIRA

DESENVOLVIMENTO E IMPLANTAÇÃO DE UMA PLATAFORMA WEB PARA PREDIÇÃO DE INTERAÇÕES ATÔMICAS DE ESTRUTURAS TRIDIMENSIONAIS DE PROTEÍNAS COM O USO DO YSERA

Trabalho de Conclusão de Curso apresentado ao curso de Ciências da Computação do Centro de Informática, da Universidade Federal da Paraíba, como requisito parcial para a Obtenção do grau de Bacharel em Ciências da Computação.

Orientadora: Profa. Dra. Thaís Gaudencio do Rêgo

João Pessoa - PB

2023

UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
COORDENAÇÃO DO CURSO

Centro de Informática
UFPB

Ata da Sessão Pública de Defesa de Trabalho de Conclusão de Curso de **Ciência da Computação**, realizada em 23 de junho de 2023.

Aos **23** dias do mês de **junho**, do ano de **2023**, às 09:00 horas, reuniram-se no Centro de In formativa os membros da Banca Examinadora constituída para julgar o Trabalho de Conclusão de Curso do Sr**. Yuri Felix Ramalho de Oliveira**, matrícula nº **11508386**, aluno do Curso de Bacharelado em Ciência da Computação da Universidade Federal da Paraíba. A comissão examinadora foi composta pela professora **Thais Gaudêncio do Rêgo** (UFPB), orientadora e presidente da banca, e pelas professoras **Thaís de Almeida Ratis Ramos** com Doutorado em Bioinformática (UFRN) examinadora externa e **Annie Elisabeth Beltrão de Andrade,** mestra no Programa de Pós-Graduação em Informática da UFPB. Iniciando os trabalhos, a presidente da banca cumprimentou os presentes, comunicou-os da finalidade da reunião e passou a palavra ao candidato para que fizesse a exposição oral da monografia intitulada ***"Desenvolvimento e Implantação de uma Plataforma web para Predição de Interações Atômicas de Estruturas Tridimensionais de Proteínas com o uso do Ysera"***. Concluída a exposição, o candidato foi arguido pela Banca Examinadora que, em seguida, emitiu o seguinte parecer: "***aprovado***", com conceito 9,5. Do ocorrido, eu, Leandro Carlos de Souza, Coordenador do Curso de Bacharelado em Ciência da Computação, lavrei a presente ata que vai assinada por mim e pelos membros da banca examinadora. João Pessoa, 23 de junho de 2023.

Prof. Leandro Carlos de Souza
Coordenador do Curso de Ciência da Computação
SIAPE 1140339

Profa. Thaís Gaudencio do Rêgo
Orientadora (UFPB)

Thaís Gaudencio do Rêgo

Thaís de Almeida Ratis Ramos
Doutora em Bioinformática (UFRN)

Thaís de Almeida Ratis Ramos

Profa. Annie Elisabeth Beltrão de Andrade
Mestra em Informática (UFPB)

ANNIE ELISABETH BELTRÃO DE ANDRADE

Emitido em 23/06/2023

ATA Nº -/2023 - CI - CCC (18.56.01)
(Nº do Documento: 9)

(Nº do Protocolo: NÃO PROTOCOLADO)

(Assinado digitalmente em 26/06/2023 15:16)
LEANDRO CARLOS DE SOUZA
COORDENADOR DE CURSO
1140339

Para verificar a autenticidade deste documento entre em https://sipac.ufpb.br/documentos/ informando seu número: **9**, ano: **2023**, documento (espécie): **ATA**, data de emissão: **26/06/2023** e o código de verificação: **f8ae46ced1**

Catalogação na publicação
Seção de Catalogação e Classificação

O48d Oliveira, Yuri Felix Ramalho de.
Desenvolvimento e implantação de uma plataforma web para predição de interações atômicas de estruturas tridimensionais de proteínas com o uso do ysera / Yuri Felix Ramalho de Oliveira. - João Pessoa, 2023.
44 f.

Orientação: Thaís Gaudêncio do Rêgo.
TCC (Graduação) - UFPB/CI.

1. Plataforma web. 2. Predição. 3. Interações atômicas. 4. Estruturas tridimensionais. 5. Proteínas. 6. Ysera. I. Rêgo, Thaís Gaudencio do. II. Título.

UFPB/CI CDU 004.777

Elaborado por Michelle de Kássia Fonseca Barbosa - CRB-738

Dedico este trabalho à minha família e amigos, que sempre me incentivaram.

AGRADECIMENTOS

Agradeço aos meus professores e colegas por me ajudarem a desenvolver este trabalho. Agradeço também a minha família, amigos e a minha noiva Beatriz, por me incentivar e dar apoio à conclusão deste curso.

"*Só sei que nada sei.*" (Sócrates)

RESUMO

O desenvolvimento de uma plataforma capaz de identificar, quantificar e qualificar as interações moleculares é essencial para o estudo e a predição das interações moleculares de proteínas, que é uma importante ferramenta para o desenvolvimento de fármacos, agroquímicos e várias outras áreas da biologia. O objetivo deste trabalho é implementar uma plataforma web para predição de interações atômicas de estruturas tridimensionais de proteínas, utilizando o preditor Ysera. A plataforma desenvolvida permite que os usuários acessem o preditor de forma online, sem a necessidade de instalar o software em seus computadores. Além disso, a plataforma foi projetada para oferecer uma interface amigável e intuitiva, permitindo que os usuários enviem as estruturas de proteínas, em formato PDB, e obtenham os resultados da predição de forma fácil e rápida. A monografia descreve detalhadamente todo o processo de desenvolvimento da plataforma, desde a escolha das tecnologias utilizadas até a implementação das funcionalidades. Também são discutidos os desafios encontrados durante o desenvolvimento e as soluções adotadas para superá-los. Por fim, são apresentados os resultados obtidos com a implantação da plataforma e discutidas as perspectivas futuras para o seu aprimoramento. Desta forma, esse projeto contribui para o avanço e democratização do conhecimento científico, entregando uma plataforma simples, intuitiva e de fácil acesso.

Palavras-chave: Plataforma web; predição; interações atômicas; estruturas tridimensionais; proteínas; Ysera

ABSTRACT

The development of a platform capable of identifying, quantifying, and qualifying molecular interactions is essential for studying and predicting protein molecular interactions, which is an important tool for drug development, agrochemicals, and various other areas of biology. The objective of this work is to implement a web platform for predicting atomic interactions of three-dimensional protein structures using the Ysera predictor. The developed platform allows users to access the predictor online without the need to install the software on their computers. Furthermore, the platform has been designed to offer a user-friendly and intuitive interface, allowing users to submit protein structures in PDB format and obtain prediction results easily and quickly. The thesis provides a detailed description of the entire platform development process, from the selection of technologies used to the implementation of functionalities. It also discusses the challenges encountered during development and the solutions adopted to overcome them. Finally, the results obtained from the platform deployment are presented, and future prospects for its improvement are discussed. Thus, this project contributes to the advancement and democratization of scientific knowledge, delivering a simple, intuitive, and easily accessible platform.

Keywords: Web platform; prediction; atomic interactions; three-dimensional structures; proteins;Ysera.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

O estudo e a predição das interações moleculares de proteínas é uma importante ferramenta para o desenvolvimento de fármacos, agroquímicos e várias outras áreas da biologia. Diante disso, se faz necessário o desenvolvimento de uma plataforma capaz de fornecer ao pesquisador, um acesso fácil e parametrizável a um software capaz de identificar as interações moleculares e quantificar e qualificar as interações intramoleculares.

A disponibilização de ferramentas das mais diversas, surgem na internet, como forma de disseminar o conteúdo científico e sobretudo, permitir o acesso simples, rápido e democrático a artifícios tecnológicos das mais diversas áreas do conhecimento.

Softwares de cálculo científico, como o mencionado acima, são frequentemente conhecidos por demandarem um custo computacional elevado. Isso significa que os usuários precisam de uma capacidade de processamento significativa em seus dispositivos ou, em alternativa, estarem preparados para esperar por longos períodos de tempo para obter os resultados do cálculo. Felizmente, uma aplicação web e assíncrona pode descomplicar este tempo de espera, fornecendo um poder de processamento dedicado para executar o script desejado, sem prejudicar o desempenho do dispositivo pessoal do usuário.

Essa é uma solução efetiva que pode reduzir a barreira de entrada para o uso de softwares científicos. Como a aplicação é executada em um servidor, o usuário não precisa se preocupar em ter um hardware poderoso para realizar os cálculos. Em vez disso, a aplicação pode ser acessada de qualquer dispositivo com acesso à internet, como smartphones, tablets, laptops ou desktops, e a solução da carga computacional pode ser facilmente terceirizada para o servidor da aplicação.

Ao disponibilizar esses serviços online, a democratização do acesso ao conhecimento científico se torna uma realidade. Esse tipo de ferramenta pode ser particularmente importante em países como o Brasil, onde uma parcela significativa da população tem dificuldade para acessar dispositivos de alto poder tecnológico. Com uma solução web e assíncrona, essas pessoas podem ter acesso a serviços de

cálculo científico sem precisar investir em hardware sofisticado ou enfrentar longos períodos de espera para obter os resultados do cálculo. Isso pode representar uma grande oportunidade para estudantes, pesquisadores e profissionais de diversos setores, que buscam soluções eficazes para seus problemas sem precisar lidar com as limitações impostas por seus dispositivos pessoais.

Além disso, o armazenamento de arquivos de forma remota se tornou uma realidade cada vez mais presente na computação, visto que as pessoas estão cada vez mais conectadas e necessitam acessar seus arquivos de qualquer lugar do mundo. Nesse sentido, este projeto permite aos usuários o acesso de seus resultados de qualquer dispositivo conectado à internet, sem precisar baixar ou manter os arquivos armazenados no próprio dispositivo. Isso além de economizar espaço de disco, também permite que o compartilhamento e a colaboração em projetos científicos sejam mais fáceis de serem realizados.

1.1 OBJETIVOS

O objetivo deste projeto é simplificar o acesso a um software preditor de interações moleculares, de modo que reduza a necessidade de grandes conhecimentos computacionais, como linguagens de programação e uso de terminais. Ainda assim, visa manter a capacidade de parametrização do software, fornecendo assim uma ferramenta simples, customizável e capaz de ser executada em computadores domésticos.

1.2 OBJETIVOS ESPECÍFICOS

Como objetivos específicos deste trabalho encontra-se:

- Realizar uma pesquisa acerca das tecnologias web adequadas para o projeto;
- Implantar uma plataforma web capaz de realizar predições de interações moleculares;
- Desenvolver funcionalidades como cadastro e login
- Comunicação entre APIs

- Envio e leitura de dados
- Realizar uma análise das opções disponíveis no meio científico
- Realizar comparação de ferramentas de predição, escolhendo a com melhores resultados.

1.3 ORGANIZAÇÃO DO TRABALHO

No segundo capítulo deste trabalho, é apresentado a base teórica por trás da construção da plataforma construída, desde conceitos relacionados à comunicação entre o servidores e clientes, tecnologias e conceitos focados no desenvolvimento de aplicações web e o software YSERA.

No terceiro capítulo, é descrita a metodologia utilizada no desenvolvimento do produto, a estruturação do projeto e os serviços criados para cada parte da aplicação. Bem como as tecnologias implantadas em cada funcionalidade.

Por fim, apresenta-se os resultados obtidos e compara-se com outros serviços cujo objetivos são semelhantes. Além de apresentar uma conclusão daquilo que foi o desenvolvimento do produto.

2 FUNDAMENTAÇÃO TEÓRICA

O objetivo deste capítulo é fornecer um referencial teórico acerca dos métodos de desenvolvimento de software apresentados nesse projeto, as linguagens e os *frameworks* utilizados. Serão ainda apresentadas as principais vantagens e os pontos negativos das tecnologias escolhidas e, desta forma, pretende-se justificar as escolhas de tais ferramentas.

2.1 O MODELO CLIENTE-SERVIDOR

Segundo Marcial Porto Fernández (2019), clientes e servidores são dispositivos computacionais com sistemas operacionais, distintos e conectados entre si através da rede. Um dos objetivos da conexão entre esses dispositivos é garantir uma comunicação transparente, a fim de garantir que os mais diversos usuários, os clientes,possam solicitar e enviar informações aos servidores, independente do sistema operacional que esteja usando. Desta forma, é extremamente importante manter a padronização dos protocolos de rede, garantindo assim a interoperabilidade e a compatibilidade entre diversos sistemas e dispositivos (FERNÁNDEZ, 2019).

2.1.1 Cliente

O lado cliente de uma conexão cliente-servidor é representado como um dispositivo que conecta a interface do usuário à rede. Esse cliente, resumidamente, se caracteriza como um navegador, que é um programa desenvolvido para interpretar e exibir páginas da internet e permitir a navegação entre elas, através de hiperlinks (FERNÁNDEZ, 2019). Como explicado por Tanenbaum (2003), um

navegador funciona como a principal interface de comunicação do usuário e a Web, facilitando assim a interação e visualização dos conteúdos disponíveis online.

Segundo Tanenbaum (2003), durante o processo de navegação, o lado cliente começa ao selecionar um link contendo uma *Uniform Resource Locator* - URL (do português, Localizador Uniforme de Recursos) específica. Para conseguir se comunicar com o site associado a essa URL, o cliente precisa ter o endereço IP do servidor correspondente. Para isso, ele estabelece uma conexão com um servidor específico, chamado servidor de *Domain Name System* - DNS (do português, Sistema de Nomes de Domínio). O servidor de DNS então, fornece ao cliente o endereço de *Internet Procotol* - IP (do português, Protocolo de Rede) da página desejada, permitindo assim que o cliente saiba para onde disparar as solicitações de conteúdo.

De posse do endereço de IP, o lado cliente inicia uma conexão do tipo *Trasmission Control Protocol* - TCP (do português, Protocolo de Controle de Transmissão) com o servidor do endereço recebido pelo servidor de DNS. No próximo passo, realiza uma solicitação ao servidor, para que este responda com o arquivo *HyperText Markup Language*, HTML (do português, Linguagem de Marcação de HiperTexto) associado à página desejada. Por exemplo, o arquivo "index.html" habitualmente representa a página principal e inicial de um site. Ao fim dessa solicitação, encerra-se a conexão TCP e a página em questão é interpretada e renderizada para o usuário, por meio do navegador (Tanenbaum, 2003).

O cliente então, representa uma importante parte desta conexão. A outra parte se dá no servidor da aplicação.

2.1.2 Servidor

Os servidores e clientes são dispositivos computacionais com sistemas operacionais distintos, conectados por meio de redes. Uma das principais metas é assegurar uma comunicação transparente entre esses dispositivos, permitindo que

usuários com diferentes sistemas operacionais possam solicitar informações aos servidores. Nesse sentido, é fundamental padronizar os protocolos de rede utilizados nessa comunicação, garantindo assim a interoperabilidade e a compatibilidade entre diversos sistemas e dispositivos (FERNÁNDEZ, 2019).

Conforme apontado por Tanenbaum (2003) em seu livro "Redes de Computadores", o lado cliente da conexão é representado por um equipamento que conecta a interface do usuário à rede. Esse cliente, em termos simples, consiste em um navegador, que é um programa projetado para exibir páginas da Web e permitir a navegação entre elas por meio de hiperlinks.

O cliente então, de posse da URL selecionada no hiperlink, precisa conectar um servidor de DNS para saber qual é o endereço IP do site em questão. Após isso, é necessário estabelecer uma conexão TCP com o servidor localizado nesse endereço IP. O cliente solicita o arquivo HTML da página em questão (por exemplo, "index.html"). A conexão então é encerrada e o navegador interpreta o arquivo HTML (TANENBAUM, 2003).

Essas informações baseadas nas diretrizes de Marcial Porto Fernández (2019) e nos ensinamentos de Tanenbaum (2003) evidenciam a importância da padronização dos protocolos de rede para garantir a comunicação eficiente entre servidores e clientes. Além disso, compreende-se a relevância do navegador como a interface principal entre o usuário e o mundo da Web, permitindo a exibição de conteúdos online por meio da interação com os servidores.

Nesse contexto, é possível compreender a sequência de ações realizadas pelo cliente, desde a obtenção do endereço IP, por meio do servidor de DNS, até a solicitação e interpretação dos arquivos HTML. Essa interação entre cliente e servidor desempenha um papel fundamental na experiência de navegação na Web, facilitando o acesso a informações e conteúdos por parte dos usuários.

Algumas linguagens podem auxiliar o desenvolvimento no lado cliente, outras no lado do servidor. Entretanto, algumas linguagens buscam se colocar nos dois ambientes, como é o caso de Javascript.

2.2 JAVASCRIPT

Segundo Wirfs-Brock (2020), JavaScript ocupa a posição de linguagem de programação mais amplamente utilizada em todo o mundo. De acordo com uma pesquisa realizada pelo Stack Overflow em 2018, mais de 70% dos desenvolvedores profissionais utilizavam JavaScript. Essa linguagem de programação foi desenvolvida pela Netscape nos meados dos anos 90. Conforme descrito no livro "Professional JavaScript for Web Developers" de Nicholas C. Zakas, o objetivo inicial do JavaScript era lidar com validações de entrada que, até então, eram tratadas no lado do cliente (do inglês, *server-side*) por linguagens como Perl. Desde então, JavaScript se tornou uma funcionalidade essencial em todos os principais navegadores, proporcionando um suporte poderoso para a criação de aplicações web interativas (WIRFS-BROCK, 2020).

Com o passar dos anos, Javascript começou a ser utilizado no lado do servidor de uma aplicação web, com o advento do Node.JS

2.3 NODE.JS

Em 2009, Ryan Dahl desenvolveu o Node.js, inspirado por implementações como Python e Ruby, e utilizando a *engine* JavaScript V8 do Chrome, que é um software destinado à execução de JavaScript, desenvolvido pelo Google, juntamente com a biblioteca de eventos *Libev*. Essa inovação permitiu a execução do JavaScript fora do ambiente de navegadores (*client-side*), proporcionando uma plataforma versátil para o desenvolvimento de aplicações *server-side* (do português, lado do servidor) (WANDSCHNEIDER, 2020).

Conforme destacado por Bangare *et al*. (2016), uma parte fundamental de um servidor web é a capacidade de gerenciar eficientemente múltiplos usuários. Nesse contexto, o Node.js se destaca como ferramenta para o processamento e entrega eficiente de dados para e de um servidor. Sua arquitetura baseada em eventos e a

natureza assíncrona possibilitam um desempenho de alta velocidade e escalabilidade para aplicações backend. Uma ferramenta de grande importância para a escalabilidade, segurança e velocidade de desenvolvimento, são os *frameworks*, como o React.

2.4 REACT

O React.js é uma biblioteca JavaScript amplamente utilizada para a construção de interfaces de usuário interativas e reativas. Desenvolvido pelo Facebook, o React.js oferece uma abordagem declarativa para a construção de componentes reutilizáveis, o que permite uma divisão clara das responsabilidades e facilita a manutenção do código. Com seu modelo de programação baseado em componentes, o React.js permite a construção de interfaces dinâmicas e responsivas, onde as mudanças de estado são automaticamente refletidas na interface sem a necessidade de atualizações manuais. Além disso, o React.js possui uma comunidade ativa e uma vasta gama de recursos e bibliotecas complementares, o que contribui para sua popularidade e adoção generalizada no desenvolvimento web moderno (SCHAUERMAN, 2020; BAKER, 2018).

Assim, é muito importante que sejam observados os requisitos iniciais de uma aplicação. Um bom desenvolvimento em React, garante que tal objetivo seja cumprido.

2.5 REQUISITOS DE SOFTWARE

Segundo Sommerville (2011), os requisitos de software são descrições das funcionalidades, restrições e características que o sistema deve possuir para atender às necessidades dos usuários e das partes interessadas. Esses requisitos

podem ser divididos em duas categorias principais: requisitos funcionais e requisitos não funcionais.

Os requisitos funcionais, conforme Pressman (2016), definem as funções específicas que o sistema deve ser capaz de realizar. Eles descrevem as ações, os comportamentos e as operações que o sistema deve executar para atender aos objetivos do usuário. Esses requisitos geralmente são expressos em termos de entradas, processamento e saídas do sistema. Exemplos de requisitos funcionais podem incluir a capacidade de realizar determinadas operações, processar informações de entrada de acordo com regras específicas ou fornecer determinadas saídas para o usuário.

Por outro lado, os requisitos não funcionais, de acordo com Sommerville (2011), são focados nas características e restrições relacionadas ao desempenho, segurança, usabilidade, confiabilidade e outros aspectos do sistema. Eles descrevem como o sistema deve se comportar em termos de desempenho, capacidade, tempo de resposta, disponibilidade, segurança, entre outros. Esses requisitos são importantes para garantir que o sistema atenda aos padrões de qualidade e às expectativas dos usuários. Exemplos de requisitos não funcionais podem incluir tempos de resposta rápidos, alta disponibilidade, segurança robusta, facilidade de uso e escalabilidade.

Atualmente, uma das formas utilizadas para aplicar-se requisitos como este, é através de um SAAS.

2.6 SAAS

O modelo de Software como Serviço (do inglês, *Software as a Service* - SaaS) tem se tornado cada vez mais popular nas organizações. Conforme definido por Mell e Grance (2011), o SaaS é um modelo de distribuição de software em que os aplicativos são disponibilizados aos usuários por meio da internet, permitindo acesso fácil e rápido, sem a necessidade de instalação local. Isso proporciona

benefícios como redução de custos de infraestrutura e manutenção, além de oferecer maior flexibilidade e escalabilidade para atender às demandas dos usuários. Esse modelo tem sido adotado por empresas de diversos setores, impulsionando a transformação digital e a agilidade nos negócios (MELL; GRANCE, 2011).

A adoção do modelo SaaS tem impulsionado a colaboração e a integração entre empresas. De acordo com Bhatnagar e Sharma (2020), o SaaS permite que diferentes organizações compartilhem aplicativos e dados de forma mais eficiente, facilitando a colaboração em projetos conjuntos e o compartilhamento de informações entre parceiros de negócios. Isso promove a agilidade e a inovação, uma vez que as empresas podem se concentrar em suas principais competências e utilizar aplicativos prontos para uso fornecidos por provedores de SaaS para outras funcionalidades. Essa colaboração entre empresas contribui para o aumento da eficiência e da competitividade no mercado (BHATNAGAR; SHARMA, 2020).

Através da plataforma desenvolvida aqui, é possível transformar o YSERA em um SAAS.

2.7 YSERA

Ysera é um software preditor de interações moleculares, auxiliando assim o desenvolvimento e o estudo científico de proteínas, baseado em arquivos de estruturas tridimensionais, cuja extensão utilizada é o .pdb. Este programa consegue predizer interações moleculares como pontes salinas, ligações de hidrogênio e outras. Partindo da localização espacial, ou seja, com as coordenadas tridimensionais, dos átomos que constituem a proteína, calcula-se a distância euclidiana de todos os átomos entre si, analisando as interações encontradas (ANDRADE, 2020).

2.8 TRABALHOS RELACIONADOS

Dentre os trabalhos relacionados na área de análise de interações atômicas em estruturas protéicas, um deles é o RING (CLEMENTEL et al., 2022). Este software fornece ao usuário um software, que permite a identificação de diferentes tipos de interações, como ligações covalentes e não covalentes, incluindo interações de π-π e π-cátion. Algumas características de RING é ser veloz e preciso, devido uma série de reparametrização empírica dos limites de distância (*distance thresholds*) realizada no arquivo PDB lido. O RING pode ser executado por meio do Cytoscape, scripts ou na web. Mais à frente, discutiremos as semelhanças e diferenças do RING para este trabalho.

3 METODOLOGIA

Este capítulo visa descrever a metodologia utilizada para o desenvolvimento da plataforma. A primeira parte descreve os requisitos levantados para a aplicação. Após, é mostrada a estruturação do projeto. Por fim, as sessões finais buscam detalhar cada parte do projeto, a api do projeto, o *front-end* e o serviço específico para o software preditor.

3.1 REQUISITOS FUNCIONAIS E NÃO-FUNCIONAIS DA PLATAFORMA

A plataforma web desenvolvida para predição de interações atômicas de estruturas tridimensionais de proteínas, com o uso do Ysera, foi planejada para atender a uma série de requisitos funcionais e não-funcionais, a fim de garantir o bom funcionamento da aplicação e a satisfação dos usuários.

3.1.1 Requisitos Funcionais

Os requisitos funcionais descrevem as funcionalidades que a plataforma deve oferecer para que os usuários possam realizar as tarefas necessárias para a predição de interações atômicas de estruturas tridimensionais de proteínas. São eles:

- Submissão de arquivos PDB contendo a estrutura tridimensional da proteína a ser analisada;
- Execução do preditor Ysera na estrutura tridimensional da proteína;
- Geração de resultados da predição de interações atômicas;

- Disponibilização dos resultados para download pelo usuário;
- Login de usuários para acesso à plataforma;
- Recuperação de senha em caso de esquecimento;
- Envio de e-mails para notificações de status da predição.

3.1.2 Requisitos Não-Funcionais

Os requisitos não-funcionais descrevem as características que a plataforma deve possuir para garantir a qualidade e o bom desempenho da aplicação. São eles:

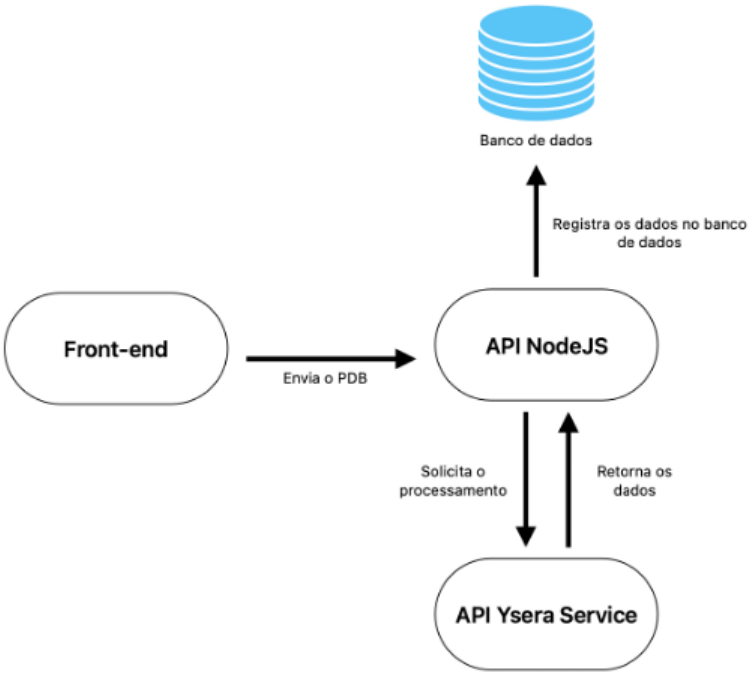
- Usabilidade: a plataforma deve ser intuitiva e fácil de usar para que os usuários possam realizar as tarefas com eficiência;
- Performance: a plataforma deve ser capaz de processar grandes quantidades de dados em tempo hábil, a fim de evitar atrasos e garantir a satisfação do usuário;
- Segurança: a plataforma deve garantir a segurança dos dados dos usuários e a privacidade das informações submetidas;
- Confiabilidade: a plataforma deve ser confiável e estar disponível para os usuários sempre que necessário;

3.2 ESTRUTURA

A forma de desenvolvimento deste sistema consiste em três projetos que serão descritos a seguir. Um dos projetos é a aplicação *front-end*, desenvolvida

utilizando ReactJS. Uma *Application Programming Interface* - API (do português, Interface de Programação de Aplicação) em NodeJS, responsável pelo núcleo (do inglês, *core*) do projeto e uma API em python que executa o Ysera. A Figura 1 ilustra o modelo.

Figura 1 — Estrutura do sistema.



Fonte: Autoria própria (2023).

O *front-end* da aplicação foi desenvolvido utilizando a biblioteca *ReactJS*, que é amplamente utilizada por criar interfaces de usuário (UI) robustas para aplicações web. O React foi fundamental na criação de componentes reutilizáveis e arquitetura escalável, permitindo que a interface gráfica do usuário fosse construída rapidamente e eficientemente.

A API em NodeJS conecta o *front-end* ao serviço Python do Ysera. O NodeJS tornou possível o desenvolvimento de *endpoints*, que permitem que o *front-end* envie arquivos PDB para serem processados pelo Ysera, além de receber e preservar os resultados desses cálculos. Além disso, a API em NodeJS também

realiza validações e autenticações de usuários, garantindo a segurança e a integridade do sistema.

Por fim, a API em Python executa o Ysera, o software utilizado para cálculos moleculares. A API em Python permite o envio dos arquivos PDB recebidos da API em NodeJS para o Ysera e a recuperação dos resultados desses cálculos. A biblioteca *Flask*, amplamente utilizada na criação de APIs em Python, foi usada para desenvolver a API em Python e configurada para se comunicar com a API em NodeJS por meio de solicitações HTTP.

Nas seções a seguir, serão fornecidas descrições detalhadas de cada seção da implementação do sistema.

3.3 FRONT-END

3.3.1 Visão geral

Como mencionado anteriormente, o desenvolvimento do *front-end* desta aplicação foi construído com a utilização da biblioteca *ReactJS* na sua versão 17. Para otimizar o tempo de desenvolvimento e facilitar a implementação de recursos específicos, algumas outras bibliotecas foram utilizadas na construção do *front-end*, como a *Styled-components*, uma biblioteca que permite escrever CSS (do inglês, *Cascading Style Sheets* e do português, Folhas de Estilo em Cascata) diretamente no código JavaScript, tornando a estilização dos componentes mais intuitiva e fácil de ser gerenciada.

Outra biblioteca utilizada foi a *Axios*, uma ferramenta de requisição HTTP que permite a realização de operações assíncronas com facilidade. Com essa biblioteca, é possível realizar solicitações de dados a partir do servidor, sem a necessidade de recarregar a página, ou interromper o fluxo do usuário.

Por fim, a biblioteca gráfica *AntDesign* também foi utilizada na construção do *front-end*. Essa biblioteca oferece uma série de componentes e recursos para a criação de interfaces modernas e atrativas, incluindo botões, menus, tabelas, formulários, gráficos, entre outros. Além disso, seus componentes seguem as práticas de design e experiência do usuário (do inglês, *User Experience* - UX), oferecendo uma experiência de usuário fluida e intuitiva.

3.3.2 Partes da aplicação

O *front-end* desenvolvido para essa aplicação é composto por diversos recursos e funcionalidades importantes para a sua utilização. Em primeiro lugar, existe um fluxo intuitivo para o envio de arquivos PDBs e um formulário que permite o preenchimento de algumas configurações utilizadas no Ysera, tornando o processo de análise de estruturas moleculares mais eficiente e prático.

Além disso, a aplicação oferece uma área logada, na qual o usuário pode consultar as requisições já processadas e obter os resultados em formato de arquivo de texto. Essa funcionalidade é essencial para permitir que o usuário acompanhe o progresso das análises solicitadas e obtenha os resultados de maneira rápida e fácil.

Outro recurso importante presente no *front-end* é o gerenciamento de acesso, com o cadastro e *login* do usuário. Esse sistema de autenticação é fundamental para garantir a segurança dos dados do usuário e permitir que ele acesse as funcionalidades restritas do sistema.

Por fim, a aplicação possui uma sessão de descrição do projeto, que oferece informações detalhadas sobre a finalidade e os objetivos da ferramenta Ysera, além de fornecer orientações e dicas para a utilização correta da aplicação. Essa sessão é essencial para orientar o usuário e garantir que ele compreenda plenamente as funcionalidades e benefícios da ferramenta.

3.3.3 Estrutura

Para manter o projeto bem organizado e fácil de ser gerenciado, foram criadas algumas pastas essenciais no desenvolvimento do *front-end*. A primeira delas é a pasta *Components*, que consiste em componentes reutilizáveis que podem ser utilizados em várias páginas do projeto. Esses componentes são projetados para serem modulares e escaláveis, permitindo a fácil adaptação às necessidades específicas de cada página.

A pasta *Pages* contém componentes que servem como raiz para a renderização de uma ou mais rotas. Esses componentes geralmente incluem outras sub-rotas e componentes menores, formando uma hierarquia de elementos que compõem a página em questão.

A pasta *Assets* contém arquivos de imagem, logotipos e outros recursos visuais que são utilizados no projeto. Esses recursos são organizados de forma a permitir uma fácil localização e acesso, facilitando o processo de desenvolvimento e manutenção do projeto.

Já a pasta *Services* encapsula serviços importantes para o projeto, como serviços de autenticação e API, que são essenciais para a comunicação entre o *front-end* e o *back-end*. Esses serviços são projetados para serem facilmente gerenciados e escaláveis, permitindo que o projeto seja facilmente adaptável a diferentes necessidades.

Por fim, a pasta *Utils* contém funções úteis em todo o projeto, como funções de validação de dados, formatação de *strings* e outras funcionalidades genéricas. Essas funções são projetadas para serem facilmente acessíveis e reutilizáveis em diferentes partes do projeto, tornando o desenvolvimento mais eficiente e escalável.

3.4 BACK-END

Neste projeto, o *back-end* foi desenvolvido utilizando a linguagem NodeJS, que é amplamente utilizada no desenvolvimento de aplicações web, devido à sua eficiência e escalabilidade.

Para auxiliar no desenvolvimento do *back-end*, foram utilizadas algumas bibliotecas importantes. A biblioteca *Express*, por exemplo, é uma das mais populares em NodeJS e foi utilizada para criar o servidor HTTP que recebe as requisições do *front-end* e gerencia as rotas e *endpoints* da aplicação.

Outra biblioteca importante utilizada foi o *Sequelize*, que é um mapeamento objeto relacional (do inglês, *Object-Relational Mapping* - ORM) para NodeJS. Essa biblioteca permite a conexão com o banco de dados MySQL, de forma simples e intuitiva, além de possibilitar a criação de modelos e relacionamentos entre as tabelas do banco de dados.

Para gerenciar a imagem do banco de dados MySQL, foi utilizada a tecnologia Docker. Com ela, foi possível criar um ambiente isolado para o banco de dados, garantindo maior segurança e facilidade na configuração do banco de dados em diferentes ambientes de desenvolvimento e produção.

Além disso, foram utilizadas algumas bibliotecas de segurança e autenticação, como *Bcrypt*, que é uma biblioteca de *hash* de senha que garante maior segurança na armazenamento de senhas de usuários no banco de dados. O *Jsonwebtoken* foi utilizado para gerar e validar *tokens* de autenticação, enquanto o *Nodemailer* foi utilizado para enviar emails para os usuários em diferentes partes da aplicação.

Em resumo, o *back-end* da aplicação foi desenvolvido com a utilização de diversas bibliotecas que garantem segurança, eficiência e escalabilidade. Com elas, foi possível criar uma aplicação robusta e funcional, que atende às necessidades dos usuários de forma confiável e eficiente.

3.5 SERVIÇO DO YSERA

Nessa parte do sistema, foi desenvolvida uma aplicação em Python que tem como objetivo realizar o processamento dos arquivos PDB pelo Ysera. Essa aplicação conta com um *endpoint*, que recebe as configurações informadas pelo usuário, juntamente com o arquivo PDB a ser processado.

Para garantir maior eficiência no processamento dos arquivos, foi utilizada a biblioteca *threading* do Python. Essa biblioteca permite que várias *threads* sejam executadas simultaneamente, o que é essencial em aplicações que precisam lidar com grande volume de processamento.

Além disso, a aplicação Python foi desenvolvida utilizando o *framework* Flask. Esse *framework* é conhecido por ser leve e flexível, sendo uma excelente escolha para o desenvolvimento de aplicações web em Python. Ele é capaz de gerenciar as rotas e *endpoints* da aplicação de forma eficiente, além de contar com diversas extensões, que facilitam a implementação de recursos como: autenticação, segurança e comunicação com bancos de dados.

Após o processamento do arquivo PDB, a aplicação Python dispara um POST para o *back-end* da aplicação em NodeJS, enviando os resultados do processamento. Essa comunicação é feita de forma segura e eficiente, garantindo que os resultados sejam corretamente armazenados no banco de dados da aplicação e disponibilizados para consulta pelo usuário através do *front-end*. Em resumo, essa parte do sistema foi desenvolvida com o objetivo de garantir maior eficiência e confiabilidade no processamento dos arquivos PDB, utilizando recursos avançados do Python e do Flask para oferecer uma aplicação robusta e funcional.

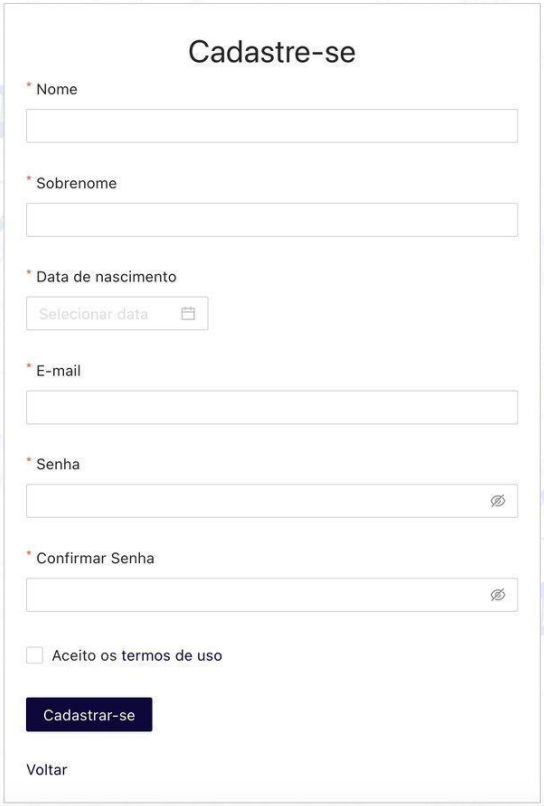
4 RESULTADOS E DISCUSSÃO

Ao final do desenvolvimento deste projeto, foi possível realizar predições através da plataforma, além de consultar os resultados obtidos através do login, realizado com e-mail e senha cadastrados anteriormente.

4.1 CADASTRO E LOGIN

O início da experiência do usuário na plataforma se dá com um cadastro a ser realizado, com poucos dados, a fim de autenticar o usuário (Figura 2).

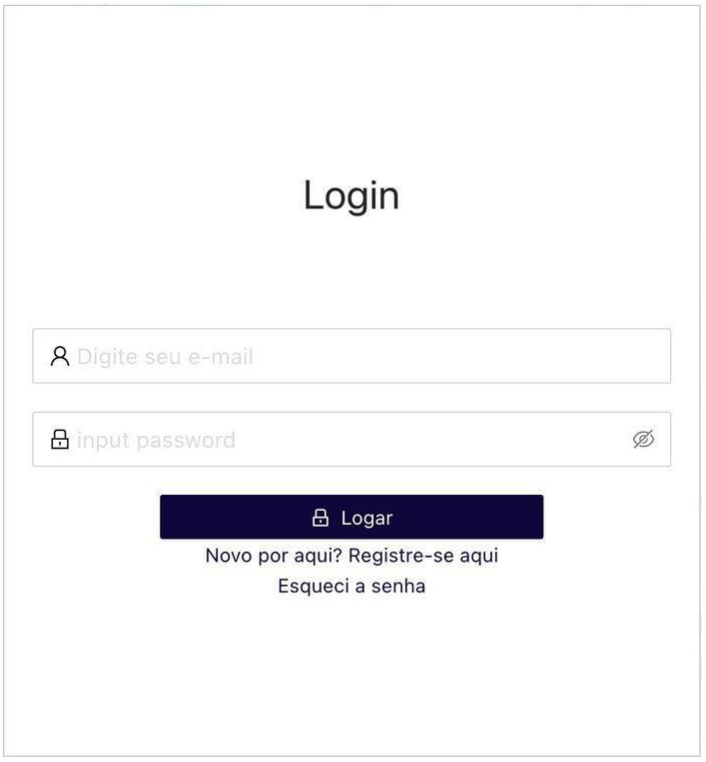
Figura 2 — Tela de Cadastro.



Fonte: Autoria própria (2023).

Usuários já cadastrados podem prosseguir diretamente para o login, apresentado na Figura 3.

Figura 3 — Tela de Login.



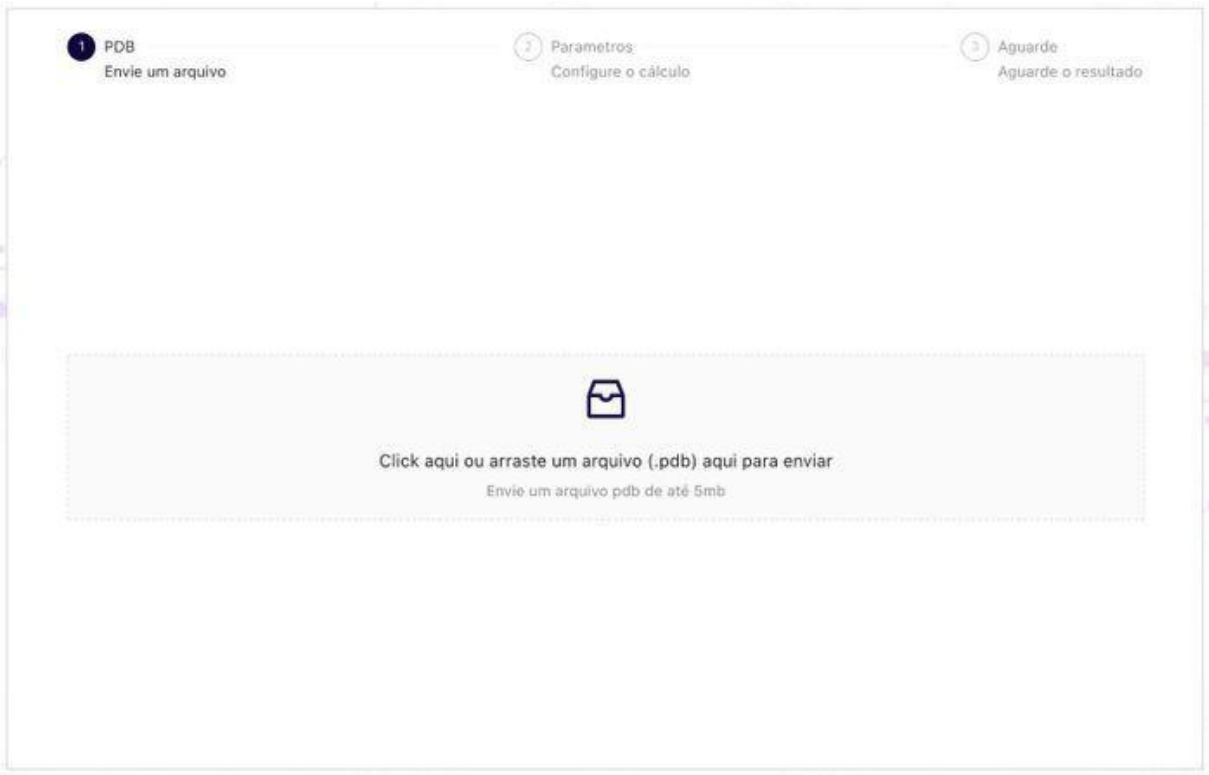
Fonte: Autoria própria (2023).

4.2 ENVIO DO ARQUIVO PDB

O processo de envio de arquivos PDB foi desmembrado em etapas, de forma mais pragmática. Além disso, as etapas, ainda que poucas, estão descritas em uma barra de navegação, mostrando a etapa atual e as restantes. Isso fornece, a quem utiliza, uma importante percepção de andamento da tarefa, evitando assim possíveis cancelamentos ao longo do processo.

A Figura 4 mostra a página na qual o usuário pode fazer o upload do arquivo PDB, por meio de um botão. ou simplesmente arrastando o arquivo para a página. Esta opção, fornece uma facilidade a mais ao usuário, principalmente quando se trata de usuários que utilizam-se de *touchpads* em notebooks, onde a função de arrastar se torna difícil.

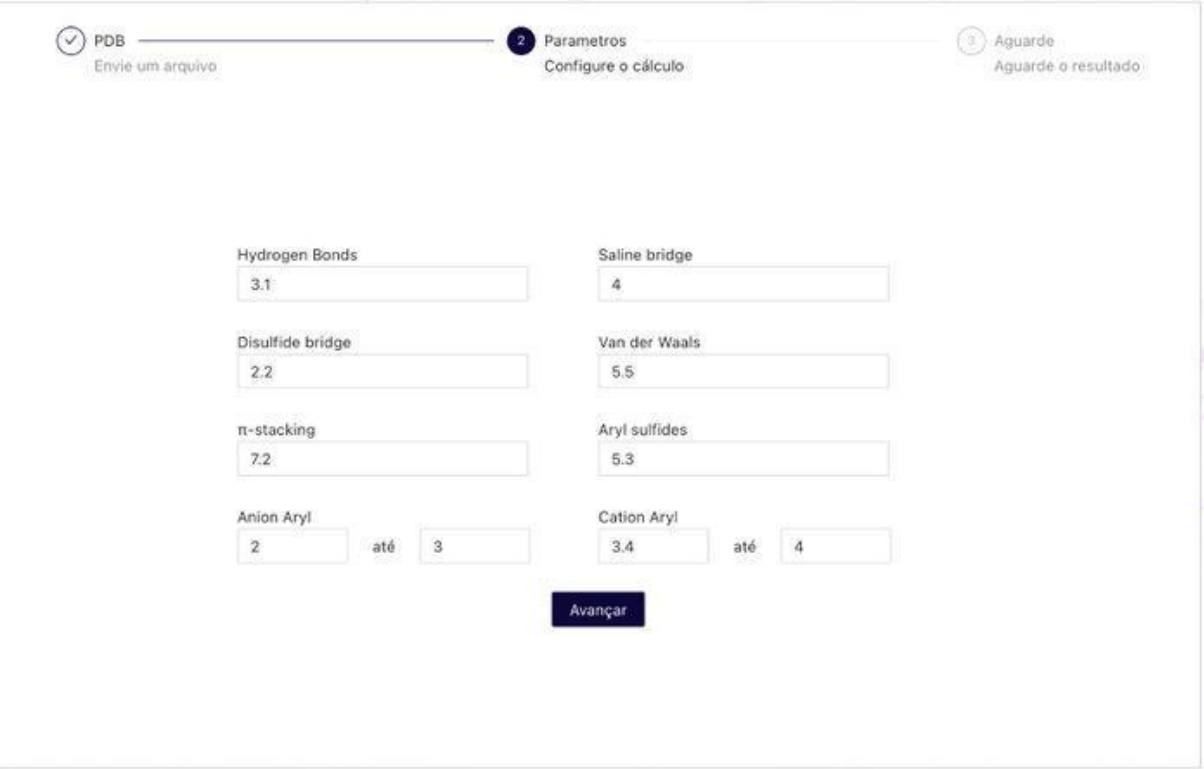
Figura 4 — Envio do arquivo PDB.



Fonte: Autoria própria (2023).

Em seguida, a plataforma solicita ao usuário que forneça os parâmetros para o cálculo das distâncias entre átomos e a identificação de possíveis interações entre eles. Os campos já carregam parâmetros padrões, de modo que, caso queira, basta ao usuário clicar no botão de avançar para submeter o arquivo para o cálculo, conforme podemos ver na Figura 5.

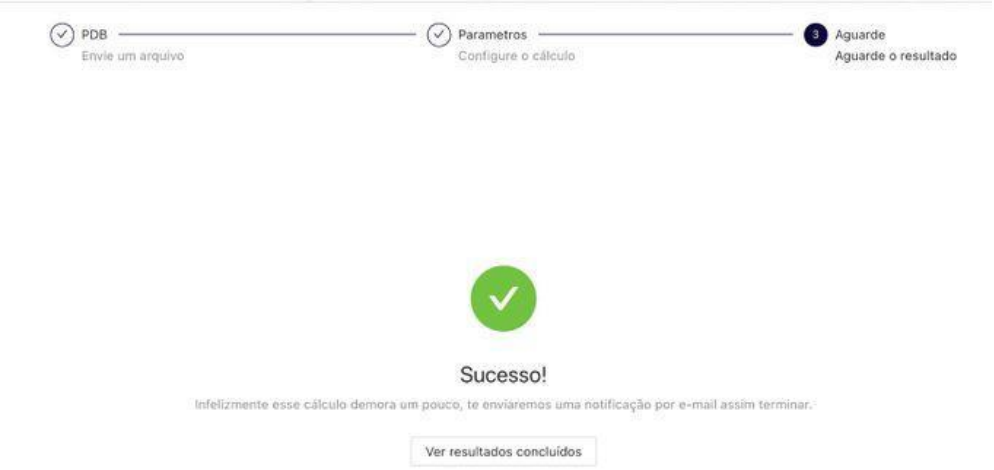
Figura 5 — Configuração dos parâmetros.



Fonte: Autoria própria (2023).

Finalmente, a plataforma apresenta uma mensagem amigável (Figura 6), explicando que o processo foi bem sucedido e será enviado um e-mail com a sua conclusão.

Figura 6 — Mensagem final.

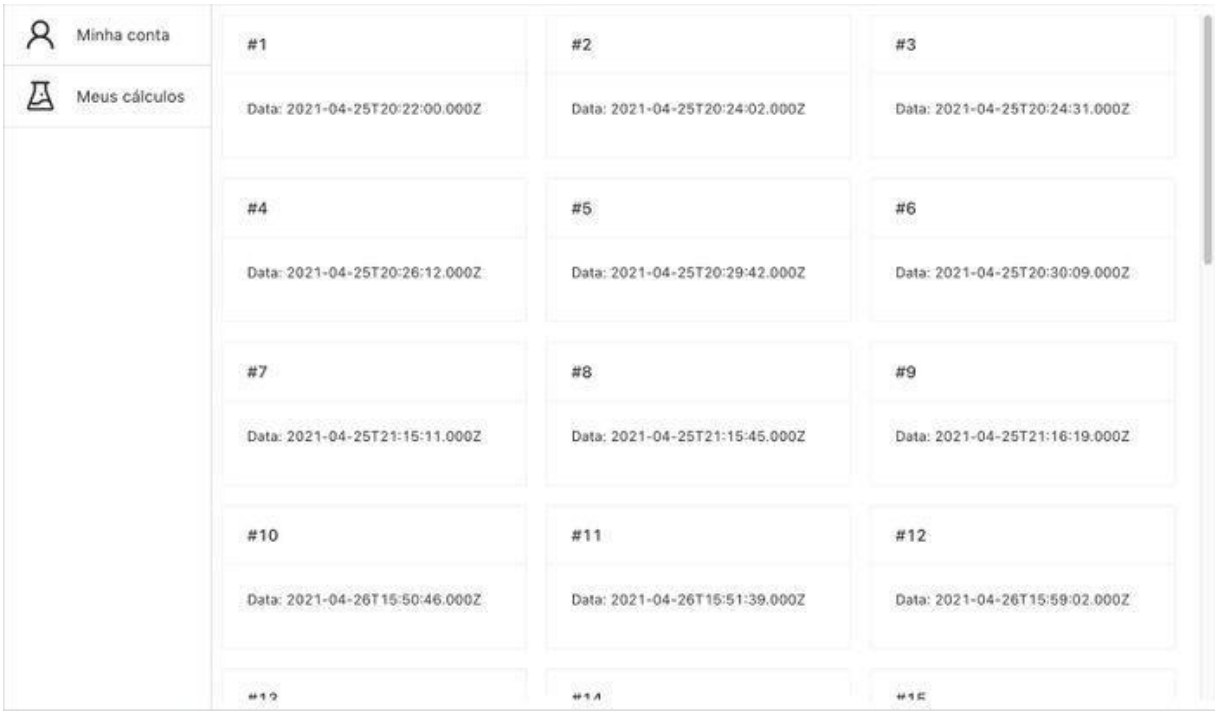


Fonte: Autoria própria (2023).

4.3 VISUALIZAÇÃO DE RESULTADOS

Após isso, é possível ver uma lista com todos os resultados anteriormente realizados (Figura 7). Isso auxilia o usuário, pois evita a preocupação em armazenar esses dados, além de poder acessar de qualquer lugar.

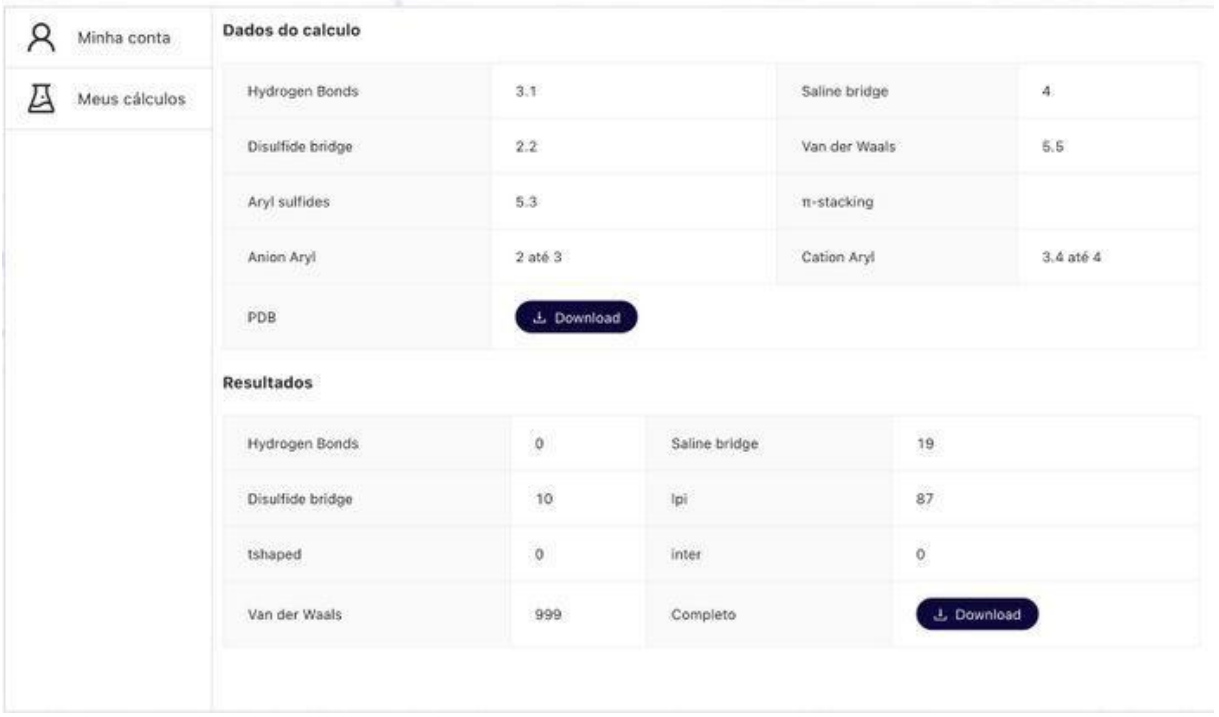
Figura 7 —E xemplo da lista de cálculos solicitados pelo usuário.



Fonte: Autoria própria (2023).

Ao selecionar qualquer um dos cálculos, é apresentada uma tela com os resultados obtidos, além de botões para *download* do PDB enviado e um arquivo de texto em formato (.txt) com o resultado completo (Figura 8).

Figura 8 — Resultado.



Fonte: Autoria própria (2023).

4.4 COMPARANDO OS RESULTADOS OBTIDOS

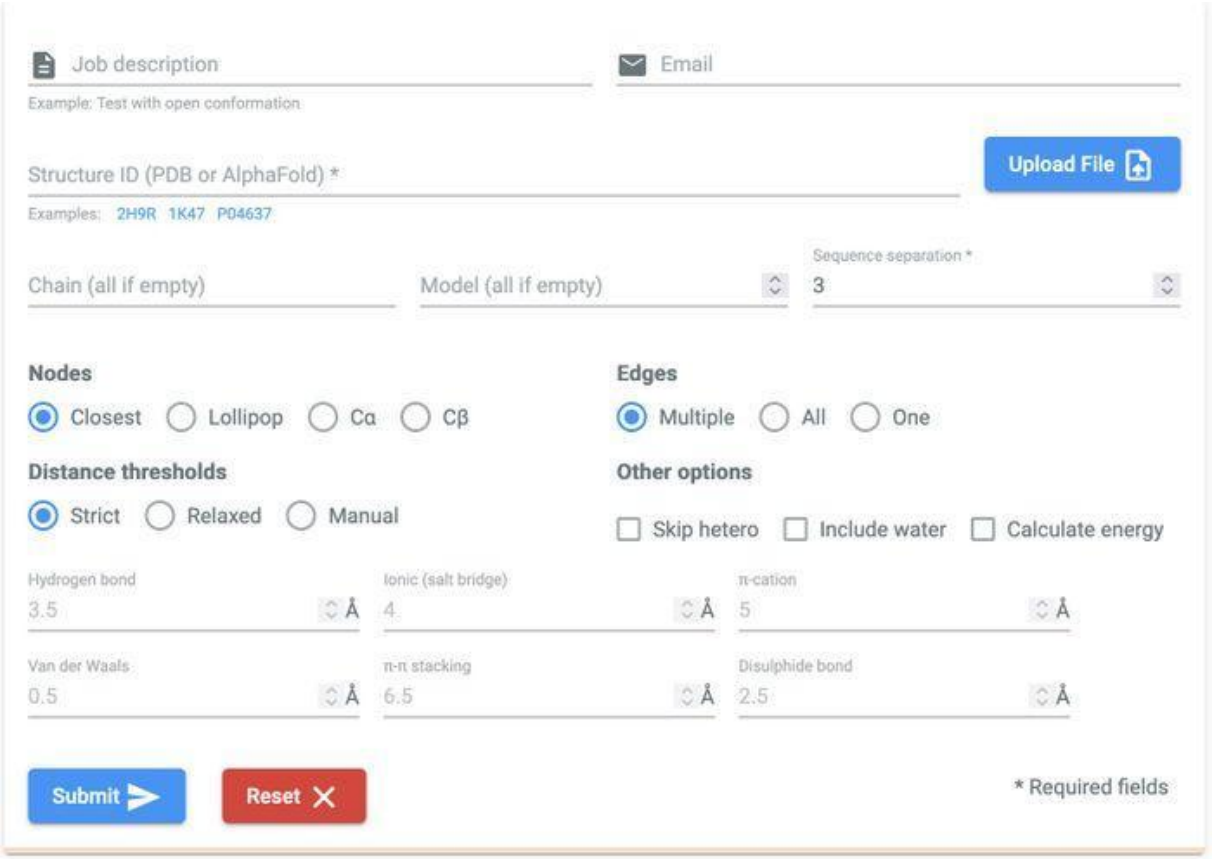
É possível comparar os resultados obtidos com uma outra plataforma semelhante, obtida no link https://ring.biocomputingup.it/submit. Este site fornece também um login ao usuário, porém o usuário precisa possuir cadastro no OrcID. Isto dificulta o acesso de alguns usuário, como estudantes que ainda não possuem identificador desse tipo, por exemplo. A plataforma também permite que o usuário acesse cálculos realizados anteriormente, mas apenas na sessão atual do navegador. Nesse caso, o usuário não conseguiria acessar o histórico em um outro dispositivo.

Uma outra vantagem da plataforma aqui desenvolvida, é separar o envio em etapas, conforme dito anteriormente. Isto não foi observado neste site, como pode-se ver na Figura 9.

Porém, é possível perceber que o resultado do RING apresenta opções de parametrização que não são encontradas no YSERA, como d*istance thresholds e edges*. Estas opções de parametrização conseguem controlar se o RING irá considerar as distancias de modo mais rigoroso, relaxado ou manual e se o software retorna uma ou múltiplas interações entre um par de nós.

Além disso, o RING consegue apresentar o resultado em uma visão tridimensional, que se encontra como parte dos objetivos futuros da plataforma do YSERA, além da inclusão de mais opções de parametrização.

Figura 9 — RING.



Fonte: RING (2023).

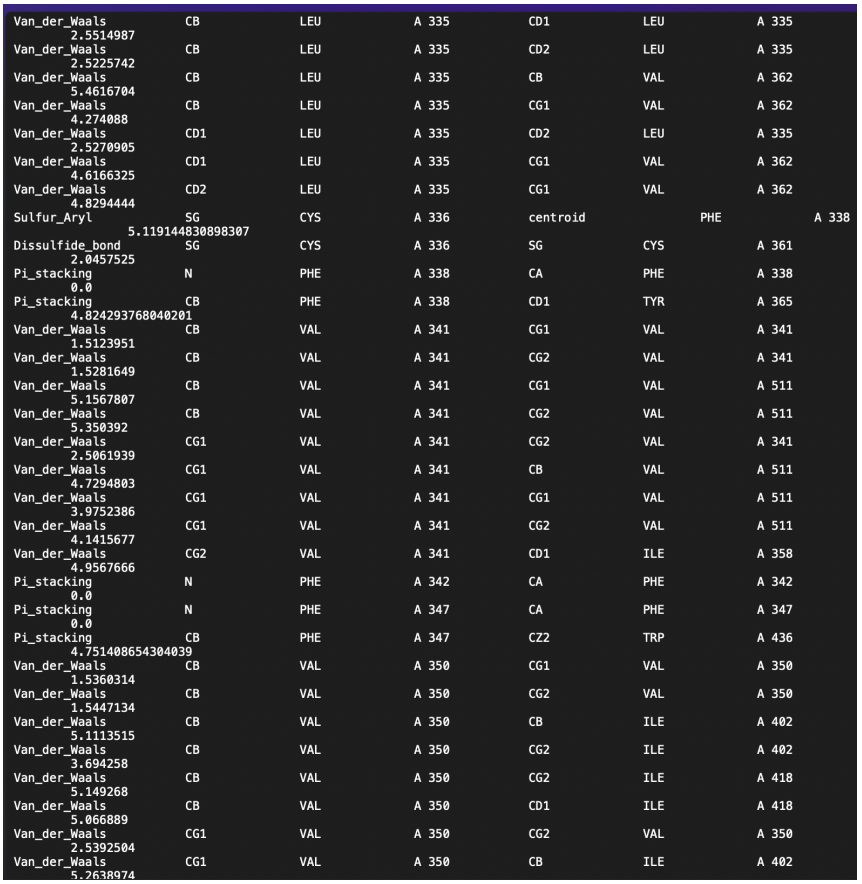
A Figura 10 apresenta o início de um arquivo típico no formato PDB, como um exemplo de entrada na plataforma, para predição no YSERA. Já a Figura 11, apresenta um trecho do arquivo de texto resultado do processamento. Este arquivo apresenta a contagem de interações encontradas de tipo como as interações de Van Der Waals, como a apresentada na imagem.

Figura 10 — Arquivo PDB.fOJTE:

```
HEADER    VIRAL PROTEIN/IMMUNE SYSTEM            26-APR-20   7BZ5
TITLE     STRUCTURE OF COVID-19 VIRUS SPIKE RECEPTOR-BINDING DOMAIN COMPLEXED
TITLE    2 WITH A NEUTRALIZING ANTIBODY
COMPND    MOL_ID: 1;
COMPND   2 MOLECULE: SPIKE PROTEIN S1;
COMPND   3 CHAIN: A;
COMPND   4 SYNONYM: S GLYCOPROTEIN,E2,PEPLOMER PROTEIN;
COMPND   5 ENGINEERED: YES;
COMPND   6 MOL_ID: 2;
COMPND   7 MOLECULE: HEAVY CHAIN OF B38;
COMPND   8 CHAIN: H;
COMPND   9 ENGINEERED: YES;
COMPND  10 MOL_ID: 3;
COMPND  11 MOLECULE: LIGHT CHAIN OF B38;
COMPND  12 CHAIN: L;
COMPND  13 ENGINEERED: YES
SOURCE    MOL_ID: 1;
SOURCE   2 ORGANISM_SCIENTIFIC: SEVERE ACUTE RESPIRATORY SYNDROME CORONAVIRUS
SOURCE   3 2;
SOURCE   4 ORGANISM_COMMON: 2019-NCOV;
SOURCE   5 ORGANISM_TAXID: 2697049;
SOURCE   6 GENE: S, 2;
SOURCE   7 EXPRESSION_SYSTEM: SPODOPTERA FRUGIPERDA;
SOURCE   8 EXPRESSION_SYSTEM_TAXID: 7108;
SOURCE   9 MOL_ID: 2;
SOURCE  10 ORGANISM_SCIENTIFIC: HOMO SAPIENS;
SOURCE  11 ORGANISM_TAXID: 9606;
SOURCE  12 EXPRESSION_SYSTEM: HOMO SAPIENS;
SOURCE  13 EXPRESSION_SYSTEM_TAXID: 9606;
SOURCE  14 MOL_ID: 3;
SOURCE  15 ORGANISM_SCIENTIFIC: HOMO SAPIENS;
SOURCE  16 ORGANISM_TAXID: 9606;
SOURCE  17 EXPRESSION_SYSTEM: HOMO SAPIENS;
SOURCE  18 EXPRESSION_SYSTEM_TAXID: 9606
KEYWDS    COVID-19 ANTIBODY, VIRAL PROTEIN, VIRAL PROTEIN-IMMUNE SYSTEM COMPLEX
EXPDTA    X-RAY DIFFRACTION
AUTHOR    Y.WU,J.QI,F.GAO
REVDAT   2   27-MAY-20 7BZ5    1       JRNL
REVDAT   1   13-MAY-20 7BZ5    0
JRNL        AUTH   Y.WU,F.WANG,C.SHEN,W.PENG,D.LI,C.ZHAO,Z.LI,S.LI,Y.BI,Y.YANG,
JRNL        AUTH 2 Y.GONG,H.XIAO,Z.FAN,S.TAN,G.WU,W.TAN,X.LU,C.FAN,Q.WANG,
JRNL        AUTH 3 Y.LIU,C.ZHANG,J.QI,G.F.GAO,F.GAO,L.LIU
JRNL        TITL   A NONCOMPETING PAIR OF HUMAN NEUTRALIZING ANTIBODIES BLOCK
JRNL        TITL 2 COVID-19 VIRUS BINDING TO ITS RECEPTOR ACE2.
JRNL        REF    SCIENCE                                       2020
JRNL        REFN                   ESSN 1095-9203
JRNL        PMID   32404477
JRNL        DOI    10.1126/SCIENCE.ABC2241
REMARK   2
```

Fonte: Autoria própria (2023).

Figura 11 — Resultado completo



```
Van_der_Waals           CB              LEU             A 335           CD1             LEU             A 335
        2.5514987
Van_der_Waals           CB              LEU             A 335           CD2             LEU             A 335
        2.5225742
Van_der_Waals           CB              LEU             A 335           CB              VAL             A 362
        5.4616704
Van_der_Waals           CB              LEU             A 335           CG1             VAL             A 362
        4.274088
Van_der_Waals           CD1             LEU             A 335           CD2             LEU             A 335
        2.5270905
Van_der_Waals           CD1             LEU             A 335           CG1             VAL             A 362
        4.6166325
Van_der_Waals           CD2             LEU             A 335           CG1             VAL             A 362
        4.8294444
Sulfur_Aryl             SG              CYS             A 336           centroid                PHE             A 338
                5.119144830898307
Dissulfide_bond         SG              CYS             A 336           SG              CYS             A 361
        2.0457525
Pi_stacking             N               PHE             A 338           CA              PHE             A 338
        0.0
Pi_stacking             CB              PHE             A 338           CD1             TYR             A 365
        4.8242937680402O1
Van_der_Waals           CB              VAL             A 341           CG1             VAL             A 341
        1.5123951
Van_der_Waals           CB              VAL             A 341           CG2             VAL             A 341
        1.5281649
Van_der_Waals           CB              VAL             A 341           CG1             VAL             A 511
        5.1567807
Van_der_Waals           CB              VAL             A 341           CG2             VAL             A 511
        5.350392
Van_der_Waals           CG1             VAL             A 341           CG2             VAL             A 341
        2.5061939
Van_der_Waals           CG1             VAL             A 341           CB              VAL             A 511
        4.7294803
Van_der_Waals           CG1             VAL             A 341           CG1             VAL             A 511
        3.9752386
Van_der_Waals           CG1             VAL             A 341           CG2             VAL             A 511
        4.1415677
Van_der_Waals           CG2             VAL             A 341           CD1             ILE             A 358
        4.9567666
Pi_stacking             N               PHE             A 342           CA              PHE             A 342
        0.0
Pi_stacking             N               PHE             A 347           CA              PHE             A 347
        0.0
Pi_stacking             CB              PHE             A 347           CZ2             TRP             A 436
        4.751408654304039
Van_der_Waals           CB              VAL             A 350           CG1             VAL             A 350
        1.5360314
Van_der_Waals           CB              VAL             A 350           CG2             VAL             A 350
        1.5447134
Van_der_Waals           CB              VAL             A 350           CB              ILE             A 402
        5.1113515
Van_der_Waals           CB              VAL             A 350           CG2             ILE             A 402
        3.694258
Van_der_Waals           CB              VAL             A 350           CG2             ILE             A 418
        5.149268
Van_der_Waals           CB              VAL             A 350           CD1             ILE             A 418
        5.066889
Van_der_Waals           CG1             VAL             A 350           CG2             VAL             A 350
        2.5392504
Van_der_Waals           CG1             VAL             A 350           CB              ILE             A 402
        5.2638974
```

Fonte: Autoria própria (2023).

5 CONCLUSÃO

Neste trabalho, foi desenvolvida uma plataforma web para a predição de interações atômicas de estruturas tridimensionais de proteínas com o uso do Ysera. Através da análise dos resultados obtidos e da avaliação dos requisitos funcionais e não-funcionais, é possível concluir que a plataforma desenvolvida é capaz de atender às necessidades dos usuários, garantindo a eficiência e qualidade na predição de interações atômicas.

Durante o desenvolvimento da plataforma, foram enfrentados desafios relacionados à escolha de tecnologias e estratégias de desenvolvimento que garantem o desempenho, segurança e usabilidade da aplicação. Foram adotados padrões de codificação para garantir a confiabilidade da plataforma.

Os resultados obtidos mostram que a plataforma é capaz de processar grandes quantidades de dados em tempo hábil, oferecendo uma interface intuitiva e de fácil utilização para os usuários. Além disso, a plataforma atende a requisitos não-funcionais como segurança, escalabilidade e performance, garantindo a qualidade e eficiência da predição de interações atômicas de estruturas tridimensionais de proteínas.

A plataforma web desenvolvida neste trabalho pode ser uma ferramenta importante para pesquisadores na área de biologia estrutural, permitindo a predição de interações atômicas de estruturas tridimensionais de proteínas de maneira rápida e eficiente. Além disso, a plataforma pode ser uma contribuição para a comunidade científica, possibilitando a validação de resultados obtidos por outras técnicas experimentais.

Em suma, este trabalho apresenta uma solução viável e eficiente para a predição de interações atômicas de estruturas tridimensionais de proteínas com o uso do Ysera, além de contribuir para o avanço da ciência na área de biologia estrutural.

REFERÊNCIAS

ANDRADE, E. B. **Ysera:** Preditor de interações atômicas de estruturas tridimensionais de proteínas.58f. 2020. Monografia (Bacharelado em Biotecnologia), Centro de Biotecnologia, Universidade Federal da Paraíba, João Pessoa. 2020.

BAKER, R.. **Mastering React:** : Build robust and maintainable web apps with React 16.8 and beyond.. Birmingham: Packt Publishing, 2018.

CLEMENTEL, D. et al. RING 3.0: fast generation of probabilistic residue interaction networks from structural ensembles. Nucleic Acids Research, v. 50, n. W1, p. W651-W656, 2022.

FERNANDEZ, M. P..: **Rede de computadores**, 2019

MELL; GRANCE: **The NIST Definition of Cloud Computing,** 2011

WIRFS-BROCK, 2020: **JavaScript: the first 20 years, 2020:** Proc. ACM Program. Lang., Vol. 4, No. HOPL, Article 77.

PRESSMAN, R. S.. **Engenharia de Software:**: Uma abordagem profissional. 7 ed. Porto Alegre: AMGH, 2016.

RÊGO, T. G. **Preditor de Interações Atômicas de Estruturas Tridimensionais de Proteínas**. Desenvolvimento de software para análise de interações atômicas de proteínas com mutações deletérias e não deletérias para a previsão de câncer. Programa Institucional de Bolsas de Iniciação Científica (PIBIC/CNPq/UFPB) e de voluntários de iniciação científica PIVIC). 2020.

RING. Disponível em: https://ring.biocomputingup.it/submit. Acesso em: 14 jun. 2023.

SCHAUERMAN, J.. **React:**: Up & Running. Sebastopol: O'Reilly Media, 2020.

SOMMERVILLE, I.. **Engenharia de Software**. 9 ed. São Paulo: Pearson Addison Wesley, 2011.

TANENBAUM, A. S.: **Redes de Computadores**. 4 ed. Editora Campus, 2003