UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Abraão Homualdo Alves Moreira

Plataforma Semana da Computação: Gerenciamento de Eventos acadêmicos do CI, com verificação através de Geolocalização e automação de Certificados

JOÃO PESSOA
2024

Abraão Homualdo Alves Moreira

PLATAFORMA SEMANA DA COMPUTAÇÃO: GERENCIAMENTO DE EVENTOS ACADÊMICOS DO CI, COM VERIFICAÇÃO ATRAVÉS DE GEOLOCALIZAÇÃO E AUTOMAÇÃO DE CERTIFICADOS

Relatório Técnico apresentado à Banca Examinadora da Universidade Federal da Paraíba, como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação sob orientação do Prof. Raoni Kulesza

JOÃO PESSOA
2024

Catalogação na publicação
Seção de Catalogação e Classificação

M838pp Moreira, Abraao Homualdo Alves.
Plataforma semana da computação: gerenciamento de eventos acadêmicos do CI, com verificação através de geolocalização e automação de certificados / Abraao Homualdo Alves Moreira. - João Pessoa, 2024.
42 f. : il.

Orientação: Raoni Kulesza.
TCC (Graduação) - UFPB/Informática.

1. Aplicações WEB. 2. NextJS. 3. Frameworks. 4. Plataforma de eventos. 5. React. 6. NodeJS. 7. Semana da Computação. I. Raoni Kulesza. II. Título.

UFPB/CI CDU 004.774

Elaborado por Michelle de Kássia Fonseca Barbosa - CRB-738

UNIVERSIDADE FEDERAL DA PARAÍBA
CIÊNCIA DA COMPUTAÇÃO

COORDENAÇÃO DO CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

O relatório técnico "Plataforma Semana da Computação: Gerenciamento de Eventos acadêmicos do CI, com verificação através de Geolocalização e automação de Certificados", apresentado e defendido por ABRAÃO HOMUALDO ALVES MOREIRA, matrícula 20200095558, foi aprovada pela Banca Examinadora constituída pelos seguintes professores:

Examinado por:

Prof. Dr. Raoni Kulesza
Orientador

Profa. Dra. Thaís Gaudêncio
Universidade Federal da Paraíba - UFPB

Prof. Dr. Marcelo Iury
Universidade Federal da Paraíba - UFPB

João Pessoa, 19 de novembro de 2024

Este trabalho é dedicado totalmente a Deus, que habitou em mim com Seu Espírito Santo em todos os momentos, concedendo-me sabedoria e discernimento nesta longa jornada acadêmica. Dedico também aos meus pais, Francisco Marinho Alves Irmão e Maria Edileusa Moreira, cujos sacrifícios e sonhos renunciados edificaram minha criação e me permitiram trilhar meu próprio caminho. À minha amada, Vitória Helen de Sousa de Lima, cujo amor foi o equilíbrio perfeito entre ser meu futuro e me desafiar a ser alguém melhor, minha eterna gratidão.

Agradecimentos

A Deus, acima de todas as coisas, que sempre me proporcionou sabedoria para lutas diárias, fortalecendo-me espiritualmente e fisicamente.

À minha amada, Vitória Helen de Sousa de Lima, por ser o equilíbrio perfeito entre ser meu futuro e me desafiar a ser alguém melhor. Obrigado por ser a razão pela qual busco um futuro promissor, para que possamos construir nossa família juntos. A tua presença foi o conforto nos dias difíceis, a âncora que me manteve firme nas tempestades, e o sorriso que ilumina minha **vitória**.

"Tudo o que vier às tuas mãos para fazer, faze-o conforme as tuas forças..." **(Eclesiastes 9:10)**. Este versículo foi a base da minha motivação na vida acadêmica e em competições de karatê, lembrando-me sempre de dar o meu melhor em todas as ocasiões.

"Eu acredito que às vezes são as pessoas que ninguém espera nada que fazem as coisas que ninguém consegue imaginar." (Alan Turing)

Resumo

Desde o início das aplicações web, os desenvolvedores têm buscado aperfeiçoamentos no desenvolvimento de novos *recursos* desses softwares, gerando uma cadeia de evolução das ferramentas disponíveis no mercado. Entretanto, aplicações que não são desenvolvidas para casos específicos enfrentam o desafio de se manter no mercado, especialmente quando envolvem esses casos mais específicos. No presente relatório, objetivou-se o desenvolvimento de uma plataforma de eventos que foi desenvolvida com *os frameworks* React (NextJS) e NodeJs (NestJS). Foram apresentados exemplos comparativos com plataformas existentes no mercado, como a Even3, mostrando como a plataforma Semana da Computação dispõe de novas funcionalidades de verificação, utilizando geolocalização e autenticação através de armazenamento em *LocalStorage*. Tal aplicação consiste numa plataforma WEB desenvolvida por 3 alunos do PET Computação (Abraão Moreira, Aran Leite e Lucas Garrafielo) do Curso de Ciência da Computação do Centro de Informática da Universidade Federal da Paraíba (UFPB) . Inicialmente foram realizadas análises de segurança, inspeção e desempenho do Even3, e visto que haviam necessidades a serem supridas, surgiu a ideia da criação de um sistema de gerenciamento de eventos, denominado Plataforma SDC. Posteriormente, foram definidas as etapas e *sprints* de desenvolvimento e produção da aplicação, garantindo que as técnicas avançadas de autenticação fossem efetivadas e gerasse benefício para toda comunidade. Como resultado, a plataforma não só aumentou a credibilidade e a transparência na gestão de eventos, mas também facilitou a organização e a verificação da presença de participantes.

Palavras-chave: Aplicações WEB, NextJS, NestJS, Frameworks, Plataforma de Eventos, React, NodeJS, Semana da Computação.

Abstract

Since the early days of web applications, developers have sought improvements in developing new *features* for these softwares, generating a chain of evolution of the tools available in the market. However, applications that are not developed for specific use cases face the challenge of remaining competitive, especially when they involve more specific scenarios. In this report, the objective was to develop an event platform using the *frameworks* React (NextJS) and NodeJs (NestJS). Comparative examples were presented with existing platforms in the market, such as Even3, showing how the Semana da Computação platform offers new verification features using geolocation and authentication through *LocalStorage*. This application is a web platform developed by 3 students from PET Computação of the Computer Science Course at the Center of Informatics at UFPB (Abraão Moreira, Aran Leite, and Lucas Garrafielo). Initially, security, inspection, and performance analyses were carried out on Even3, and seeing that there were needs to be addressed, the idea of creating an event management system, called Plataforma SDC, emerged. Subsequently, the development and production stages and *sprints* of the application were defined, ensuring that advanced authentication techniques were implemented to benefit the entire community. As a result, the platform not only increased credibility and transparency in event management but also facilitated the organization and verification of participant attendance.

Keywords: Web Applications, Next.js, NestJS, Frameworks, Event Platform, React, NodeJS, Computer Science Week.

Lista de Figuras

Lista de Abreviaturas

- **API**: *Application Programming Interface*
- **APP**: Aplicativo
- **SDC**: Semana da Computação
- **JS**: *JavaScript*
- **TS**: *TypeScript*
- **CI**: Centro de Informática
- **SSR**: *Server Side Rendering*
- **SPA**: *Single Page Application*
- **DDD**: *Domain-Driven Design*
- **CSS**: *Cascading Style Sheets*
- **NPM**: *Node Package Manager*
- **HTML**: *HyperText Markdown*
- **UI**: *User Interface*
- **JSON**: *JavaScript Object Notation*
- **JSX**: *JavaScript XML*
- **DOM**: *Document Object Model*
- **SEO**: *Search Engine Optimization*
- **ORM**: *Object Relational Mapper*
- **IDE**: *Integrated Development Environment*
- **BD**: Banco de Dados

- **SGBD**: Sistema de Gerenciamento de Banco de Dados
- **UUID**: *Universally Unique Identifier*
- **TDD**: *Test Driven Development*

Sumário

Capítulo 1

Introdução

A Semana da Computação do Centro de Informática da UFPB, que ocorre no início de cada semestre, tem como objetivo promover o aprendizado e o networking entre discentes e profissionais da área de TI. O evento é organizado pelo Programa de Educação Tutorial (PET), um programa desenvolvido por estudantes em grupos tutoriais de aprendizagem, fundado pelo Professor Leonardo Vidal Batista e orientado pelo Professor Ed Porto. O PET é estruturado em nível de graduação e consolida ações extracurriculares, orientadas pelo princípio da indissociabilidade entre ensino, pesquisa e extensão.

No entanto, um dos grandes desafios enfrentados por essa iniciativa é garantir a presença efetiva dos participantes. Historicamente, o registro de presença sempre foi um problema, com questões como a dificuldade de verificar a presença dos discentes durante os eventos e autenticidade de seus certificados.

A falta de um sistema eficiente e seguro de verificação torna possível que discentes obtenham certificados sem realmente participar das atividades, comprometendo a credibilidade do evento. Essa situação não apenas prejudica a qualidade da entrega de conteúdo, mas também afeta a confiança na validade dos certificados emitidos.

Diante desse cenário, surge a necessidade de implementar uma plataforma que trate esses problemas, garantindo um registro de presença eficaz e tenham certificados de forma injusta. Isso é essencial para assegurar a integridade do evento, proporcionar um ambiente de aprendizado real e fomentar a participação ativa dos discentes.

1.1 Tema

Os recursos de plataformas WEB passam por atualizações contínuas, com versões anteriores se tornando obsoletas e novos aplicativos trazendo aprimoramentos significativos em funcionalidades, suporte, manutenção de código, segurança e outros benefícios. O presente trabalho tem como objetivo desenvolver uma plataforma WEB dedicada a promover eventos acadêmicos, especialmente a Semana da Computação (SDC), que ocorre no início de cada semestre, com duração de 4 a 5 dias e capacidade de atender

entre 200 à 300 pessoas. A primeira edição do evento ocorreu em 2012. A plataforma SDC, visa reduzir as taxas de evasão de discentes em eventos futuros por meio de validações de localização ou de rede acadêmica.

1.2 Problema

Um dos grandes desafios das plataformas de eventos é garantir a presença dos discentes, evitando situações de registro de presença sem efetiva participação. A falta de validações externas em relação à localização do discente gera riscos, como a possibilidade de falsificação de presença.

A plataforma Even3[1], utilizada como objeto de estudo, não implementa nenhuma tratativa para assegurar a presença dos participantes. Como resultado, a utilização do Even3 para promover eventos no Centro de Informática da UFPB gera uma série de problemas, tais como:

- Riscos de Segurança: A emissão dos certificados é exigida pós evento como justificativa para substituir faltas em aulas, que podem acontecer durante o evento, diante disso, para a emissão não há nenhum tipo de gerenciamento de faltas, exigindo apenas que o usuário espere o evento se encerrar, para sua emissão.

- Otimização e Recursos Adicionais: Um dos problemas recorrentes, era a emissão dos certificados. Como a equipe organizadora do evento era pequena (2 pessoas), a emissão ficava consgestionada, pois era feita utilizando um script que gerava através de uma tabela mapeada com base nos inscritos, demorando mais de um mês para entrega final.

A migração para uma nova plataforma ou a implementação de novas medidas são passos essenciais para evitar esses riscos, garantindo a credibilidade do evento e a satisfação dos participantes.

1.3 Objetivo Geral

O objetivo deste trabalho foi a implementação de uma nova Plataforma de Eventos para promover eventos da Semana da Computação. Essa migração teve como propósito a modernização dos eventos, assegurando a conformidade, segurança e credibilidade. Além disso, essa implementação fornece o acompanhamento de monitoramento de toda programação do evento em tempo real.

[1]Even3 é uma plataforma criada para simplificar a organização de eventos técnicos e científicos como congressos, workshops, simpósios e minicursos. Disponível em https://www.even3.com.br/

1.4 Objetivos Específicos

Em relação aos objetivos específicos, este trabalho realizou:

- Identificar e especificar requisitos funcionais e não funcionais da plataforma, bem como a arquitetura geral da aplicação.
- Apresentar as soluções criadas ao problema, mediante propostas de autenticidade, verificações de localização para participantes que queiram burlar a presença no evento e lidar com grande volume de requisições durante o evento.
- Avaliar as percepções pós evento, realizando uma análise comparativa com os outros anos.

1.5 Estrutura do Relatório Técnico

No Capítulo 2, são apresentadas as fundamentações teóricas do trabalho. Logo após, tem-se o Capítulo 3, que aborda a metodologia utilizada, descrevendo as tecnologias empregadas, os usuários envolvidos, os colaboradores e a estrutura de hospedagem, além de detalhar os requisitos funcionais e não funcionais da aplicação, a interface e a arquitetura, a fim de servir como medida comparativa para o presente estudo. Em seguida, no Capítulo 4, a Plataforma SDC é apresentada e detalhada por meio da especificação dos requisitos e da modelagem da arquitetura. Por fim, no último capítulo (quinto), são apresentados *insights* de acesso durante o evento e a conclusão da implementação, resumindo os resultados alcançados.

Capítulo 2

Conceitos Gerais

Neste capítulo, são descriminados os principais conceitos das ferramentas usadas, que fazem parte da construção da aplicação.

2.1 JavaScript

JavaScript (JS) é uma das linguagens mais populares[5], para desenvolvimento WEB, destacando-se por sua leveza, baseada em protótipos, multiparadigma e dinâmica, suportando estilos de orientação a objetos, imperativos e declarativos. Atualmente, está contida na tríade de desenvolvimento WEB, junto com HTML (*Hypertext Markup Language*) e CSS (*Cascading Style Sheets*).

O JS proporciona *frameworks* e bibliotecas, como React, Angular, Vue.js, otimizando o desenvolvimento de aplicações *frontend*, enquanto Express, Fastify, NestJS são utilizados para o *backend*. Além disso, a comunidade da linguagem desenvolve milhares de módulos, dentro do maior repositório de pacotes o NPM (*Node Package Manager*).

2.2 Frontend

O *frontend* é a forma mais chamada a parte visual de uma aplicação[9]. Formada pela junção da tríade (HTML, CSS e JS), os dois primeiros são responsáveis pela *"pintura da casa"*, e através do JavaScript, o usário é capaz de interagir, vendo *"comportamentos da casa"*, de acordo com suas requisições.

2.3 Backend

Por outro lado, o *backend* é a "sustentação da casa" , tradicionalmente chamado como servidor da aplicação. O destino das requisições são sempre enviadas para essa área, responsável por recebimento de dados, manipulação dos dados e uma resposta

que é devolvida para usuário[2], habitualmente , no formato JSON (*JavaScript Object Notation*).

2.4 ReactJS

ReactJS é uma biblioteca JavaScript de código aberto, mantida pelo Meta e por uma enorme comunidade, voltada principalmente para o desenvolvimento de interfaces de usuário (UI)[3].

O conceito de Virtual DOM (*Document Object Model*) foi introduzida no ambiente React, onde assegura uma representação leve de interfaces em memória. Ao invés de ser atualizado o DOM real do navegador do usuário, sempre quando houver mudança na interface (ineficiente), o React faz comparações constantes de atualização da versão antiga para nova do Virtual DOM, reduzindo custos desnecessários e melhorando o desempenho da aplicação.

O React permite o uso de JSX (JavaScript XML)[10], onde, resumidamente, conteúdos de linguagem de marcação podem ser inseridos em blocos de Javascript, combinados em um único arquivo. Há facilitação de escrita de componentes, sendo definidas estrutura e lógica comportamental, simultaneamente. Em critérios de estilização, atualmente já existem inúmeras formas de serem inseridas, desde componentização de estilos (*Styled Components*), até pré processadores de CSS (*Sass Stylesheet*), permitindo modularidade através de arquivos individuais de estilização.

2.5 NodeJS

A necessidade de uma solução poderosa e barata para criação e manutenção de ambientes com altas demandas permitiu o nascimento do NodeJS, que é um ambiente de execução de JavaScript pelo lado do servidor (*Server side*). Foi criado, destacando-se pela arquitetura orientada a objetos e execuções assícronas, permitindo o desenvolvimento de soluções escaláveis e de alto desempenho, como aplicações em tempo real (*Chat*), microserviços e arquitetura *serveless*, na criação de projetos modularizados e muito popular na construção de API's RESTful.

Utiliza de *event loop*[6] (eventos que servem para gerenciar múltiplas conexões simultâneas), sem a necessidade de múltiplos *threads*, sobressaindo-se em aplicações que precisam lidar com um grande volume de requisições. O seu funcionamento abrange o modelo de *thread* única executando as operações. Estas quando são assícronas, são enviadas para uma fila de eventos, e quando há a conclusão desta operação, a função de *callback* associada é colocada na fila de execução, oferecendo o gerencimento múltiplo de I/O (Entrada e Saída), sem a necessidade de bloquear a execução do código.

2.6 Server Side Rendering

O conceito de SSR (*Server Side Rendering*), surge da ideia de aperfeiçoar o desempenho e melhorar o SEO (*Search Engine Optimization*), diferente de projetos SPA (*Single Page Application*) onde o servidor retorna os próprios objetos com instruções carregados em um único documento. O SSR é o processo da renderização das páginas no servidor, que são geradas no lado do servidor e enviado ao usuário como uma página completa renderizada, resultando em carregamentos mais rápidos e de uma melhor indexação para motores de busca. As figuras **??** e **??**, retratam respectivamente a diferença entra a arquitetura SPA e SSR.[7]

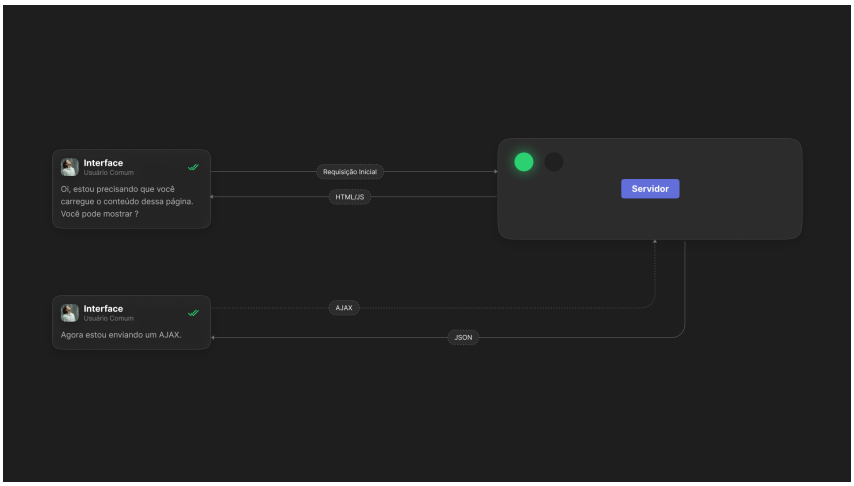


Figura 2.1: Fluxo de uma SPA.

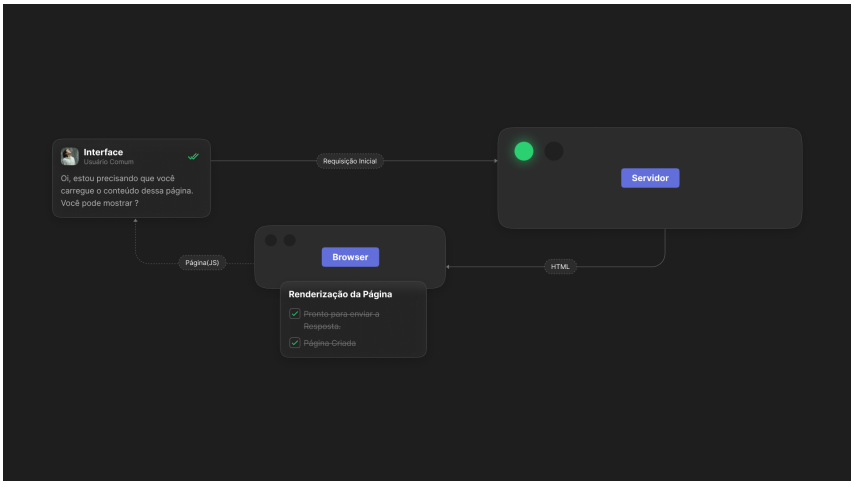


Figura 2.2: Fluxo de um SSR.

2.7 Monorepo

De uma foram sucinta, o Monorepo é uma arquitetura que utiliza apenas um repositório contendo subdiretórios (ex: subdiretório para Aplicação WEB, um subdiretório para o Servidor e um subdiretório para Testes). Os projetos compartilham diretamente das mesmas dependências.[26]

Esta arquitetura é geralmente utilizada, para maior colaboração entre o time que é uma das principais características, e também atomic changes(alterações atômicas) com o intuito de realizar operações para várias mudanças em diversos módulos do projeto.

2.8 SOLID

SOLID é um acrônimo que define cinco princípios fundamentais para design de software orientado a objetos, induzindo a criação de projetos coesos, flexíveis e com uma boa manutenção de código. Tais princípios são:

1. **Princípio da Responsabilidade Única**: Função/Classe deve ter apenas uma responsabilidade.

2. **Princípio Aberto/Fechado**: Classes, Funções e outros devem ser abertos para *"extends"*, mas fechados para modificação, podendo estender o comportamento do sistema, sem alterar o código existente.

3. **Princípio da Substituição de Liskov**: Os objetos de uma classe derivada podem ser substituídas por objetos da classe Pai, sem alterar a execução do programa.

4. **Princípio da Segregação de Interfaces**: É preferível o uso de inúmeras interfaces para propósitos específicos, do que uma interface única e global.

5. **Princípio da Inversão de Dependência**: Uma classe derivada deve ser substituível por sua classe base.

O método de arquitetura aplicado no princípio SOLID torna aplicações mais robustas, escláveis e flexíveis, permitindo manutenções e mudanças, facilitando a implementação de novas features como evolução da plataforma.[8]

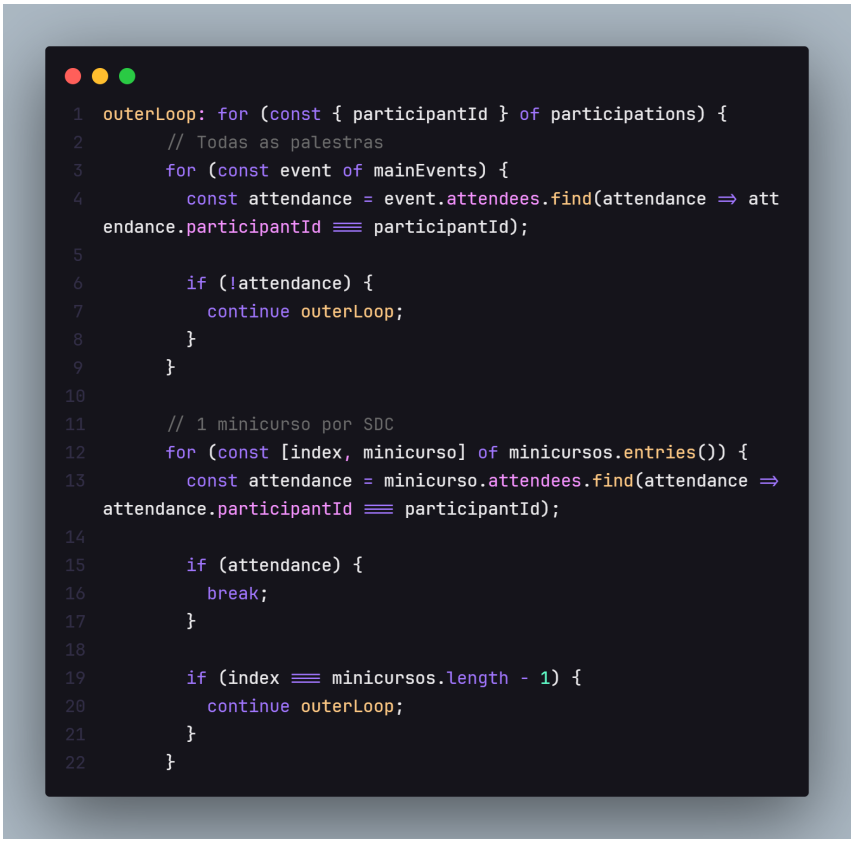
Capítulo 3

Metodologia

3.1 Visão Geral

A Plataforma SDC representa uma solução viável para promover eventos dentro do Campus e todo o ambiente acadêmico. O termo **SDC** é um acrônimo de *"Semana da Computação"*, que foi principalmente desenvolvida para atender os eventos promovidos pelo PET Computação. O próposito da plataforma era de fornecer um ambiente automatizado e de fácil acesso, para discentes, partindo de um objeto de estudo que é o Even3, a idealização foi centrada em aumentar:

- **Frequência de Alunos**: Exige-se pelo menos o cumprimento de 100% de presença em eventos do turno da manhã e 1 minicurso durante todo o evento. Dessa forma, o aluno torna-se apto a receber o certificado. A figura 3.1 retrata os parâmetro de aceitação para o cumprimento das regras.



```
outerLoop: for (const { participantId } of participations) {
      // Todas as palestras
      for (const event of mainEvents) {
        const attendance = event.attendees.find(attendance => att
endance.participantId === participantId);

        if (!attendance) {
          continue outerLoop;
        }
      }

      // 1 minicurso por SDC
      for (const [index, minicurso] of minicursos.entries()) {
        const attendance = minicurso.attendees.find(attendance =>
attendance.participantId === participantId);

        if (attendance) {
          break;
        }

        if (index === minicursos.length - 1) {
          continue outerLoop;
        }
      }
```

Figura 3.1: **Código** → Regras de negócio para Emissão

- **Verificação de Presença**: Na plataforma antiga, as presenças não eram captadas, apenas comparecia aqueles que achavam interessante, para este caso, criamos uma regra, que ao fim de cada programação, os discentes fizessem sua frequência na plataforma. Essa checagem visualmente era simples, mas, exigia as permissões de localização, captando a latitude e longitude do usuário. Dessa forma tínhamos registros de que o discente estava presente ou não no ambiente da programação em questão. Para serem validados na localização, foi criada um raio de 0,15km em torno do Centro de Informática, para a efetivação de presença. A figura 3.2 um pouco da área compreendida, e a figura 3.3 a implementação da verificação.

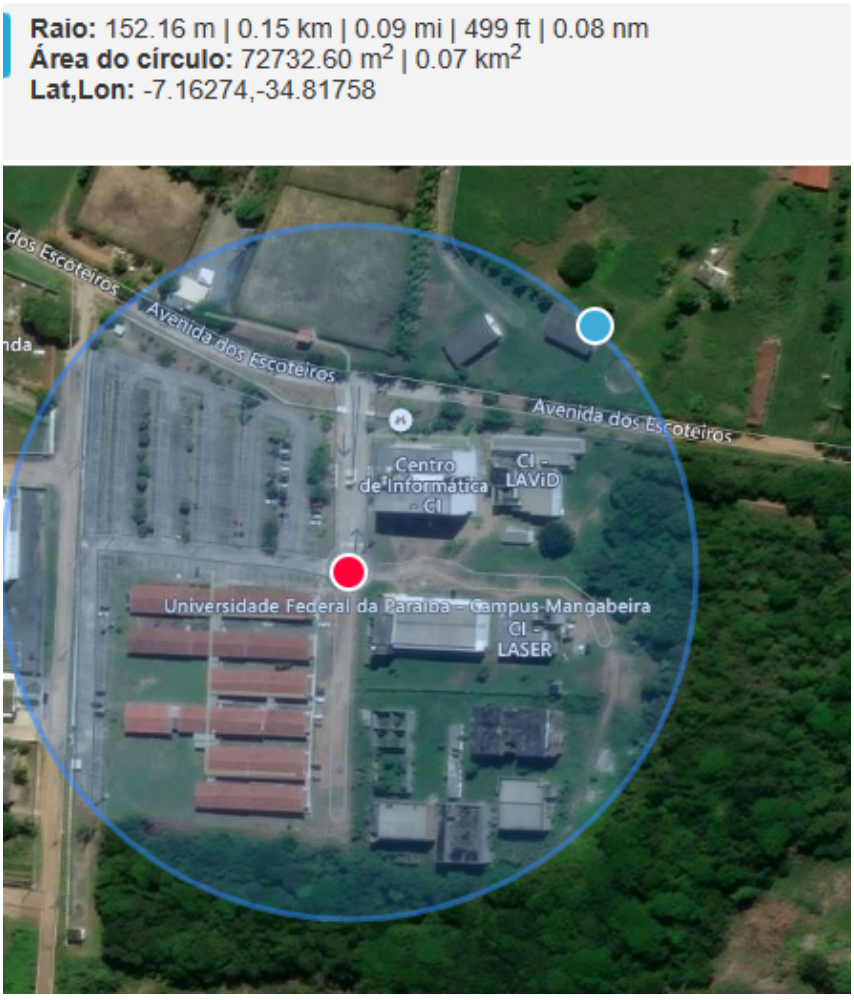


Figura 3.2: Raio de Aceitação para Verificação de Geolocalização

```
if (
      distanceBetweenCoordinates(
        [userLocation.coords.latitude, userLocation.coords.long
itude],
        [-7.16252, -34.8173],
      ) > 2000
    ) {
      toast.dismiss(i);
      toast.error("Você está fora da área do Centro de Informát
ica.", {
        position: "top-center",
        autoClose: 5000,
        hideProgressBar: false,
        closeOnClick: true,
        pauseOnHover: true,
        draggable: true,
        progress: undefined,
        theme: "dark",
        style: {
          position: "sticky",
          textAlign: "center",
          color: "#fgfgfg",
        },
      });
      return;
    }

```

Figura 3.3: **Código** → Condição de Verificação de Frequência

- **Automação de Certificados**: Outro recurso que impulsionou o projeto foi a emissão de certificados quase imediatamente após o evento. Como o Even3 solicitava os nomes e matrícula de cada discente participante, a equipe postergava a tarefa, pois exigia um trabalho manual de listagem de discentes, resultando em um atraso na entrega dos certificados. Como fonte de solução, essas tarefas foram automatizadas, através da regra de negócio anterior, sendo necessário o cumprimento para emissão. Ao fazer a consulta do ID do Participante (a emissão de certificados é feita automaticamente em cada edição da SDC), saberíamos de forma instantânea se o certificado foi emitido ou não para o participante em questão. A figura 3.4 retrata a emissão dos certificados.

```
await this.projectsRepository.createCertificates(certificateInf
o);

    const certificates = await this.projectsRepository.findCertif
icatesByEditionId(editionId);

    for (const { participantId } of certificates) {
      const participant = (await this.projectsRepository.findPart
icipantById(
        participantId,
      )) as ProjectParticipant;

      let project: Project;
      if (!existingEdition.name) {
        project = (await this.projectsRepository.findProjectById
(existingEdition.projectId)) as Project;
      }

      const certificateTitle =
        existingEdition.name || `${existingEdition.number}° ediçã
o do(a) ${project!.title}`;

      await this.mailProvider.sendMail({
        to: participant.email,
        subject: `Certificado do(a) ${certificateTitle}`,
        body: `Olá!<br/><br/>Estamos passando para informar que s
eu certificado do(a) ${certificateTitle} já está disponível.<br/>
<br/>Você pode acessá-lo em: ${process.env.WEB_URL}/sdc/certifica
dos/${editionId}?participantId=${participantId}`,
      });
    }
```

Figura 3.4: **Código** → Emissão de Certificados automáica para cada Edição.

Como alternativa para dispositivos que não permitiam localização em tempo real, outra verificação criada foi através do reconhecimento em rede. Caso o dispositivo estivesse dentro da rede do Campus, poderiam ser verificadas as informações de frequência. Na figura 3.5 podem ser observados os parâmetros para checagem em rede.

```
<FrequenciaForm
          type="cancel"
          id={event.id}
          sections={[
            { title: "Seu nome", placeholder: "João da Silva", id: "name" },
            { title: "E-mail cadastrado", placeholder: "seuemail@exemplo.com", id: "email" },
          ]}
          eventName={event.name}
          eventType={event.type}
          editionName={schedule.name || ""}
          date={{
            day: formatInTimeZone(new Date(event.startTime), "America/Fortaleza", "dd MMMM yyyy", {
              locale: ptBR,
            })
              .split(" ")
              .join(" de "),
            time: `${formatInTimeZone(new Date(event.startTime), "America/Fortaleza", "HH:mm")}h`,
          }}
          endTime={new Date(event.endTime)}
          isEventOnSite={event.onSite}
          confirmType="confirm"
          isFromUFPB={headers().get("x-user-ip")?.startsWith("150.165.200") ? true : false}
        />
```

Figura 3.5: **Código** → Verificação através do uso da rede.

- **Autenticidade de Documentos**: Dentre todas, existe um recurso adicional focada nos docentes, autenticando se o certificado enviado pelo discente era válido ou não. Com base nesse parâmetro, cada certificado contém um ID para verificação, sendo necessário para autenticar as informações para justificativa de falta pós evento.

A aplicação WEB foi desenvolvida utilizando o *framework* em NextJS da biblioteca de ReactJS, para construção das páginas, o ORM *Prisma*, para mapear o banco de dados, o *PostgreSQL*, para guardar os dados e o NestJS, para o *backend*. Essa combinação de tecnologias escolhidas proporcionou compatibilidade com todas as regras de negócio e navegadores flexíveis em todos os dispositivos.

Como a etapa de elicitação de requisitos foi realizada antes da prototipagem, toda a interface foi estruturada através de reuniões semanais, validando todas as funções da plataforma. Seguindo a estratégia *desktoo first*, as telas maiores foram estruturadas visando os usuários que utilizam *desktop* ou *notebook*, para depois ser acrescentada a responsividade, para que aplicação se adaptasse de forma automática para dispositivos móveis.

3.2 Tecnologias

No desenvolvimento da aplicação, foram utilizadas diversas tecnologias que, em conjunto, possibilitaram a criação de um sistema robusto, escalável e de alta performance. A escolha dessas tecnologias foi feita com base em critérios como compatibilidade, eficiência, facilidade de manutenção e adequação às necessidades específicas do projeto. Abaixo, são detalhadas as tecnologias adotadas e os motivos de sua escolha.

3.2.1 TypeScript

O *TypeScript* é um *superconjunto* do *JavaScript*, desenvolvida pela Microsoft, com a finalidade de adicionar recursos adicionais à linguagem. Adicionando tipagens as variáveis, visa diminuir a incidência de erros no desenvolvimento, visto que o JS é uma linguagem muito dinâmica, onde ao longo do tempo as variáveis podem assumir diferentes formatos. Esse recurso aumenta drastricamente a produtividade do time de desenvolvedores, pois atualmente as IDEs (*Integrated Development Environment*) ajudam a identificar o formato das variáveis, alertando erros de tipagem antes da execução do código.[18]

3.2.2 NextJS

O *NextJS* é um *framework* que permite inicializar uma aplicação *React* com suporte integrado a TypeScript, fornecendo um ambiente de desenvolvimento completo, configurado, reduzindo trabalhos de inicialização do projeto. Um dos pontos mais importantes para adoção do *framework*, foi o critério de roteamento em arquivos, e principalmente pela renderização no lado do servidor (SSR)[11].

3.2.3 NestJS

Framework progressivo baseado em NodeJS, com uma arquitetura modular e baseada em *decorators*, há simplificação na criação de APIs, sistemas em tempo real, microsserviços e outras soluções *backend*, sendo escalável e de fácil manutenção.[23]

3.2.4 Prisma

Para facilitar o mapeamento do BD (Banco de Dados), o ORM (*Object-Relational Mapping*) simplifica a interação com o banco, proporcionando otimização no desenvolvimento[29]. A modelagem e o gerenciamento de banco de dados são realizadas através do *Prisma*, utlizando uma sintaxe declarativa, gerando de forma automática as tipagens como prevenção na produtividade e maior segurança. Ao mesmo tempo que otimiza as consultas, mantém a flexibilidade para diferentes tipos de BD.

3.2.5 PostgreSQL

Sistema de Gerenciamento de Banco de Dados (SGBD), PostgreSQL é robusto e conhecido por sua conformidade com o padrão SQL. É composto por suporte nativo a tipos de dados complexos, transações e índices avançados, utilizando o controle de concorrência (MVCC). Foi essencialmente escolhido em termos de eficiência, reduzindo o esforço e tempo necessário para o desenvolvimento da API por completo.[24]

3.2.6 Redis

Redis é um estrutura de dados em memória, utilizando principalmente o sistema de cache para otimização e performance de aplicações, sobretudo em casos onde há um grande volume de requisições ao banco de dados principal. Armazenando temporariamente dados que são frequentemente reutilizados, foi o motivo central para escolha, reduzindo a carga e a quantidade de requisições ao banco de dados persistente.[12] Resumidamente, se o dados estiverem no Redis (*cache hit*), os dados são retornados imediatamente, economizando tempo e recurso. De outro modo (*cache miss*), a aplicação faz a consulta ao banco de dados, armazenando o resultado no Redis e retorna os dados para a aplicação.

3.2.7 Styled Components

A biblioteca *Styled Components* estiliza através de componentes React, que utiliza o conceito CSS-in-JS, permitindo que seja com JS ou TS, criando componentes encapsulados e dinâmicos. Isso facilita a modularidade do projeto e manutenção do código, evitando uma poluição global do CSS[27]. As estilizações são definidas por *template literals*, e aplicadas como extensão do componente React, fortemente utilizado auxiliando de *props*, para criação de objetos dinâmicos, que se adaptam confome os valores são instanciados.

3.3 Usuários

Essa aplicação foi desenvolvida principalmente para discentes e docentes da Unidade Mangabeira da UFPB, para fins acadêmicos e de fácil utlização na criação de eventos instituicionais.

3.4 Colaboradores

A plataforma SDC foi interiamente desenvolvida por 3 desenvolvedores do PET Computação, financiada pelos próprios recursos do grupo idealizador, disposto a utilizar a aplicação em todos os eventos que promover.

3.5 Hospedagem

A hospedagem refere-se ao processo de disponibilizar um site ou aplicação na internet, permitindo que usuários acessem os serviços oferecidos por meio de servidores. Nesta seção, serão abordados os detalhes da hospedagem do frontend e backend da plataforma, incluindo as escolhas feitas.

3.5.1 Frontend

O *frontend* foi hospedado na plataforma Vercel, por meio do plano gratuito, mas direcionado para um DNS de registro insitucional PET. A Vercel é uma solução popular para hospedagem de aplicações JavaScript, especialmente aquelas construídas com frameworks como React e Next.js, pois oferece otimizações automáticas e facilidade de integração com repositórios de código.[28]

3.5.2 Backend

Todo o *backend* foi hospedado no Railway, sendo necessária a alteração do Plano Gratuito para o Plano Hobby, necessitando de mais processamento para requisições na plataforma. O Railway é uma plataforma que simplifica o processo de implantação de aplicações backend, oferecendo um ambiente eficiente e escalável para desenvolvimento e produção.[19]

3.6 Requisitos Funcionais

Os requisitos funcionais são as especificações que definem as funcionalidades e comportamentos que um sistema deve apresentar. Eles descrevem o que o sistema deve fazer e são fundamentais para garantir que as necessidades dos usuários sejam atendidas. De acordo com Sommerville (2011), requisitos funcionais são "as funcionalidades que o sistema deve fornecer"e podem incluir ações, comportamentos e informações que o sistema deve processar.[13]

3.6.1 [RF01] Cadastrar no Evento

A aplicação deve permitir o cadastro do usuário no evento, utilizando nome, e-mail, senha, telefone, matrícula, curso e data de nascimento, para criação da conta e possibilitar o acesso aos recursos da aplicação.

3.6.2 [RF02] Registrar Programação

A aplicação deve permitir que o usuário se registre em uma programação (Palestra, Minicurso ou Roda de Conversas), utilizando e-mail cadastrado e nome, efetivando a vaga do usuário, dando acesso e permissões na hora do evento.

3.6.3 [RF03] Registrar Frequência

A aplicação deve permitir que o usuário registre sua participação em uma programação (Palestra, Minicurso ou Roda de Conversas) através da frequência, utilizando e-mail cadastrado e matrícula, efetivando o cumprimento das regras, com base em sua participação.

3.6.4 [RF04] Visualizar Programação em andamento e encerradas

A aplicação deve permitir que o usuário visualize toda a programação do evento registrada, estando ativa ou encerrada.

3.6.5 [RF05] Emitir Certificado

A aplicação deve permitir que o usuário solicite e visualize a emissão de seu certificado, observando os critérios de emissão.

3.6.6 [RF06] Autenticar Certificado

A aplicação deve permitir que o usuário verifique a autenticidade dos documentos emitidos, utilizando a matrícula e o UUID (*Universally unique identifier*) do documento gerado anteriormente.

3.6.7 [RF07] Verificar Geolocalização

A aplicação deve permitir que o usuário valide sua geolocalização, assegurando que a localização do dispositivo esteja dentro dos parâmetros predefinidos para a participação no evento.

3.6.8 [RF08] Verificar Rede de Acesso

A aplicação deve permitir que o usuário valide sua conexão de rede, assegurando que a rede conectada do dispositivo esteja no mesmo domínio do campus predefinido como parâmetro de aceitação no evento.

3.7 Requisitos não Funcionais

Os requisitos não funcionais descrevem critérios que podem ser usados para julgar a operação de um sistema, em vez de suas funcionalidades específicas. Eles abrangem aspectos como desempenho, segurança, usabilidade e confiabilidade. Segundo Sommerville (2011), os requisitos não funcionais "especificam como um sistema deve se comportar e limitar as opções de projeto".[13]

3.7.1 Usabilidade

A usabilidade refere-se à facilidade com que os usuários podem interagir com uma aplicação, garantindo que a experiência de uso seja eficiente e agradável. Um sistema com boa usabilidade permite que os usuários atinjam seus objetivos de forma rápida e sem frustrações. De acordo com ISO 9241-11 (1998), usabilidade é "a eficácia, eficiência e satisfação com que usuários específicos alcançam objetivos específicos em um contexto de uso específico".[22]

3.7.1.1 [RNF01] Facilidade de uso

A aplicação deve ser intuitiva, permitindo que os usuários a utilizem sem a necessidade de treinamento prévio. Para alcançar essa facilidade de uso, será essencial aplicar as Heurísticas de Nielsen, que oferecem princípios de usabilidade para orientar o design da interface. Isso inclui garantir a visibilidade do status do sistema, assegurar uma correspondência clara entre o sistema e o mundo real, e proporcionar controle ao usuário sobre suas ações.

3.7.1.2 [RNF02] Responsividade

A aplicação deverá funcionar em tablets com Android 8 ou superior e iOS 12 ou superior, smartphones e notebooks, suportando uma variedade de tamanhos de tela, desde 4 polegadas (smartphones) até 15 polegadas (notebooks)

3.7.1.3 [RNF03] Prevenção de erros

A aplicação deverá possuir mecanismos de confirmação e recuperação para prevenção de ações acidentais do usuário. Oferecendo mensagens de erro claras e específicas em caso de erro, indicando o que ocorreu e como o usuário pode corrigir a situação.

3.7.2 Portabilidade

A portabilidade refere-se à capacidade de uma aplicação ser executada em diferentes plataformas e ambientes sem a necessidade de modificações significativas. A aplicação

portátil pode ser facilmente adaptada para funcionar em diferentes sistemas operacionais ou navegadores, aumentando sua acessibilidade e utilidade para os usuários.[14]

3.7.2.1 [RNF04] Compatibilidade

A aplicação deverá funcionar nos navegadores que suportam ES6, *Chrome* versão 61+, *Edge* versão 16+, *Firefox* versão 60+, *Opera* versão 48+ e *Safari* versão 10.1+.

3.7.3 Confiabilidade

A confiabilidade é a capacidade de uma aplicação de realizar suas funções de forma consistente e sem falhas, garantindo que o sistema se comporte como esperado em diferentes condições de operação.[15]

3.7.3.1 [RNF05] Disponibilidade

A aplicação deverá ter alta disponibilidade, garantindo que esteja funcional e acessível durante o período do evento em todas as suas edições. Caso aconteça evetualidades com a infraestrutura da plataforma, o objetivo é minimizar interrupções e maximizar o tempo de atividade, assegurando que os usuários possam acessar as funcionalidades da aplicação sempre que necessário.

3.7.4 Aspecto Organizacional

O aspecto organizacional refere-se à estrutura e práticas de desenvolvimento que envolvem a equipe, as ferramentas e as metodologias utilizadas na criação da aplicação.

3.7.4.1 [RNF06] Ambientes e ferramentas da Implementação

A aplicação deverá ser desenvolvida utilizando o *framework NextJS* versão 14.x.x, *NestJS* versão 9.x.x, *Prisma* versão 4.12.x.

3.7.4.2 [RNF07] Controle de Versão.

A aplicação deverá ser desenvolvida a partir da plataforma Github do repositório da PlataformaSDC.

3.7.5 Interoperabilidade

A interoperabilidade é a capacidade de um sistema de interagir e funcionar com outros sistemas e componentes, permitindo a troca e o uso de informações de forma eficiente.[25]

3.7.5.1 [RNF08] Acesso à API

A aplicação deverá se comunicar com a API, utilizando *Railway V2* para gerenciamento dos dados do sistema.

3.7.5.2 [RNF09] Conexão à internet

A aplicação requer uma conexão estável à Internet com uma largura de banda mínima de 1 Mbps para funcionar adequadamente. Essa exigência é fundamental para garantir o acesso a funcionalidades essenciais, como a atualização de dados em tempo real e a interação com serviços online.

3.7.6 Privacidade

A privacidade refere-se ao direito dos usuários de controlar suas informações pessoais e de decidir como essas informações são coletadas, usadas e compartilhadas.[16]

3.7.6.1 [RNF10] Dados do usuário

O aplicativo não revelará aos usuários informações de natureza privada, como dados pessoais de outros usuários, obedecendo todas as regras da Lei Geral de Proteção de Dados (LGPD).

3.7.7 Segurança

A segurança envolve medidas e práticas que visam proteger um sistema contra acessos não autorizados e garantir a integridade, confidencialidade e disponibilidade dos dados (CID).[21]

3.7.7.1 [RNF11] Registro de usuário

O aplicativo registrará os usuários por meio de e-mail e senha.

3.7.7.2 [RNF12] Controle de acesso

O aplicativo assegurará que apenas usuários autorizados, como participantes e administradores, tenham acesso aos dados. Os participantes terão acesso a informações relacionadas ao evento, como programação, materiais e seu próprio status de inscrição e presença. Os administradores, por outro lado, terão acesso a dados gerenciais, incluindo informações dos participantes, relatórios de presença e estatísticas do evento. Todos os acessos serão protegidos por mecanismos de autenticação e autorização para garantir a segurança das informações.

3.8 Arquitetura

A arquitetura da Plataforma da SDC segue o padrão em Camadas (**Layers**) [20], a figura 3.6 retrata de forma ilustrativa. No contexto dessa aplicação, o *NextJS* age nas camadas mais altas (Interface de Usuário e Gerenciamento de Autenticação e Autorização), e o *NestJS* representa os dois níveis mais baixos, que estão responsáveis pelas Regras de Negócio e todo o apoio da Aplicação.

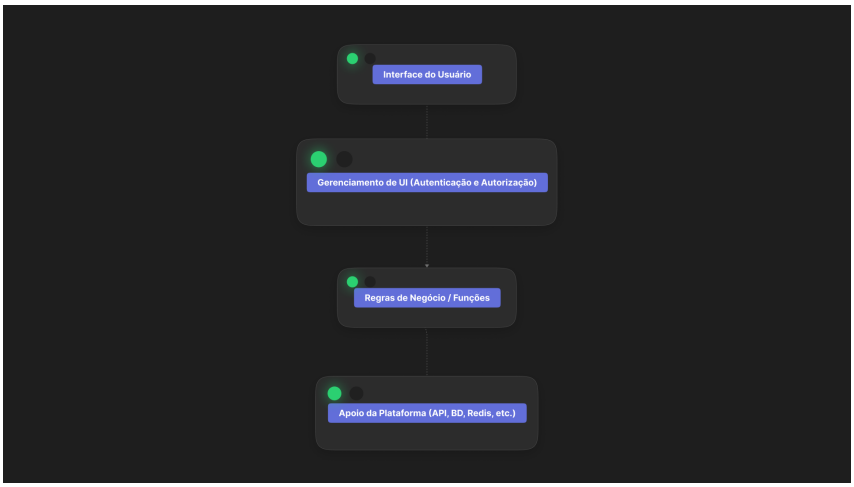


Figura 3.6: Conceito de Arquitetura em Camadas

Na representação, a camada de Interface de Usuário, que está no mais alto nível, é a de renderização de elementos, que é ligada à camada abaixo, que por sua vez gerencia o processo de autenticação e autorização do usuário e sistema. Abaixo dessas duas primeiras camadas, são encontradas mais duas, a que acolhe as regras de negócio e toda sua lógica principal e os recursos da plataforma, e a camada mais profunda se responsabiliza pelo apoio, no que se diz ao aramazenamento de dados, API, dentro outros.

3.9 Interface da Aplicação

Nesta seção, apresentamos uma visão geral da interface da Plataforma Semana da Computação, incluindo o fluxo de aplicação e as telas principais. A subseção "Fluxo de Aplicação" descreve o caminho que o usuário percorre dentro da aplicação, desde a tela inicial até a autenticação de certificados. As subseções seguintes detalham cada tela relevante, explicando suas funcionalidades e como o usuário interage com elas.

3.9.1 Fluxo de Aplicação

A figura 3.7 apresenta uma representação simplificada do fluxo de aplicação entre telas da Plataforma SDC. Ao abrir a aplicação WEB, a primeira tela a ser renderizada é a Tela de Programação Geral do Evento, com o intuito de informar qual será a programação ofertada

por um determinado evento, ficando a critério do usuário decidir participar ou não. Após a decisão de participar ou não, o usuário, ao apertar no botão “Garantir minha Vaga” é destinado a sua inscrição na plataforma, onde é redirecionado para o preenchimento de dados necessários para registro logo após sua conclusão, é redirecionado novamente para programção, onde pode escolher dentre os minicursos ofertados, apenas um minicurso para participar. Ao clicar em ”Fazer Inscrição”, relacionado alguma programação, o usuário é destinado para tela de registro utilizando sua identificação primária, havendo a necessidade de inserir novamente os dados comprobatórios de sua inscrição geral.

Figura 3.7: Fluxo de Aplicação

A tela de Frequência solicita ao usuário matrícula e e-mail, para efetivar sua participação na atividade em questão, desde que sejam permitidos os critérios de localização, para prosseguir com a análise de frequência. Por último, a tela de Autenticação de Certificados exige do usuário a matrícula e o UUID de seu certificado, para verificar autenticidade do seu documento, geralmente utilizado pelos professores, com o intuito de verificar se realmente o aluno participou do evento.

3.9.2 Telas

3.9.2.1 Tela Inicial - Programação

A Figura 3.8 apresenta a tela inicial da Plataforma SDC. Apresentada para todo o redirecionamento de página, exibindo sempre a programação e atividades em progresso ou já encerradas.

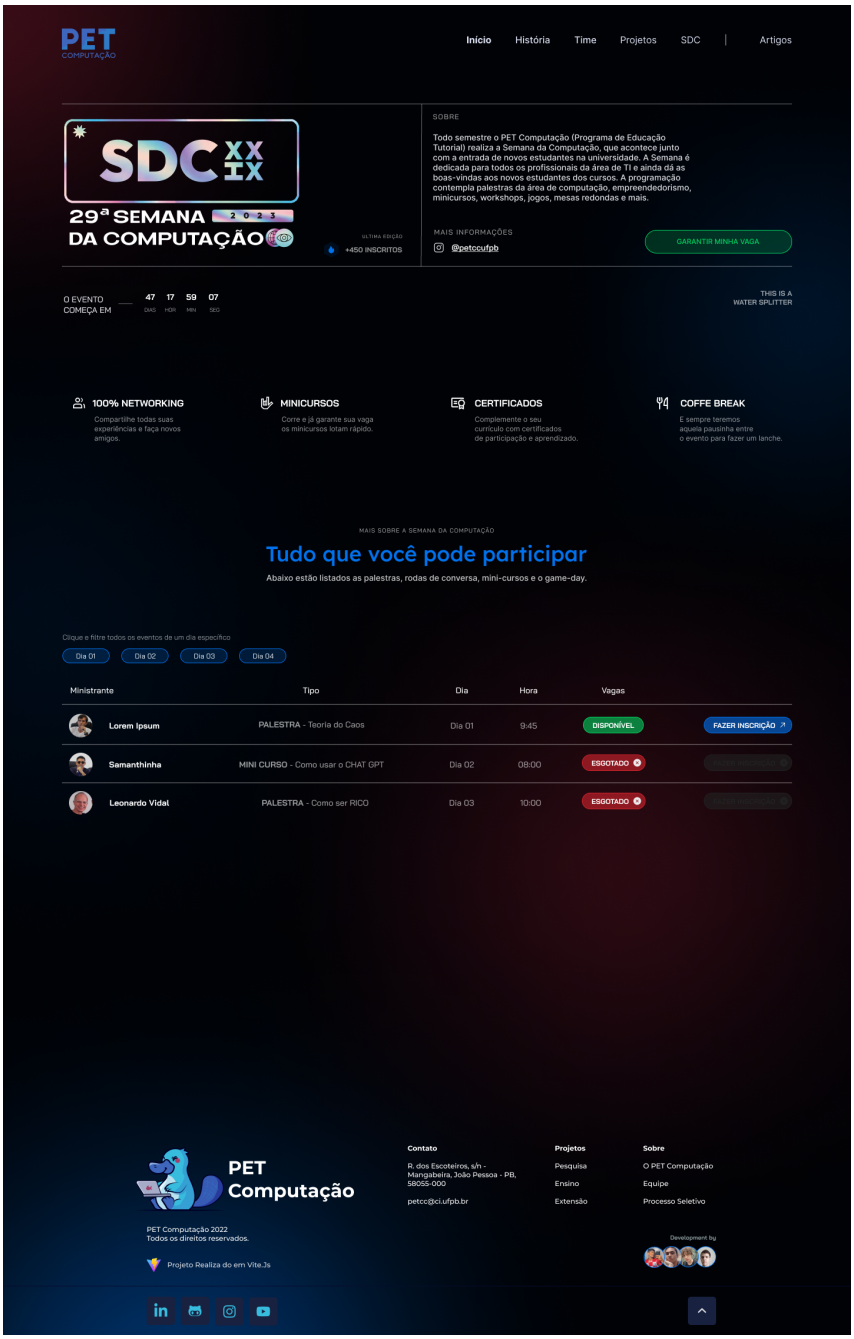


Figura 3.8: Tela inicial - Programação

3.9.2.2 Tela de Inscrição Geral

A Figura 3.9 apresenta a tela Inscrição Geral do Evento na Plataforma SDC. Apresentada para o usuário caso aperte o botão "Garantir minha vaga"na tela inicial.

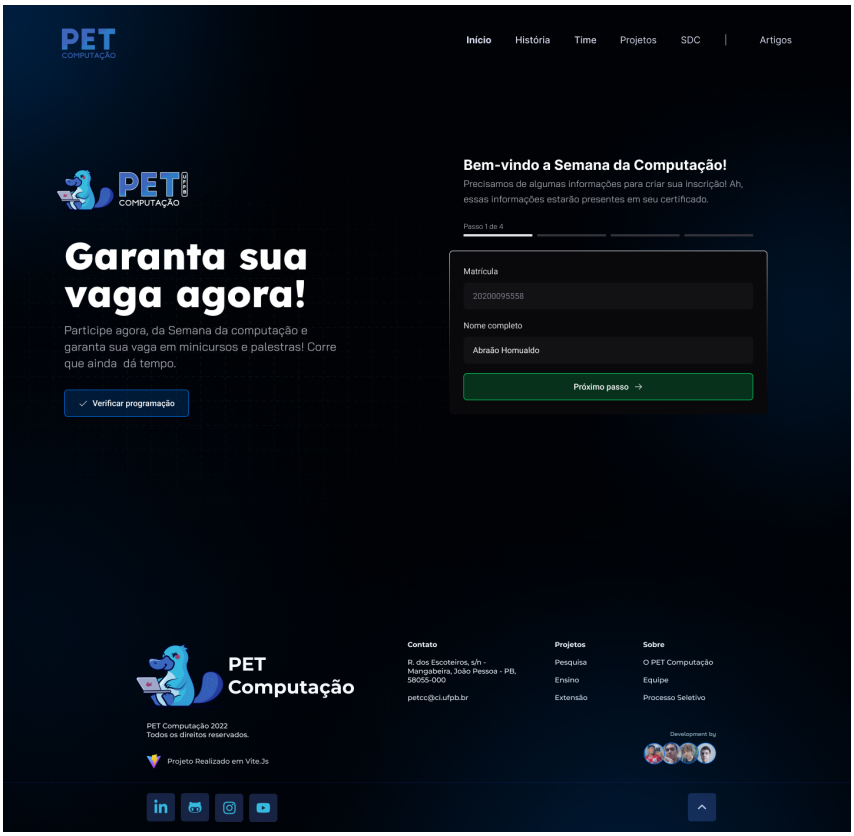


Figura 3.9: Tela de Inscrição Geral

3.9.2.3 Tela de Inscrição Minicurso

A Figura 3.10 apresenta a tela Inscrição para um Minicurso ofertado no Evento da Plataforma SDC. Apresentada para o usuário caso aperte o botão “Fazer Inscrição” na tela inicial, nela é possível selecionar alguma atividade dentro da programação visualizada.

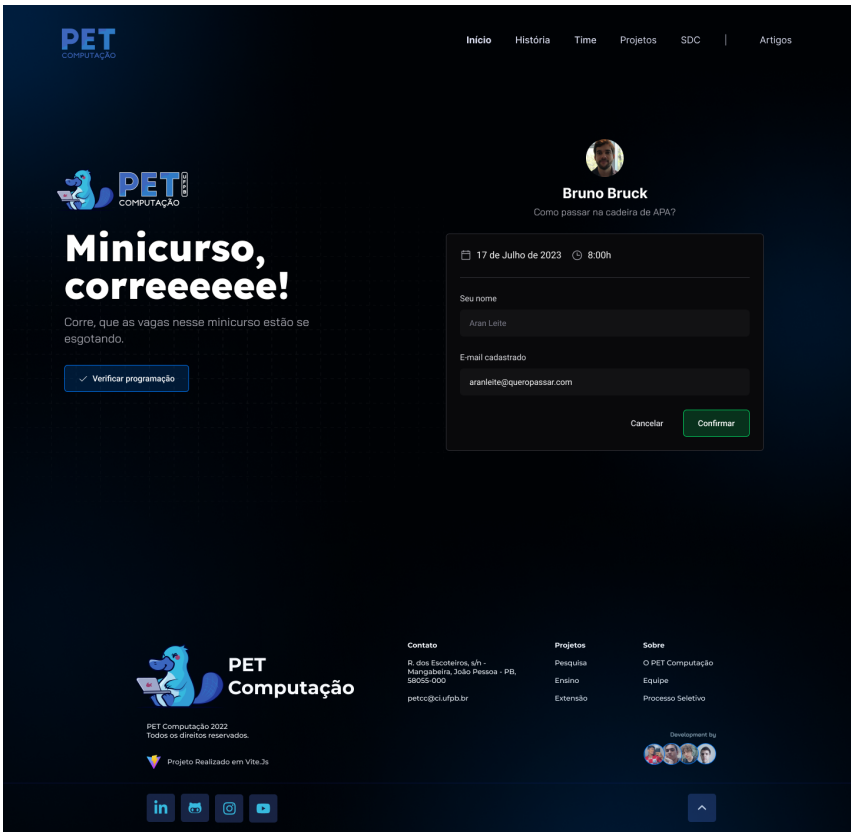


Figura 3.10: Tela de Inscrição para Minicurso

3.9.2.4 Tela de Frequência Minicurso

A Figura 3.11 apresenta a tela de frequência para um Minicurso cadastrado para o usuário no Evento da Plataforma SDC. É apresentada para o usuário, caso ele esteja dentro do ambiente do evento e receba o link da rota da frequência, inserindo os dados solicitados baseados na inscrição geral.

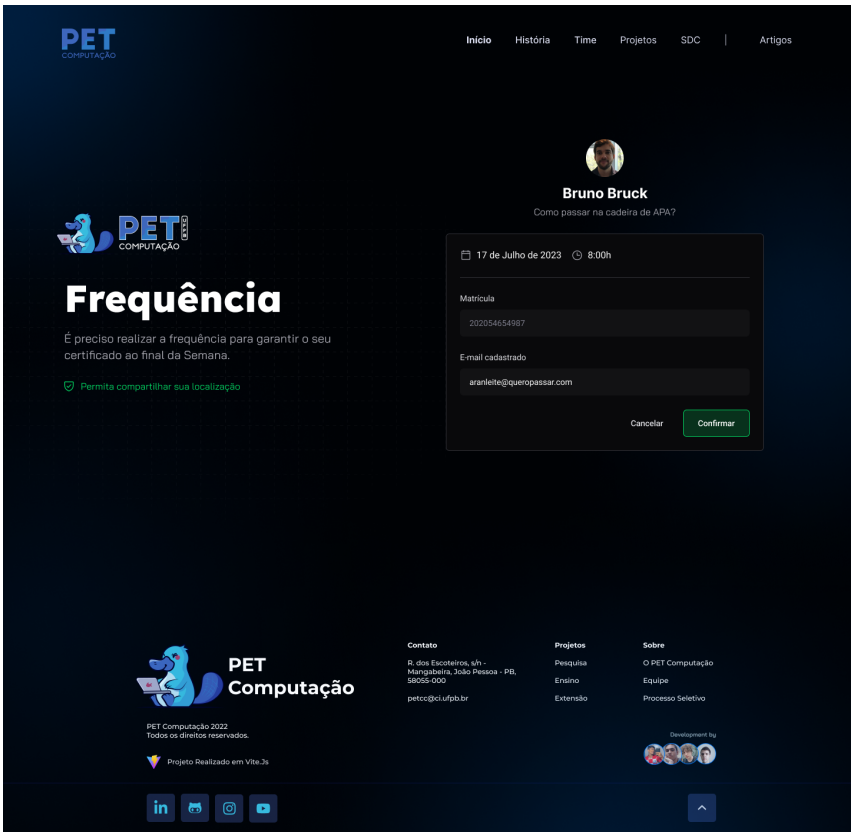


Figura 3.11: Tela de Frequêncioa para Minicurso

3.9.2.5 Tela de Autenticador

A Figura 3.12 apresenta a tela de autenticador para checar a emissão de certificados no Evento da Plataforma SDC. A tela é apresentada para o usuário caso ele clique no botão do *header* "Autenticador", onde será redirecionado e solicitado à inserir os dados para verificar as informações do seu certificado. Após clicar no botão "Verificar", o usuário poderá ser redirecionado para a seguinte, caso seja válido, e após 5 segundos o usuário será reconduzido para tela de programação.

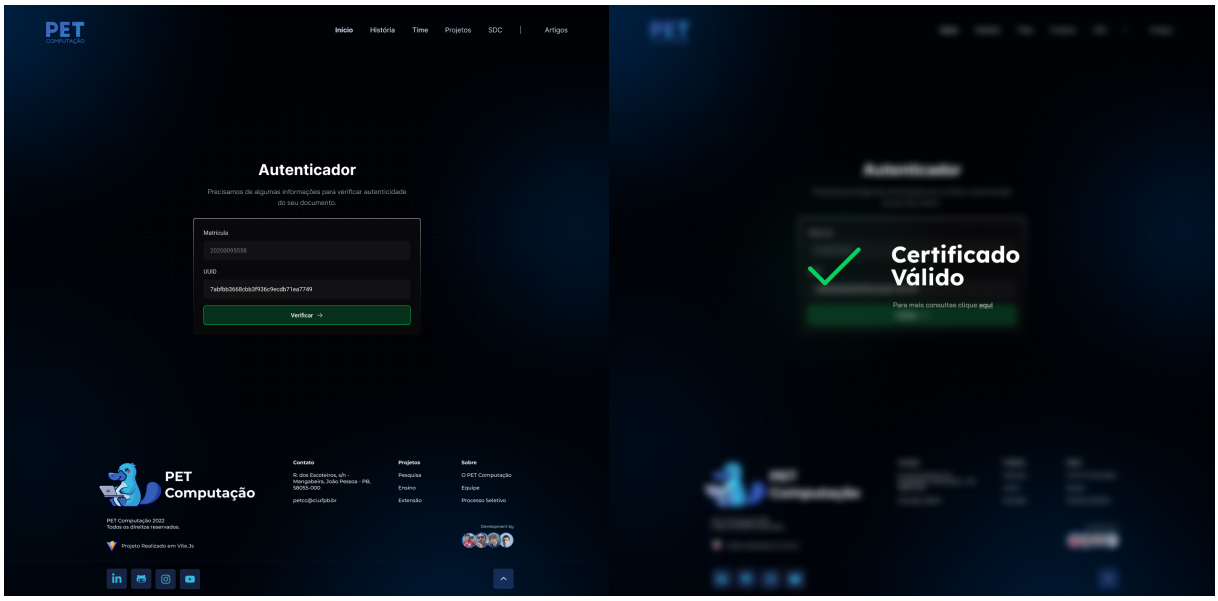


Figura 3.12: Telas de Autenticidade na Emissão de Certificados

Capítulo 4

Análise de Resultados

Esse capítulo trata dos resultados e análises das regras de negócio e testes unitários, com intuito de avaliar a Plataforma SDC, apresentando o método utilizado e analisando os resultados. Todos as chegagens e testes unitários foram feitos antes do primeiro evento, após a utilização da plataforma, foram incluídos todos os casos e problemas da primeira edição.

4.1 Plano de Testes

A seguir serão apresentada a checagem de regras de negócio da Plataforma SDC, bem como seu planejamento para todos os testes unitários. Através dessas validações, foram reduzidos pelo menos 87,3% de eventuais problemas de inserção de dados errados, dados repetitivos e tentantivas de intrusão antes do próprio *deploy* da aplicação, a fim de que, ao utilizar nos eventos, as condições sejam as melhorias possíveis para o uso.

Esse plano de testes objetivou a navegação por todas as principais funções, simulando desde a inserção de dados à, ações do usuário, por toda a plataforma, em um dia de evento. Para tais testes, foram criados 3 personas, apresentadas a seguir, para representar usuários que utilizaram a plataforma. Em seguida, os testes unitários determinados pelo fluxo da aplicação foram avaliados, partindo desde a criação de um evento, até a verificação de um certificado criado por aquele evento. Os testes foram elaborados e executados em ambiente de desenvolvimento pelo autor desse trabalho. As ferramentas usadas foram um *notebook* rodando WSL e o *framework* Jest, que foi integrado ao espaço NestJS.

4.2 Personas

Neste contexto, personas são representações fictícias de usuários que ajudam a entender melhor as necessidades, comportamentos e objetivos dos diferentes públicos-alvo de um produto ou serviço. Elas são construídas a partir de dados reais e representam uma

tipologia de usuário, permitindo que equipes de desenvolvimento e design criem soluções mais eficazes e centradas no usuário. Segundo Pruitt e Adlin (2006), "uma persona é um retrato de um usuário típico que ilustra suas necessidades, desejos e hábitos".[17]

4.2.1 Persona A

Israel Sousa é um ingressante no curso de Engenharia da Computação na UFPB. Com ensino médio completo, ele se mudou para João Pessoa e não conhece ninguém da cidade e nem do curso. Ele vê na Semana da Computação uma oportunidade para se conectar e fazer amizades com estudantes e profissionais da área. Israel efetiva sua inscrição na plataforma e planeja participar de todas as palestras e um minicurso ofertado no evento, expandindo conhecimento e *networking*. A Tabela 4.1 apresenta as informações do Israel.

Informações do Usuário	
Nome	Israel Sousa
Idade	18 anos
Curso	Engenharia da Computação
Ano de Ingresso	2024
Cidade de Origem	Sousa, PB
Objetivo	Fazer *networking* e se integrar ao ambiente acadêmico

Tabela 4.1: Persona A

4.2.2 Persona B

Lucas Pereira é um veterano do curso de Ciência da Computação na UFPB. Com 24 anos, ele já está no último período e participou 7 vezes da Semana da Computação. Porém, ele já se encontra muito cansado, e perdeu o interesse nos eventos ofertados e pariticpa de forma esporádica, seu objetivo principal é emitir o certificado de uma forma fácil e rápida para justificar suas faltas da semana corrente do Evento. A Tabela 4.2 apresenta as informações de Lucas.

Informações do Usuário	
Nome	Lucas Pereira
Idade	24 anos
Curso	Ciência da Computação
Ano de Ingresso	2019
Cidade de Origem	João Pessoa, PB
Objetivo	Obter o Certificado de Participação para justificar suas faltas

Tabela 4.2: Persona B

4.2.3 Persona C

Prof. Marcos Lima é docente do Centro de Informática na UFPB e leciona disciplinas de Análise de Projeto de Algoritmos. Com 45 anos, valoriza eventos que permitem o desenvolvimento dos alunos, sempre se preocupando com assiduidade e o real aproveitamento durante os eventos. Pós evento, o Professor valida os certificados entregues por seus alunos, garantindo que eles realmente estiveram presentes e participaram, obedecendo as regras estipuladas pela organização. O foco não é prejudicar os discentes, mas verificar se realmente vale a pena liberá-los das aulas para o evento. A Tabela 4.3 apresenta as informações do Prof. Marcos.

Informações do Usuário	
Nome	Prof. Marcos Lima
Idade	45 anos
Profissão	Docente do Centro de Informática
Experiência	20 anos
Cidade de Origem	Bragança Paulista, SP
Objetivo	Avaliar a autenticidade dos certificados para avaliar a participação dos alunos

Tabela 4.3: Persona C

4.3 Checagem de Regras de Negócio

A seguir são descritas, em formatos de tabela, as regras de negócio com maior relevância. A linha de descrição, trata das condições do usuário. A linha de entradas apresenta o que foi inserido pelo usuário para execução da checagem. A linha de resultado obtido exibe a resposta esperada para a respectiva checagem. A linha de observações apresenta as análises da checagem em questão.

Checagem RN01	**Frequência de Atividade fora do Raio de Aceitação**
Descrição	Esta checagem visa avaliar se o participante está na localização exigida, para efetivar sua frequência na atividade.
Entradas	O usuário preenche as informações solicitadas: Nome e e-mail. Após inserir os dados, são solicitados as permissões de localização em seu dispositivo.
Resultado	Uma notificação deve aparecer no canto superior da tela, exibindo que o usuário está fora do alcance de checagem dos seus dados.
Observações	Nenhuma

Tabela 4.4: Regra de Negócio 1

Checagem RN02	**Frequência de Atividade dentro do Raio de Aceitação**
Descrição	Esta checagem visa avaliar se o participante está na localização exigida, para efetivar sua frequência na atividade.
Entradas	O usuário preenche as informações solicitadas: Nome e e-mail. Após inserir os dados, são solicitadas as permissões de localização em seu dispositivo.
Resultado	Uma notificação deve aparecer no canto superior da tela exibindo que o usuário completou todos os requisitos para verificar sua frequência.
Observações	Nenhuma

Tabela 4.5: Regra de Negócio 2

Checagem RN03	**Verificação Dupla**
Descrição	Esta checagem tem o intuito de impedir que participantes dentro do CI validem informações para outros participantes fora da área do CI.
Entradas	O usuário preenche as informações solicitadas: Nome e e-mail, após inserir os dados.
Resultado	Uma notificação deve aparecer no canto superior da tela exibindo que o usuário já validou uma vez os dados, sendo impossível verificar mais de uma vez.
Observações	Nenhuma

Tabela 4.6: Regra de Negócio 3

Checagem RN04	**Verificação através da Rede Acadêmica**
Descrição	Esta checagem foca nos dispositivos dos participantes onde não é perimitido o uso de localização. Suas validações são através da rede do prórprio campus.
Entradas	O usuário preenche as informações solicitadas: Nome e e-mail, após inserir os dados.
Resultado	Uma notificação deve aparecer no canto superior da tela, exibindo que o usuário validou suas informações com sucesso, utilizando a rede do Centro de Informática.
Observações	Nenhuma

Tabela 4.7: Regra de Negócio 4

4.4 Testes Unitários com Jest

Nesta seção será discutido cada teste unitário realizado no ambiente de desenvolvimento, utilizando Jest, um *framework* desenvolvido pelo Meta, utilizado para testar aplicações *frontend* e *backend*. O seu ponto forte é o suporte a funcionalidades como *mocks*, *spies* e *assertions*, sendo integrado de forma versátil no projeto com *NestJS* (sendo integrada de forma nativa), facilitando o TDD (Desenvolvimento Orientado a Testes). Os testes serão brevemente apresentados, no formato Descrição/Código, considerando que existem, em cada teste individual, outras condicionais complementares.[1]

4.4.1 Criar Evento

O objetivo deste teste é fornecer a garantia que um evento não possa ser criado antes da data de início da edição associada. O teste recebe como parâmetro a data da edição, e aguarda a execução do **CreateEvent**, para que ele lance uma exceção do tipo **HttpException**. A Figura 4.1 mostra um dos trechos do código mencionado na descrição.

```
it("should not be able to create an event before the edition's start date", async () => {
  const startTime = new Date(edition.date);
  startTime.setDate(edition.date.getDate() - 1);

  await expect(
    service.execute({
      about: "",
      editionId: edition.id,
      endTime: new Date(),
      location: "Test Room",
      name: "Test Event",
      onSite: true,
      speakerId: speaker.id,
      startTime,
      type: "main",
    }),
  ).rejects.toBeInstanceOf(HttpException);
});
```

Figura 4.1: **Código** → TU Criar Evento

4.4.2 Criar Edição

O objetivo deste teste é verificar se o serviço **CreateEdition** impede a criação de uma edição, com o mesmo número para um projeto, que já possui uma edição com esse número. O teste cria uma edição inicial e depois tenta criar com o mesmo número, o esperado é que retorne **HttpException**, indicando duplicidade no número de edição. A Figura 4.2 mostra um dos trechos do código mencionado na descrição.

```
it("should be able to create an edition", async () => {
    const { id: projectId } = await fakeProjectsRepository.createProject({
      title: "Test",
    });

    await service.execute({
      projectId,
      date: new Date(),
      number: 1,
    });

    await expect(
      service.execute({
        projectId,
        date: new Date(),
        number: 1,
      }),
    ).rejects.toBeInstanceOf(HttpException);
  });
```

Figura 4.2: **Código** → TU Criar Edição

4.4.3 Criar Certificado da Edição

A verficação neste teste, vistoria se o serviço **CreateEditionCertificates** é capaz de criar certificados para uma edição, mapeando se o participante realmente efetivou sua frequência em eventos principais (Palestras e Rodas de Conversas) e minicurso. O método inicialmente registra a presença em ambos os eventos e na execução chama o ID da edição, sendo esperado o retorno de um único certificado, verificando se todas as condições do usuário são atendidas. A Figura 4.3 mostra um dos trechos do código mencionado na descrição.

```
it("should create certificates for an edition", async () => {
    await fakeProjectsRepository.createAttendance({
      eventId: event.id,
      participantId: participant.id,
    });
    await fakeProjectsRepository.createAttendance({
      eventId: minicurso.id,
      participantId: participant.id,
    });

    const certificates = await service.execute(edition.id);

    expect(certificates).toHaveLength(1);
  });
```

Figura 4.3: **Código** → TU Criar Certificado da Edição

4.4.4 Criar Certificados para o Evento

O teste em questão verifica se **CreateEventCertificates** é capaz de criar certificados para um evento, quando um pariticipante compareceu ao evento. Primeiro, valida-se a existência de um participante para o evento, e a execução é chamada junto com o ID do evento, esperando exatamente um único certificado ao final do teste. A Figura 4.4 mostra um dos trechos do código mencionado na descrição.

```
it("should be able to create certificates for an event", async () => {
    await fakeProjectsRepository.createAttendance({
      eventId: event.id,
      participantId: participant.id,
    });

    const certificates = await service.execute(event.id);

    expect(certificates).toHaveLength(1);
  });
```

Figura 4.4: **Código** → TU Criar Certificados para o Evento

4.4.5 Criar Participante

Há uma verificação, onde o **CreateParticipant** consegue criar um participante com sucesso, quando todos os dados são válidos e únicos. Na execução o teste espera que o objeto do *return* contenha o ID, indicando a criação correta do participante. A Figura 4.5 mostra um dos trechos do código mencionado na descrição.

```
it("should be able to create a participant", async () => {
    const participant = await service.execute({
      birthDate: new Date(),
      course: "Test Course",
      email: "test@gmail.com",
      matricula: "20200015280",
      name: "Test",
      phoneNumber: "+55 83 99999-9999",
      university: "Test University",
    });

    expect(participant).toHaveProperty("id");
  });
```

Figura 4.5: **Código** → TU Criar Participante

4.4.6 Avaliar Participação

Este teste valida que o serviço pode criar uma participação, ao receber um e-mail como entrada. O objetivo principal é verificar se na execução o email do participante é válido, para adicionar uma frequência para seu ID. Ainda no código completo deste teste, há outra alternativa através da matrícula, demonstrando flexibilidade do serviço, aceitando diferentes formas de identificação do participante. A Figura 4.6 mostra um dos trechos do código mencionado na descrição.

Figura 4.6: **Código** → TU Avaliar Participação

4.4.7 Criar Palestrante

Nesse teste o **CreateSpeaker** cria um novo palestrante quando todas as informações necessárias são válidas. É possível enviar uma foto através de uma URL válida, garantindo que na execução, um novo objeto de palestrante seja gerado com um ID. Outro teste incluído no mesmo código é a aceitação do envio da foto através do GoogleDrive. A Figura 4.7 mostra um dos trechos do código mencionado na descrição.

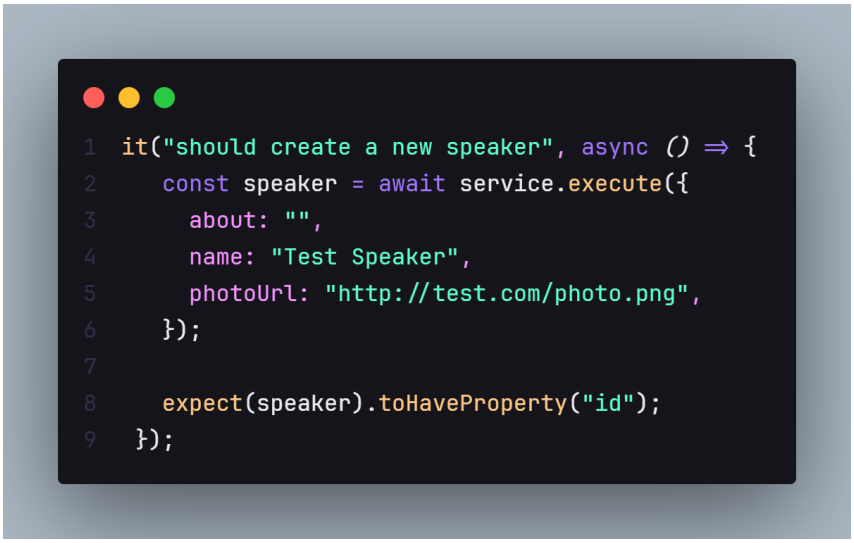


Figura 4.7: **Código** → TU Criar Palestrante

4.4.8 Localizar Ultima Edição

O teste verifica se **FindLatestEition** consegue localizar a edição mais recente de um projeto, dado o título do evento, garantindo que retorne a edição correta, assegurando que o ID da edição encontrada corresponda com o ID da última edição, um teste crucial validando a lógica da busca pelo projeto e sua edição. A Figura 4.8 mostra um dos trechos do código mencionado na descrição.

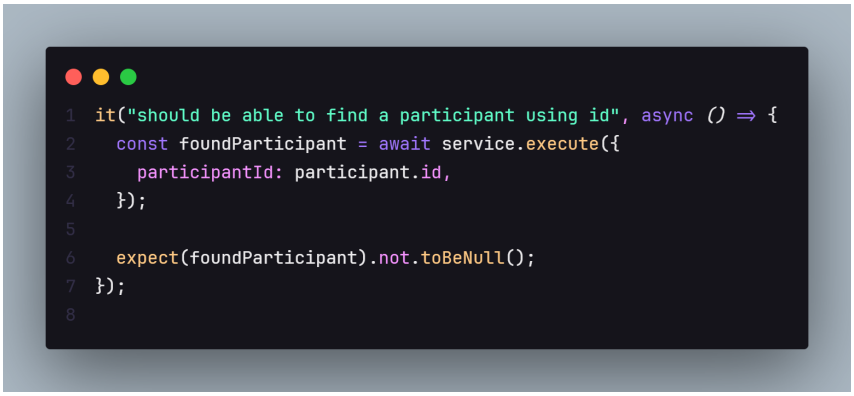
```
it("should be able to find the latest edition", async () => {
    const edition = await service.execute({
      projectTitle: project.title,
    });

    expect(edition).toHaveProperty("id", latestEdition.id);
  });
```

Figura 4.8: **Código** → TU Localizar Ultima Edição

4.4.9 Verificar Participante

O teste verifica se **FindParticipant** consegue verificar a existência de um participante baseado no seu ID. A sua garantia é que na execução retorne um objeto de participante válido, confirmando que a busca pelo ID está correta, certificando que os dados do participante possam ser acessados para operações adequeadas. A Figura 4.9 mostra um dos trechos do código mencionado na descrição.



```
it("should be able to find a participant using id", async () => {
  const foundParticipant = await service.execute({
    participantId: participant.id,
  });

  expect(foundParticipant).not.toBeNull();
});
```

Figura 4.9: **Código** → TU Verificar Participante

4.4.10 Listagem de Edições

O testo valida se **ListEditions** lista todas as edições associadas a um determinado projeto. O seu objetivo é garantir que fornecendo o ID do Projeto, retorne o número correto

de edições. Essa verificação garante a integridade dos dados e a disponibilidade das informações das edições. A Figura 4.10 mostra um dos trechos do código mencionado na descrição.

```
it("should list all editions of a project", async () => {
    const editions = await service.execute({
      projectId: project.id,
    });

    expect(editions).toHaveLength(2);
  });
```

Figura 4.10: **Código** → TU Listagem de Edições

4.4.11 Verificar Certificado

Neste teste, a função do serviço **ValidateCeritificate** é avaliada na verificação de um certificado, usando o e-mail e a matrícula de um participante, onde é aguardado um *return true*, quando as credenciais são corretas, validando a eficácia do autenticador da aplicação e garantindo uma camada de segurança. A Figura 4.11 mostra um dos trechos do código mencionado na descrição.

```
it("should be able to validate a certificate", async () => {
    const emailValidation = await service.execute({
      certificateId: certificate.id,
      email: participant.email,
    });

    const matriculaValidation = await service.execute({
      certificateId: certificate.id,
      matricula: participant.matricula,
    });

    expect(emailValidation).toEqual(true);
    expect(matriculaValidation).toEqual(true);
  });

```

Figura 4.11: **Código** → TU Verificar Certificado

4.5 Principais dificuldades e Aprendizados

Dentre as principais dificuldades durante o desenvolvimento e produção, é possível dar destaque aos seguintes pontos:

- **Incompatibilidade de Bibliotecas em Navegadores Legados**: Durante a execução do evento, foi identificado um problema ao utilizar a biblioteca 'next-top-loader', que é responsável por exibir um indicador de progresso no topo da página durante o carregamento de conteúdo, em navegadores legados como Opera Mini e versões anteriores ao Safari 10. Esses navegadores, por não suportarem todos os recursos do ECMAScript 6 (ES6), exibiam uma tela branca com a mensagem: *"Application error: a client-side exception has occurred."*. A incompatibilidade com o ES6 impediu o funcionamento adequado da biblioteca, afetando a experiência dos usuários que utilizavam esses navegadores durante o evento.

- **Volume de Requisições**: Em razão de inúmeras requisições à API, buscando os dados diretamente no banco de dados, a aplicação sobrecarregou durante o evento. A Figura 4.12 mostra o insight de experiência do usuário ao acessar a plataforma, necessitando algumas melhorias para aumentar a pontuação (*Real Experience Score*).

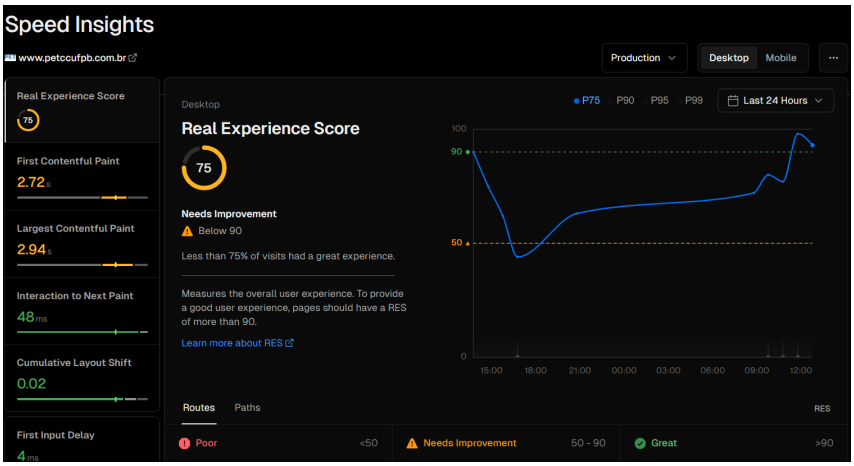


Figura 4.12: **Insights de melhorias necessárias**

Para contornar esse problema, precisamos implementar a estrutura em memória do Redis, criando um cache para requisições, preservando a necessidade de sempre solicitar os dados diretamente ao banco real. Essa decisão foi um fator importante para redução de volume de requisições desnecessárias no banco de dados, reduzindo o custo em **43,5%**, com o banco atualizando de 5 em 5 minutos com informações novas, ou já existentes. A Figura 4.13 do código mostra a solução.

```
@Module({
  imports: [
    CacheModule.registerAsync<RedisClientOptions>({
      // @ts-ignore
      useFactory: async () => ({
        store: await redisStore({
          url: process.env.REDIS_URL,
          ttl: 1 * 1 * 5 * 60 * 1000, // 5 minutes
        }),
      }),
    }),
```

Figura 4.13: **Código** → Cache atualizando de 5 em 5 minutos

Dentre os aprendizados durante o desenvolvimento e produção, é possível dar destaque aos seguintes pontos:

- **Alcance Global e Visibilidade**: O desenvolvimento da aplicação trouxe visibilidade significativa para o projeto e para os eventos acadêmicos do Centro de Informática (CI). A Figura 4.14 mostra os acessos de outros países, validando o alcance internacional da plataforma. Além disso, a Figura 4.15 destaca a maior incidência de tráfego de usuários durante o evento, evidenciando a importância de desenvolver soluções eficazes para atender às crescentes demandas dos usuários e garantir a competitividade no mercado.

Figura 4.14: **Insight Vercel sobre Visibilidade em determinadas rotas**

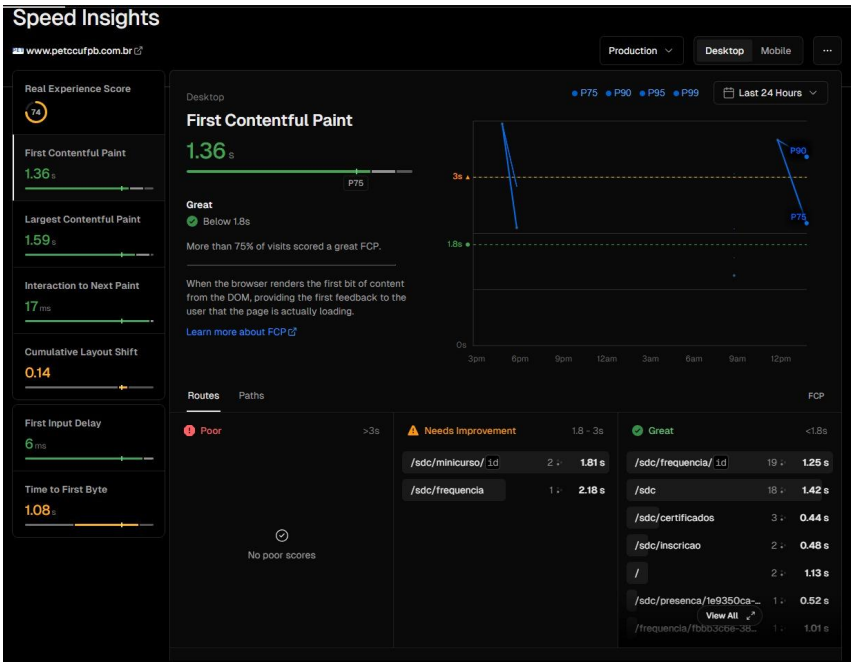


Figura 4.15: **Insight Vercel sobre maior tráfego nas rotas**

- **Necessidade de um Sistema de Gestão Eficiente**: Um aprendizado significativo durante o desenvolvimento foi a identificação da necessidade de otimizar a gestão dos eventos, simplificando o processo de cadastro de palestrantes e edições. Inicialmente, essas tarefas eram executadas manualmente, utilizando ferramentas externas como Postman, Insomnia ou Hoppscotch para interagir com a API, o que tornava o processo suscetível a erros. Para abordar essa questão, criamos a interface do usuário do dashboard administrativo (Figura 4.16), que proporciona uma visualização intuitiva e centralizada das informações. Embora a interface já esteja desenvolvida, o consumo da API ainda precisa ser implementado para permitir o cadastro e gerenciamento efetivo das informações diretamente na plataforma.

Figura 4.16: **Dashboard Administrativo da Plataforma SDC**

Capítulo 5

Conclusões e Trabalhos Futuros

O presente relatório abordou todas as fases e desafios enfrentados no desenvolvimento da **Plataforma SDC**, em um cenário de constante evolução tecnológica. Ao longo do desenvolvimento da aplicação foram identificadas novas regras de negócio, principais problemas a serem solucionados no *deploy* e a necessidade de garantir compatibilidade com navegadores mais antigos, tendo em vista o avanço do *ECMAScript*.

O foco principal deste relatório foi de descrever todo o processo de desenvolvimento com as diversas tecnologias e métodos adotados. O resultado desse esforço por 3 desenvolvedores do PET Computação permitiu que a aplicação entregasse resultados satisfatórios, referente à assiduidade do discente em eventos gerenciados por alguma equipe acadêmica, além de abrir caminho para adoção de novos recursos e melhorias diante de um retorno da comunidade.

Como trabalho futuro, recomenda-se a implementação de um *dashboard* que permita a gestão de dados dos usuários, a inserção de novos eventos e edições, além da visualização de dados e gráficos para otimizar o controle e a análise dos dados de assiduidade. Esse recurso visa substituir os processos manuais atualmente realizados por meio de plataformas de API, como Postman, Hoppscotch e Insomnia, e possibilitará uma experiência mais acessível e integrada para os gestores da plataforma. A interface do usuário (UI) já foi prototipada e desenvolvida, conforme mencionado na seção de aprendizados, sendo necessário apenas integrar o consumo da API no frontend para garantir um fluxo de trabalho simplificado. Além disso, como um avanço futuro na arquitetura do sistema, sugere-se a adoção de uma camada de abstração que possibilite a interoperabilidade com diferentes serviços de autenticação, aprimorando a segurança e a flexibilidade da aplicação.

Referências Bibliográficas

[1] 2023. "Como testar services em NestJS com Jest". Disponível em: <https://www.luiztools.com.br/post/como-testar-services-em-nestjs-com-jest/?utm_source=google&utm_medium=cpc&utm_campaign=12231680300&utm_content=157043985573&utm_term=nest%20js%20service%20test&gad_source=1&gclid=CjwKCAjwvKi4BhABEiwAH2gcww_jHLQF6RVAc-K1xNJGfhgZzYrTCrLwCl2QytQwsfekC3iVZ1Qv1hoCCW0QAvD_BwE>. Acesso em: 12 out. 2024.

[2] 2024. "Ajudando sua empresa: O que é backend e sua importância". Disponível em: <https://www.ewally.com.br/blog/ajudando-sua-empresa/backend#:~:text=O%20backend%20%C3%A9%20a%20estrutura,ambientes%20eletr%C3%B4nicos%20operem%20em%20sincronia.>. Acesso em: 10 out. 2024.

[3] 2024. "O que é ReactJS? Entenda como funciona e suas principais características". Disponível em: <https://www.hashtagtreinamentos.com/o-que-e-reactjs-javascript>. Acesso em: 10 out. 2024.

[4] 2023. "Padrões Arquiteturais: Arquitetura Software Descomplicada". Disponível em: <https://www.alura.com.br/artigos/padroes-arquiteturais-arquitetura-software-descomplicada?srsltid=AfmBOooFQGSVLfRd2CVYgrUXIh_OTj-jAK9iHDKBGAc5AkGhZgEeCW_8>. Acesso em: 10 out. 2024.

[5] 2024. "Explorando o poder do JavaScript: Guia para desenvolvimento web". Disponível em: <https://community.revelo.com.br/explorando-o-poder-do-javascript-guia-para-desenvolvimento-web/>. Acesso em: 10 out. 2024.

[6] 2024. "Node.js: Entendendo o Event Loop". . Disponível em: <https://blog.rocketseat.com.br/node-js-entendendo-o-event-loop/>. Acesso em: 10 out. 2024.

[7] 2024. “SSR: O que é e como utilizar no Next.js e ReactJS”. . Disponível em: <https://blog.rocketseat.com.br/ssr-nextjs-reactjs/>. Acesso em: 10 out. 2024.

[8] 2023. “O que é SOLID: O guia completo para você entender os 5 princípios da POO”. Disponível em: <https://medium.com/desenvolvendo-com-paixao/o-que-%C3%A9-solid-o-guia-completo-para-voc%C3%AA-entender-os-5-princ%C3%ADpios-da-poo-2b937b3fc530>. Acesso em: 10 out. 2024.

[9] 2024. “Front-end: O que é, como funciona e melhores práticas”. Disponível em: <https://www.totvs.com/blog/developers/front-end/>. Acesso em: 10 out. 2024.

[10] 2024. “O que é JSX? Entenda como funciona e suas principais características”. Disponível em: <https://www.treinaweb.com.br/blog/o-que-e-jsx>. Acesso em: 10 out. 2024.

[11] BROTHERS, A., 2023. “Server Side Rendering in React”. Disponível em: <https://asperbrothers.com/blog/server-side-rendering-in-react/>. Acesso em: 10 out. 2024.

[12] CONSOLELOG, 2021. “Cache Redis para Desempenho em API”. Disponível em: <https://consolelog.com.br/cache-redis-para-desempenho-api/>. Acesso em: 15 out. 2024.

[13] DA WEB, M., 2021. “Requisitos funcionais e não funcionais: o que são?” Disponível em: <https://www.mestresdaweb.com.br/tecnologias/requisitos-funcionais-e-nao-funcionais-o-que-sao>. Acesso em: 15 out. 2024.

[14] DESENVOLVIMENTO, C., 2023. “O que é Portabilidade?” Disponível em: <https://creapardesenvolvimento.com.br/glossario/o-que-e-portability-tecnologia/>. Acesso em: 15 out. 2024.

[15] DIGITAL, D., 2023. “O que é confiabilidade do sistema em apps móveis?” . Disponível em: <https://www.dexidigital.com.br/glossario/o-que-e-confiabilidade-do-sistema-em-apps-moveis/>. Acesso em: 15 out. 2024.

[16] DIGITAL, G., 2023. “Guia de Requisitos Mínimos para Privacidade e Segurança em Aplicações Web”. . Disponível em: <https://www.gov.br/governodigital/pt-br/privacidade-e-seguranca/ppsi/guia_requisitos_minimos_web.pdf>. Acesso em: 15 out. 2024.

[17] GROUP, D., 2024. "Saiba porque o estudo de personas é importante". Disponível em: <https://dotgroup.com.br/blog/saiba-porque-o-estudo-de-personas-e-importante/#:~:text=Uma%20persona%20tem%20como%20fun%C3%A7%C3%A3o,%C3%A0s%20demandas%20do%20p%C3%BAblico%2Dalvo>. Acesso em: 15 out. 2024.

[18] HABBEMA, 2021. "Explorando o TypeScript". Disponível em: <https://medium.com/@habbema/explorando-o-typescript-4dc21914319a>. Acesso em: 15 out. 2024.

[19] IS POWER, C., 2023. "Railway: uma das melhores, se não a melhor, hospedagem de aplicações". Disponível em: <https://www.tabnews.com.br/coffeeispower/railway-uma-das-melhores-se-nao-a-melhor-hospedagem-de-aplicacoes>. Acesso em: 15 out. 2024.

[20] JACQUES. "Arquitetura em Camadas". Disponível em: <http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/arqu/camadas.html>. Acesso em: 16 out. 2024.

[21] NETWORK, M. D., 2024. "Segurança de sites". Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/Server-side/First_steps/Website_security>. Acesso em: 15 out. 2024.

[22] PATEL, N., 2021. "Usabilidade: o que é?" Disponível em: <https://neilpatel.com/br/blog/usabilidade-o-que-e/#:~:text=Usabilidade%20%C3%A9%20o%20termo%20utilizado,o%20cumprimento%20de%20seu%20objetivo.>. Acesso em: 15 out. 2024.

[23] ROCKETSEAT, 2020. "Entendendo os Fundamentos do NestJS: Construindo Aplicações Robustas". Disponível em: <https://blog.rocketseat.com.br/entendendo-os-fundamentos-do-nestjs-construindo-aplicacoes-robustas-2/>. Acesso em: 15 out. 2024.

[24] SAPHIR, 2023. "Por que você deve considerar o uso do PostgreSQL". Disponível em: <https://blog.saphir.com.br/por-que-voce-deve-considerar-o-uso-do-postgresql/>. Acesso em: 15 out. 2024.

[25] SERVICES, A. W., 2023. "O que é Interoperabilidade?" Disponível em: <https://aws.amazon.com/pt/what-is/interoperability/#:~:text=A%20interoperabilidade%20permite%20que%20diversos,dados%20para%20atingir%20objetivos%20comuns.>. Acesso em: 15 out. 2024.

[26] STANLEY, 2023. “Monorepo: O que é e devo usar?” Disponível em: <https://dev.to/stanley/monorepo-o-que-e-devo-usar-133c>. Acesso em: 10 out. 2024.

[27] TALLER, 2020. “Como funciona o Styled Components por debaixo dos panos”. Disponível em: <https://blog.taller.net.br/como-funciona-o-styled-components-por-debaixo-dos-panos/>. Acesso em: 15 out. 2024.

[28] TOOLS, L., 2021. “Deploy de aplicação Next.js na Vercel”. Disponível em: <https://www.luiztools.com.br/post/deploy-de-aplicacao-next-js-na-vercel/>. Acesso em: 15 out. 2024.

[29] TREINAWEB, 2022. “Trabalhando com Prisma ORM e NestJS”. Disponível em: <https://www.treinaweb.com.br/blog/trabalhando-com-prisma-orm-e-nestjs>. Acesso em: 15 out. 2024.