UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE QUIXADÁ

DEPARTAMENTO DE COMPUTAÇÃO

CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

GABRIEL DE SOUZA LINS

UTILIZANDO REACTJS PARA O DESENVOLVIMENTO DE UM SISTEMA DE ALOCAÇÃO E RESERVA DE SALAS NO CAMPUS DA UFC EM QUIXADÁ

QUIXADÁ

2019

GABRIEL DE SOUZA LINS

UTILIZANDO REACTJS PARA O DESENVOLVIMENTO DE UM SISTEMA DE ALOCAÇÃO E RESERVA DE SALAS NO CAMPUS DA UFC EM QUIXADÁ

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de informação do Centro de Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Sistemas de informação.

Orientador: Prof. Dr. Jefferson de Carvalho Silva

QUIXADÁ

2019

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

L731u Lins, Gabriel de Souza.
Utilizando ReactJS para o desenvolvimento de um sistema de : alocação e reserva de salas no campus da UFC em Quixadá / Gabriel de Souza Lins. – 2019.
36 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Sistemas de Informação, Quixadá, 2019.
Orientação: Prof. Dr. Jefferson de Carvalho Silva.

1. JavaScript (Linguagem de programação de computador). 2. Software de aplicação - Desenvolvimento. 3. Universidade. I. Título.

CDD 005

À minha família, por todo o apoio durante esses anos de muita luta. Em especial aos meus pais por sempre terem feito de tudo pra nos educar, tanto eu quanto minha irmã.

AGRADECIMENTOS

Ao Prof. Dr. Jefferson de Carvalho Silva por me orientar durante este ano em minha monografia. Por dedicar seu tempo para me ajudar, dando sugestões de melhorias e de escrita para o trabalho em questão.

Aos meus pais, irmã e namorada, por entenderem meus momentos de ausência para se dedicar aos estudos. E por saberem que todo meu esforço hoje valerá a pena no meu futuro, tanto profissional quanto pessoal.

Aos professores desta instituição por se esforçarem em nos passar seus conhecimentos da melhor forma possível, para que eu possa me capacitar e me tornar um excelente profissional. Na nossa vida pessoal pelos conselhos que nos dão.

RESUMO

Com o avanço das tecnologias para web, surgiram soluções para diversos tipos de problemas das empresas e da sociedade. Várias dessas soluções facilitam atividades desenvolvidas por um time em uma empresa ou mesmo usuários comuns. Aplicações como: venda de passagens aéreas e terrestres, reserva de bilhetes no cinema, etc. De outro lado temos as soluções desenvolvidas dentro da academia que servem para facilitar atividades na universidade, ou seja, soluções implementadas por alunos dos cursos de computação. Dentro deste contexto, este trabalho apresenta a construção de uma aplicação para uso interno dos servidores da universidade com o intuito de organizar a reserva e alocação de salas, dentre outros espaços.

Palavras-chave: JavaScript. ReactJS. Tecnologias Web.

ABSTRACT

With the advancement of web technologies, solutions have emerged for various types of business and society problems. Many of these solutions facilitate activities developed by a team in a company or even common users. Applications such as: the sale of air and land tickets, reservation of tickets at the cinema, etc. On the other hand, we have the solutions developed within the academy that serve to facilitate activities in the university, that is, solutions implemented by students of the courses of computation. Within this context, this work presents the construction of an application for internal use of university servers with the purpose of organizing the reservation and allocation of rooms, among other spaces.

Keywords: JavaScript. ReactJS. Web Technologies.

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
CSS	Cascading Style Sheets
DOM	Document Object Model
ES6	ECMAScript 6
HTML	Hypertext Markup Language
JSF2	JavaServer Faces
JSON	JavaScript Object Notation
JSX	JavaScript Sintax Extension
NPI	Núcleo de Práticas em Informática
REST	Representational State Transfer
RFC	Request For Comments
URI	Uniform Resource Identifier
XML	Extensible Markup Language

SUMÁRIO

1 INTRODUÇÃO

Os frameworks para desenvolvimento de aplicações web estão cada dia mais robustos, oferecendo mais possibilidades para o desenvolvedor. As tecnologias usadas para o desenvolvimento de sistemas web eram apenas Hypertext Markup Language (HTML), Cascading Style Sheets (CSS) e JavaScript. Não se tinha o conceito de um framework front-end baseado em componentes, como temos até então. Segundo (GOEL, 2019), atualmente os frameworks mais conhecidos são: ReactJS (Desenvolvido pelo Facebook), Angular (Desenvolvido pelo Google) e VueJS. O uso de frameworks/bibliotecas torna mais simples o desenvolvimento de aplicações web tanto de pequeno quanto de grande porte, pois os mesmos permitem o reuso de componentes de telas da aplicação.

Em (BUNA, 2017), são citados os principais motivos pelo qual o ReactJS vem sendo cada vez mais usada pelas empresas. Ele é um framework/biblioteca baseado em JavaScript que se tornou uma das tecnologias mais promissoras nos últimos anos por oferecer um ótimo desempenho em relação às tecnologias que existiam e também pela sua facilidade de aprendizado. O ReactJS vem sendo utilizado pelas pequenas, médias e grandes empresas dos vários ramos de atuação, seja para construir sistemas internos como para aplicações que são utilizadas por usuários comuns. Um ponto a destacar é que um dos módulos bastante úteis para o desenvolvimento de aplicações chamado Redux, implementa a arquitetura Flux nas aplicações, foi criado pelo Facebook, sendo assim possui uma boa integração com o React mas também pode ser utilizado com outras bibliotecas já existentes no mercado.

Na Universidade Federal do Ceará em Quixadá notou-se que várias atividades do campus são feitas de forma digital, ou seja, existindo um sistema/aplicação específica para a execução da atividade, facilitando assim o trabalho dos servidores. Entretanto observou-se que a atividade de alocação e reserva de salas e outros espaços do campus, não possui um sistema. Sendo assim sua execução é trabalhosa pois tudo tem de ser feito em uma planilha eletrônica e não há uma forma centralizada onde todos poderiam ter acesso à visualização. Sendo assim, este trabalho tem como objetivo apresentar a implementação e testes de um sistema que permite a possibilidade de reservar salas de aula, dispensando a presença do requerente à sala do servidor responsável por essa atividade.

1.1 Objetivos

1.1.1 Objetivo geral

Utilizar tecnologias web no intuito de desenvolver uma ferramenta de apoio a servidores técnico administrativos e professores da universidade para que se possa ter uma forma mais prática de alocar e reservar salas no campus da universidade. Com isso o campus de Quixadá possuirá uma ferramenta para agilizar esse processo e assim também permitirá que o Núcleo de Práticas em Informática (NPI) possa melhorar a ferramenta posteriormente para que sejam feitas evoluções no software.

1.1.2 Objetivo específico

Desenvolver um sistema de alocação e reserva de salas utilizando ReactJS juntamente com uma Application Programming Interface (API), onde será possível fazer o cadastro de atividades ou aulas em salas do campus e se no início do semestre, fazer as alocações necessárias.

1.1.2.1 Efetuar testes de validação da ferramenta junto ao servidor responsável.

Após a finalização do front-end e back-end, serão realizados testes de validação para verificar as funcionalidades da aplicação.

1.1.2.2 Aplicação front-end

Será implementada uma aplicação utilizando ReactJS onde será consistirá na construção das telas do sistema.

1.1.2.3 Aplicação back-end

Para que o sistema funcione, se torna necessário o desenvolvimento de um serviço web baseado na arquitetura Representational State Transfer (REST) (RODRIGUEZ, 2015), este serviço que proverá e manipulará os dados da aplicação para servir à aplicação front-end. Será desenvolvido utilizando a tecnologia NodeJS juntamente com um banco de dados MySql.

2 FUNDAMENTAÇÃO TEÓRICA

Neste Capítulo serão apresentadas as tecnologias que serão utilizadas e os conceitos em que será baseado o desenvolvimento deste trabalho.

2.1 ReactJS

O ReactJS é uma biblioteca Front-end baseada na linguagem JavaScript, seu pricipal objetivo é permitir o desenvolvimento interfaces baseadas em componentes para aplicações web. Constitui uma base de conhecimento necessária para sua utilização, os conceitos de componentização, estado, propriedades, sintaxe JavaScript Sintax Extension (JSX), etc. Em (LIMA, 2017), o ecossistema do React é composto pelas seguintes tecnologias:

- React
- JSX
- ECMAScript 6 (ES6)
- Webpack
- Flux/Redux
- Axios/Fetch
- Jest/Mocha

O ReactJS, é uma biblioteca JavaScript voltada para a construção de interfaces web. Esta biblioteca pode ser utilizada tanto para o desenvolvimento de aplicações web quanto para aplicações mobile. O React é capaz de manipular elementos visuais de forma que se possa criar componentes reutilizáveis.

É uma tecnologia que se esforça para fornecer velocidade, simplicidade e escalabilidade. Algumas de suas características mais notáveis são JSX, componentes com estado, modelo de objeto de documento virtual. (KHUAT, 2018)

Um dos maiores motivos para utilização do ReactJS neste trabalho é a performance dessa tecnologia, pois trabalha com o que chamamos de Virtual Document Object Model (DOM), ou seja, o ReactJS trabalha com uma cópia virtual do DOM da página web, copiando-o para a memória do computador, fazendo a manipulação do elementos em tela com um ótimo desempenho.

2.2 JSX

O JSX é uma extensão do JavaScript feita para compor scripts através de uma sintaxe simples e que seja parecida com HTML, no entanto por trás do esquema do JSX, as tags usadas no arquivo JSX são interpretadas como código JavaScript que depois será convertido para HTML. Esta tecnologia permite unificar a lógica do componente com a escrita das tags da interface que no caso do React é código JavaScript da mesma forma. Sendo assim não precisa separar um arquivo com a lógica da aplicação e um arquivo HTML separado. (REACTJS.ORG, 2014)

O React não requer o uso do JSX. Porém, a maioria das pessoas acha prático como uma ajuda visual quando se está trabalhando com uma UI dentro do código em JavaScript. Ele permite ao React mostrar mensagens mais úteis de erro e aviso. (REACTJS.ORG, 2014)

2.3 ES6

Em (ECMA-INTERNATIONAL, 2015) o ES6 ou EcmaScript 2015 é definida como a sexta versão da especificação da linguagem JavaScript, e que trouxe várias novas features para serem usadas na linguagem. A primeira versão do EcmaScript foi lançada em 1997 e desde lá vem trazendo novos recursos para a o JavaScript. O ES6 definiu vários recursos novos para a linguagem, como orientação a objetos, etc. Grande parte dos recursos permitem que a tecnologia seja adepta de várias plataformas. (ECMA-INTERNATIONAL, 2015)

Grande parte das funcionalidades trazidas pelo EcmaScript 6 permitem que várias coisas que antes não poderiam ser utilizadas na linguagem ou que demandavam mais trabalho, agora são implementadas nesta versão. Ela nos permite trabalhar de uma forma bem mais próxima do que é feito com outras linguagens mais robustas, como Java. Ela Facilita a declaração de classes, arrays e iteradores, novas formas de tratar dados e objetos, e suporte e outros caracteres suplementares unicode e expressões regulares. (ECMA-INTERNATIONAL, 2015)

O EcmaScript foi criado na antiga NetScape para ser integrado ao navegador da empresa que na época seria lançado a versão 2.0 do navigator. Porteriormente a Microsoft passou a integra-lo ao Internet Explorer 3.0. Hoje a maioria dos navegadores atuais dão suporte ao EcmaScript na sua última versão.

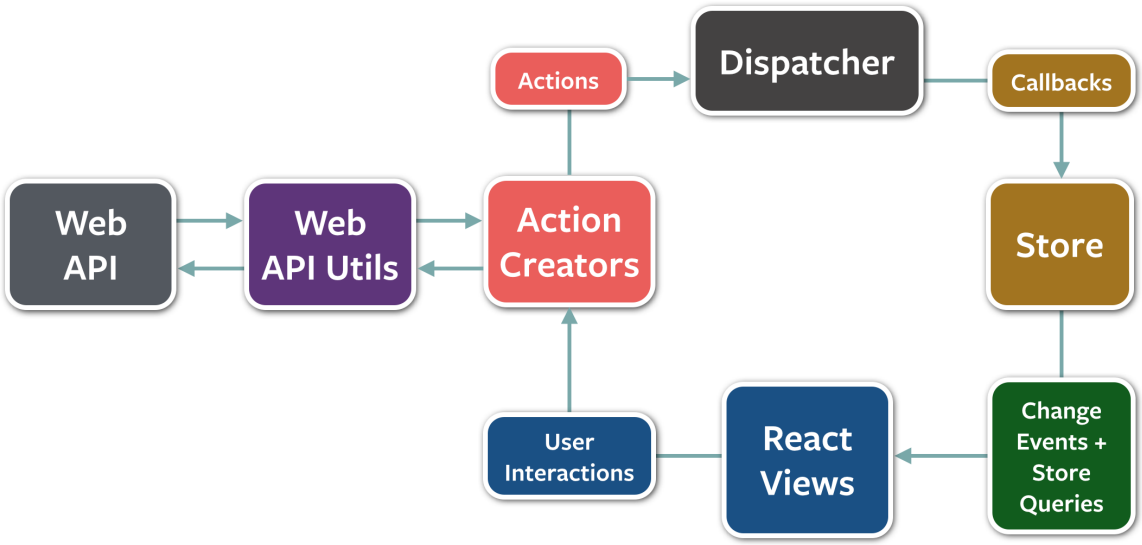
De acordo com (ECMA-INTERNATIONAL, 2015) o desenvolvimento da especificação da linguagem ECMAScript começou em novembro de 1996. A primeira edição deste padrão ECMA foi adotada pela Assembléia Geral Ecma de junho de 1997.

2.4 Flux/Redux

Flux é a arquitetura de aplicativo que o Facebook usa para criar aplicativos da Web do lado do cliente. Ele complementa os componentes de visualização compostos do React utilizando um fluxo de dados unidirecional. É mais um padrão do que um framework formal. (FACEBOOK-INC, 2014)

A Figura 1 mostra o funcionamento do Flux, que segue um princípio bem simples: os dados devem fluir em um único sentido seguindo mais ou menos o diagrama abaixo. O Redux mantém o estado do componente disponível para toda a aplicação. As actions, responsáveis por chamar os reducers. Os reducers são responsáveis por tratar as informações do estado e a store responsável por armazenar todas as informações do estado.

Figura 1 – A arquitetura Flux



Fonte: Facebook (2014)

2.5 REST/RESTFul

REST é uma gama de conceitos arquiteturais que são usados para definir como uma aplicação que utiliza esse padrão arquitetural deve seguir para garantir a aplicação adequada desses padrões. Sendo assim podendo ser usados para conceber serviços na web através do protocolo HTTP.

O REST concebe vários recursos para serem utilados junto ao protocolo HTTP, definindo como devem ser implementadas os métodos deste protocolo e como devem se comportar

nas chamadas para os serviços desenvolvidos. Cada método HTTP (GET, POST, PUT, DELETE, etc) que corresponde a um recurso de uma aplicação deve seguir as regras estabelecidas pelo padrão REST.

Na forma mais pura atual, uma implementação concreta de um serviço da web REST segue quatro princípios básicos de design (RODRIGUEZ, 2015):

- Usar métodos HTTP de forma explícita.
- Ser Stateless
- Expor Uniform Resource Identifier (URI) do tipo estrutura de diretório.
- Transferir dados via Extensible Markup Language (XML) e/ou JavaScript Object Notation (JSON).

Por ser um padrão que possibilita trabalhar com respostas de requisições no formato JSON, consegue-se atender a vários tipos de clients de aplicações, por exemplo: uma aplicação web, mobile (Android ou IOS).

2.5.1 Usar métodos HTTP de forma explícita

Os serviços web que utilizam o padrão REST, fazem o uso explícito dos métodos HTTP seguindo o protocolo que foi definido pela especificação Request For Comments (RFC) 2616. O método GET por exemplo corresponde ao retorno de um conjunto de dados vindos do servidor para através de uma chamada feita pela aplicação cliente. O método POST por sua vez faz a inserção de algum dado no servidor vindos de uma chamada do cliente que neste método é o agente que faz o envio dos dados para o recurso.

A REST pede aos desenvolvedores para usar métodos HTTP de forma explícita e de uma maneira consistente com a definição do protocolo. Esse princípio de design básico da REST estabelece um mapeamento individual entre as operações criar, ler, atualizar e excluir (CRUD) e os métodos HTTP. (RODRIGUEZ, 2015) De acordo com este mapeamento:

- Para criar um recurso no servidor, use POST.
- Para recuperar um recurso, use GET.
- Para alterar o estado de um recurso ou atualizá-lo, use PUT.
- Para remover ou excluir um recurso, use DELETE.

2.5.2 Ser stateless

Os serviços web disponibilizados por um servidor precisam manter uma boa organização pois os servidores que hospedam as APIs devem estar em conformidade com as boas práticas feitas no gerenciamento dos recursos do servidor. Sendo assim, um serviço que disponiliza um recurso web para um cliente utilizando o padrão REST, deve deixar claros todos os dados que precisam ser enviados para a chamada de um recurso, ou seja, cada requisição deve conter todos os dados necessários para o servidor não precisar manter estados localmente.

Uma solicitação completa e independente não requer que o servidor, enquanto processa a solicitação, recupere qualquer tipo de contexto ou estado do aplicativo. Um aplicativo (ou cliente) de serviço da web REST inclui, dentro dos cabeçalhos e corpo HTTP de uma solicitação, todos os parâmetros, contextos e dados de que o componente no lado do servidor precisa para gerar uma resposta. Nesse sentido, a condição stateless melhora o desempenho do serviço da web e simplifica o design e a implementação dos componentes no lado do servidor, pois a ausência de estado no servidor remove a necessidade de sincronizar os dados da sessão com um aplicativo externo. (RODRIGUEZ, 2015)

2.6 NodeJS

O avanço da linguagem JavaScript e a evolução da engine V8 tornou possível a utilização de JavaScript no lado do servidor. O NodeJS é uma Runtime, ou seja, uma plataforma de aplicação onde o código JavaScript é interpretado e executado sem necessidade de utilização de um navegador web, possibilitando construir aplicações back-end assim com é feito com Java, Ruby, PHP, etc.

2.6.1 Vantagens

Seu entendimento é bem simples. Quanto ao código não é diferente, se você já programou JavaScript alguma vez sabe como o código é fácil de ser escrito e simples de ser entendido. Utilizando JavaScript no back-end você leva a simplicidade dessa linguagem ao servidor. (FERNANDES, 2017)

É uma tecnologia Non-blocking I/O. A grande maioria das linguagens como PHP e Ruby possuem um processo bem definido de execução:

- O servidor recebe uma requisição (geralmente uma chamada HTTP).

- A partir disso, o mesmo processa a requisição e calcula sua resposta.
- A resposta é devolvida ao front-end e pronto, a conexão é perdida.

O que percebemos nesse processo é que existe um processo de início, meio e fim. No NodeJS nós temos um loop central que nunca para de executar e controla todo tipo de requisição, podemos realizar tarefas em concorrência. (FERNANDES, 2017)

Reaproveitamento de conhecimento, utilizando NodeJS poupa-se grande parte dos esforços caso opte por estudar tecnologias front-end como ReactJS, VueJS, Angular, ou React Native para desenvolvimento mobile, já que o JavaScript é o coração em todas essas ferramentas. (FERNANDES, 2017)

2.7 AdonisJS

O NodeJS possui um micro-framework chamado Express, onde é podemos criar aplicações back-end de forma simples e rápida.

O Adonis é um framework mais robusto com uma série de funcionalidades prontas, extremamente baseado em frameworks famosos de outras linguagens como o Laravel, Rails ou Django. Diferente do ExpressJS o Adonis assim que instalado já vem com uma estrutura pronta e o desenvolvedor é limitado a utilizar essa organização para seus arquivos, além de já possuir uma série de funcionalidades pré-implementadas como autenticação, ORM, validação, envio de e-mail, logging, etc. (FERNANDES, 2018)

2.7.1 Vantagens

- Estrutura de arquivos bem padronizada. Para aplicações maiores, este framework possui uma estrutura de pastas que facilita muito o desenvolvimento, trabalhando com dependency injection de forma simples.
- ORM extremamente poderoso para trabalhar com SQL. O Lucid como é nomeado, trabalha com qualquer banco de dados SQL, utilizando o padrão active record, onde são geradas migrations onde serão definidos os campos das tabelas e toda vez que precisar incluir ou excluir um campo de alguma tabela só é necessário gerar uma migration e configurar a ação a ser executada.
- Estrutura para lidar com websockets (real-time).

3 TRABALHOS RELACIONADOS

Neste Capítulo serão apresentados os trabalhos relacionados, conceituando seus principais pontos e como estão relacionados a este trabalho.

3.1 Sistema de reserva e acompanhamento de salas

Em (ALMEIDA, 2016) é descrita uma proposta de um sistema de reserva de salas de estudo no campus da UFC em Quixadá, para auxiliar na reserva de salas quando os alunos necessitam de formar grupos de estudo para estudar para uma disciplina.

A proposta utiliza tecnologias como, JavaServer Faces (JSF2), Postgres, SQL Power Architect, PrimeFaces, e Apache Tomcat.

A aplicação proposta pelo trabalho relacionado está diretamente relacionada ao trabalho que foi desenvolvido, pois teve como objetivo desenvolver uma ferramenta que faça a reserva e alocação de salas para agilizar esse processo. Que surgiu da necessidade de fazer a reserva de forma digital.

Porém, neste trabalho relacionado, seu foco é facilitar o agendamento de salas de estudo que estão localizadas ao lado da biblioteca universitária.

3.2 SIGAT (Sistema de gestão e agendamento de TCCs)

É apresentada uma proposta de sistema que utiliza Spring Boot, MySql, HTML, CSS, JavaScript e Materialize. Tem como proposta o desenvolvimento de um sistema de agendamento de TCCs utilizando tecnologias web. Nele os alunos poderiam agendar as apresentações de seus TCCs. Já que a forma antiga de se realizar essa atividade é utilizando um documento de texto online.

O projeto surgiu após uma avaliação sobre a maneira como estava sendo realizado o processo de agendamento dos TCCs, no campus da UFC de Quixadá. Após essa análise inicial, percebeu-se que o processo requer muito trabalho manual, é custoso e demorado, além de possuir diversos pontos de melhoria, sobretudo com a inclusão de uma ferramenta que seja capaz de automatizar e organizar o trabalho realizado pelos responsáveis envolvidos. (VIANA, 2017)

A proposta se relaciona com o trabalho em questão, pelo conceito de utilizar tecnologias web para desenvolver uma solução para agilizar um processo que ainda é feito de forma manual. Desta forma o foco deste trabalho relacionado é de facilitar o agendamento de defesa de

TCCs e o trabalho que foi realizado aqui se diferencia, pois tem foco na alocação e reserva de salas no campus.

3.3 Alocação de disciplinas maximizando opções de matrícula

O trabalho apresentado mostra a proposta para melhorar/solucionar a alocação de disciplinas de forma que as disciplinas sejam dispostas de forma mais organizada. Permitindo assim que os alunos tenham mais opções para fazerem suas matrículas no semestre.

Antes do início de cada semestre a direção do campus juntamente com as coordenações dos cursos se reunem para decidir as atividades que serão realizadas e como serão realizadas. Uma das atividades é organizar a demanda de disciplinas e os professores que irão ministrar cada disciplina.

Muitas universidades se deparam com esse problema, que faz parte da classe de Timetabling problems, e quanto maior o número de variáveis envolvidas(disciplinas, turmas e professores) maior será a dificuldade de se obter uma solução. (SILVA, 2016)

Esse artigo aborda o problema de alocação de disciplinas enfrentado pela universidade. Nele são discutidas soluções para decompor o problema em subproblemas para ajudar a chegar a uma solução melhor com a junção de todas soluções.

Este trabalho ajuda a entender como se pode desenvolver uma solução que possa ajudar na alocação de salas também, que é importante para a construção do sistema e como irá funcionar a alocação. Sendo assim, seu foco não é reserva de salas, mas de propor uma solução para dispor melhor as disciplinas.

3.4 Tabela de semelhanças e diferenças entre os trabalhos

Figura 2 – Comparação dos trabalhos

	Tr. Proposto	R. Almeida	B. Viana	J. Silva
Desenvolvimento	Sim	Sim	Sim	
Engloba todos os espaços do campus	Sim			
Faz uso do ReactJS	Sim			
Realiza pesquisa para construção de um algoritmo				Sim

Fonte: Elaborada pelo autor.

4 METODOLOGIA

A construção do sistema descrito neste trabalho depende de algumas tecnologias e deve-se compreender e entender bem o funcionamento de cada uma, para que se possa obter um bom software. Conhecer o ReactJS e o NodeJS é bastante importante pois as telas (front-end) foram desenvolvidas com o ReactJS e o web service foi desenvolvido com o NodeJS e AdonisJS.

4.1 Tela de todas as salas alocadas/reservadas

É a primeira tela do sistema onde podem ser visualizadas todas as salas alocadas/reservadas, ou seja, mostra somente as salas que possuem horários reservados ou alocados para aulas.

4.2 Tela de todas as salas cadastradas no sistema

Nesta tela são apresentadas todas as salas cadastradas na aplicação. Sendo assim um usuário admin pode realizar reservas ou alocações e o professr ou aluno, fazer uma solicitação.

4.3 Tela de login

Têm dois acessos permitidos a partir desta tela, usuário servidor e usuário professor. Possui dois campos: um para o email e um para a senha. Nesta parte são feitas algumas validações básicas quanto aos dados inseridos.

4.4 Tela de cadastro de sala

Pode ser realizado o cadastro de uma sala ou espaço da universidade a fim de deixar registrado na aplicação. Nesta tela não acontece a reserva da sala e sim somente a ação para adicioná-la na aplicação.

4.5 Tela de cadastro de usuário

Pode ser realizado o cadastro de um usuário professor. Somente o admin tem permissão para realizar esta tarefa.

4.6 Tela de reserva de sala

Foi construida a tela que permite a reserva de uma sala na aplicação. O usuário professor pode solicitar uma reserva em que o servidor deverá se possível, aprová-la, e o usuário servidor pode fazer a reserva diretamente caso necessite.

4.7 Tela de alocação de sala

Foi construida a tela que permite a alocação de uma sala na aplicação, semelhante à tela de reserva. Somente o servidor pode realizar esta ação.

4.8 Tela de solicitação de uma reserva

Nesta tela o professor/servidor ou aluno poderá realizar uma solicitação de reserva de uma sala para o servidor responsável, onde o servidor poderá aprovar ou cancelar.

4.9 Tela de alteração dos dados da sala

Somente o usuário servidor possui acesso a esta tela. Nela é possível alterar os dados da sala, caso ocorra algum equívoco no cadastro.

4.10 Contrução das telas com Bootstrap

Existem várias bibliotecas css para auxiliar no desenvolvimento das telas de uma aplicação, sendo assim como pré-definido foi utilizada a biblioteca Bootstrap para estilização das telas. Dessa forma foi integrada ao ReactJS para o uso de seus recursos e componentes.

Dentre as tecnologias existentes para front-end, foi desenvolvido uma biblioteca chamada Bootstrap que já possui seus componentes pré-definidos, sendo assim basta adicioná-lo ao projeto. Com isso foi feito o uso dos componentes da biblioteca para a construção de todos os outros componentes específicos da aplicação.

4.11 Construção da API

A construção da API foi feita utilizando o NodeJS e AdonisJS juntamente com o banco de dados MySQL. Quando se trata de desenvolvimento de aplicações ReactJS se faz

necessária a criação de uma API que é responsável por expôr recursos de funcionalidades implementadas no Back-End. No padrão REST, que foi definido previamnente, devemos criar rotas de comunicação que utilizam o protocolo HTTP.

Os endpoints foram definidos de acordo com cada método HTTP utilizado e a qual recurso do Back-End utilizam. Cada rota retorna um objeto JSON representando os dados solicitados pela aplicação no Front-End.

5 RESULTADOS

Neste Capítulo serão apresentados os resultados deste projeto.

5.1 Resultados da pesquisa sobre a atividade de alocação e reserva de salas

Inicialmente foi realizada um pesquisa sobre como é realizada a alocação e reserva de salas hoje no campus da UFC-Quixadá. Foi feita com o servidor técnico administrativo responsável por realizar todas as alocações no início do semestre e por fazer as reservas durante o semestre.

A construção do trabalho em questão exigiu que fosse feita uma pesquisa para saber como é realizada a atividade; se a alocação é feita de forma rápida; a frequência em que é realizada; quais os problemas e dificuldades enfrentados pelo responsável, etc.

5.1.1 Pesquisa realizada

Abaixo serão apresentadas as perguntas e as respostas da pesquisa realizada:

Figura 3 – Pergunta 1

	Bom	Regular	Ruim
O que você acha da reserva atualmente ser feita em uma planilha eletrônica?		X	

Fonte: Elaborada pelo autor

A Pergunta 1 foi elaborada para saber o nível de satisfação do servidor quanto a atividade de alocação e reserva.

Figura 4 – Pergunta 2

	Uma vez a cada semestre	Algumas vezes durante o semestre	Muitas vezes durante o semestre
Qual a frequência das reservas ou alocações de salas no campus?			X

Fonte: Elaborada pelo autor

Na Pergunta 2 podemos observar que a atividade é realizada várias vezes durante o semestre, ou seja, se torna uma atividade desagradável pois é feita de forma manual.

Figura 5 – Pergunta 3

	Sim	Não	Talvez
Você acha que a alocação ou reserva leva muito tempo para ser feita?	X		

Fonte: Elaborada pelo autor

Podemos notar na Pergunta 3 que a atividade é demorada quando realizada manualmente em uma planilha eletrônica. Dessa forma, se juntar esta com a Pergunta 2, vemos que a atividade se torna bem massante.

Pergunta 4 "Quais problemas você enfrenta ao realizar esta atividade?". Resposta: Lentidão na hora de abrir o documento de reserva de sala. Dificuldade para encontrar uma sala ideal para as turmas no início do semestre.

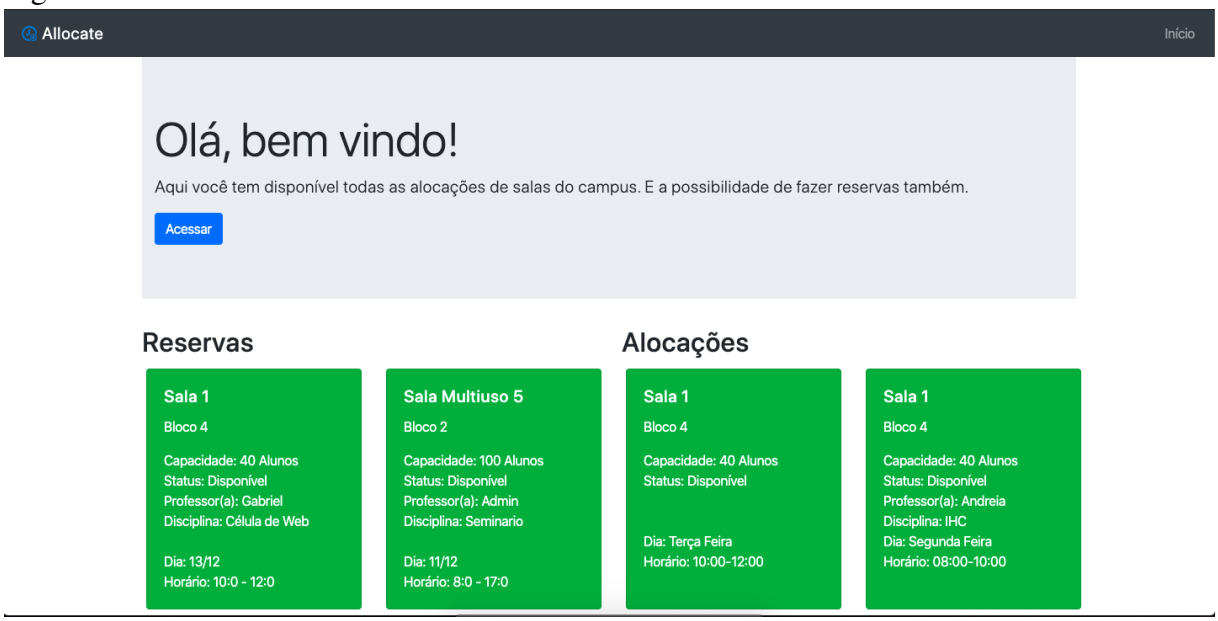
Pergunta 5 "O que você sobre a universidade possuir um sistema que facilite esta atividade?". Resposta: Ótimo, ajudaria bastante.

5.2 Telas da aplicação

Abaixo estão descritas todas as telas do software construído.

Na Figura 6 é apresentado tela inicial onde qualquer usuário que acessar o sistema poderá ver as salas que foram reservadas e as que foram alocadas.

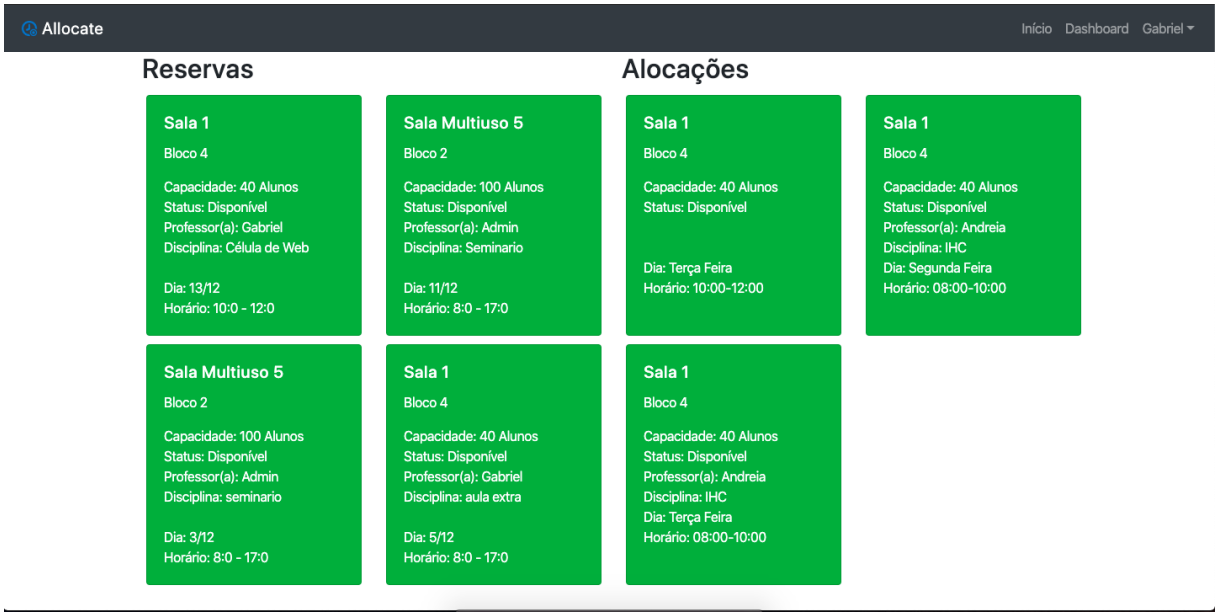
Figura 6 – Tela 1



Fonte: Elaborada pelo autor

Na Figura 7 é a mesma tela porém sendo apresentada quando o usuário está logado.

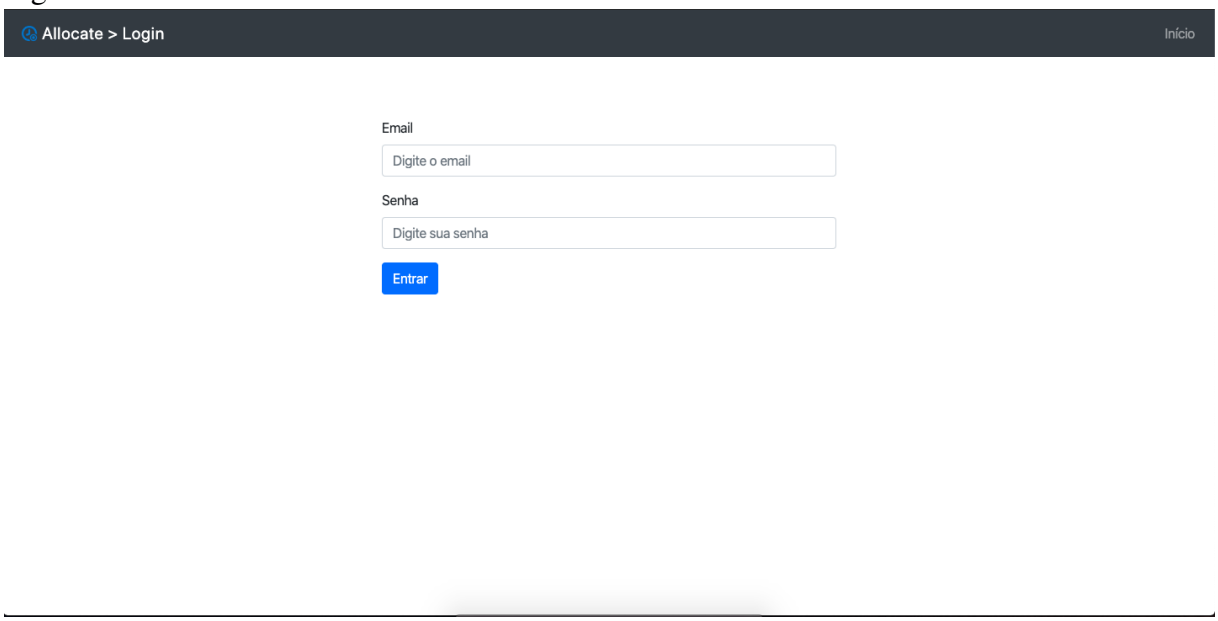
Figura 7 – Tela 2



Fonte: Elaborada pelo autor

Na Figura 8 é apresentado a tela de login, tela de login é única, pois dentro da aplicação é feito um controle de acesso a partir do nível de acesso de cada usuário.

Figura 8 – Tela 3

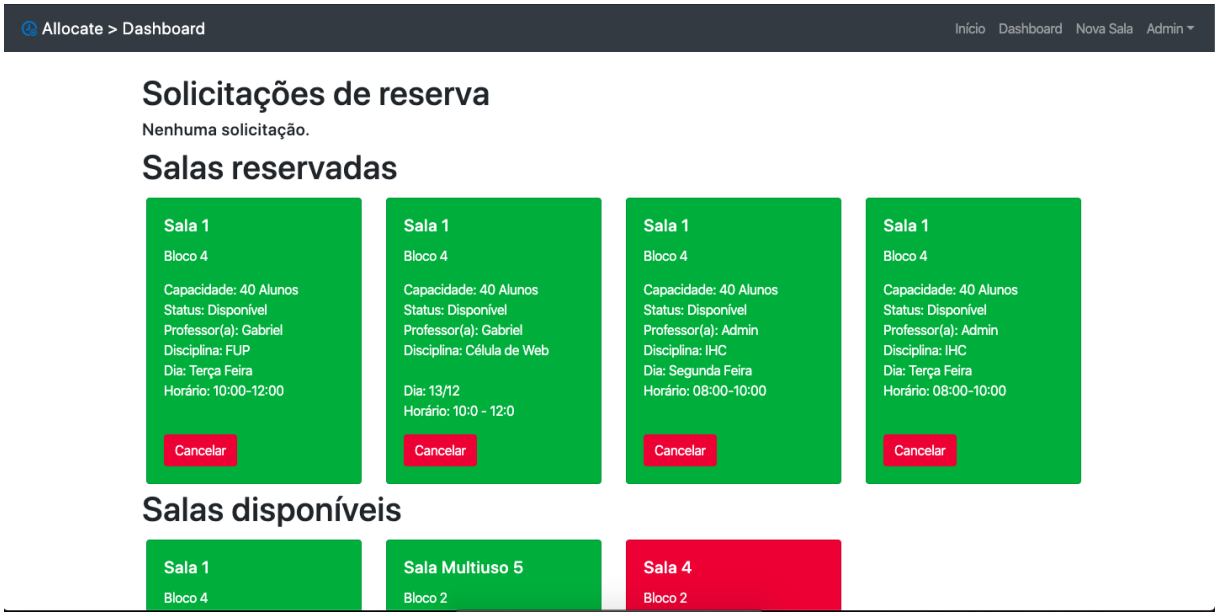


Fonte: Elaborada pelo autor

Nas Figuras 9 e 10 é apresentada a tela principal, ou seja, o dashboard do admin, que nesse caso é acesso do servidor responsável por fazer as reservas. Nesta tela existem 3 divisões, a primeira mostra todas as solicitações de reserva que o admin recebe, ou seja, que

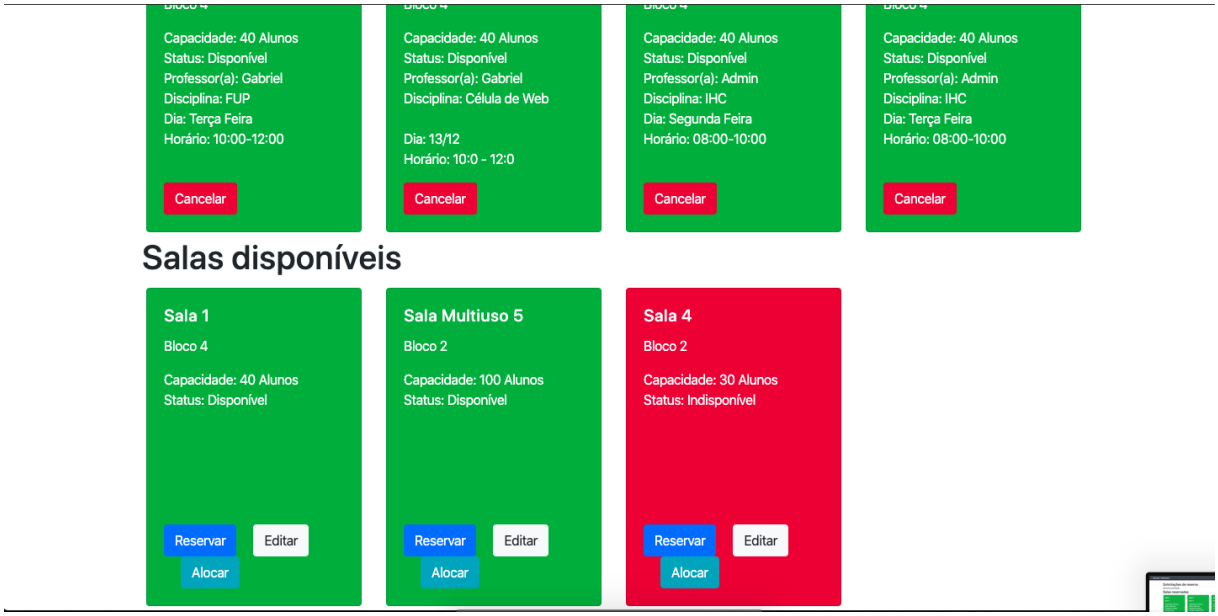
os professores/servidores ou alunos podem solicitar. Podendo o admin, aprovar ou cancelar uma solicitação. Na segunda pode-se visualizar todas as reservas e alocações que estão feitas, novamente o admin podendo cancelar uma reserva que já está feita. Na terceira são apresentadas todas as salas cadastradas no sistema, onde o admin pode editar os detalhes da sala, reservar ou alocar.

Figura 9 – Tela 4



Fonte: Elaborada pelo autor

Figura 10 – Tela 5



Fonte: Elaborada pelo autor

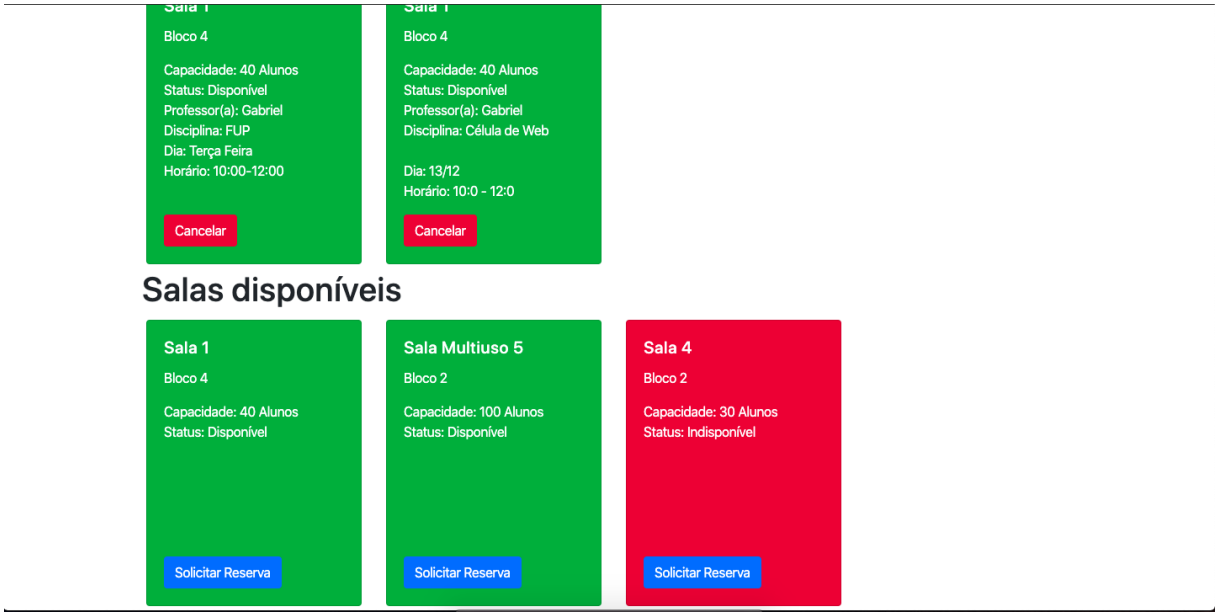
Nas Figuras 11 e 12 é apresentada a tela de dashboard do professor ou aluno, onde

a mesma tem suas restrições. Nesse caso, o usuário desse tipo tem seu acesso limitado. Nessa tela existem 3 divisões assim como a tela do admin, porém com as funcionalidades reduzidas. Na primeira o usuário pode visualizar somente as solicitações que o mesmo realizou, podendo cancelar a solicitação. Na segunda seção verá somente as salas que ele solicitou reserva e foi aprovada, podendo também cancelar. E na terceira, assim como o admin ele pode visualizar todas as salas cadastradas no sistema, podendo solicitar uma reserva.

Figura 11 – Tela 6

Fonte: Elaborada pelo autor

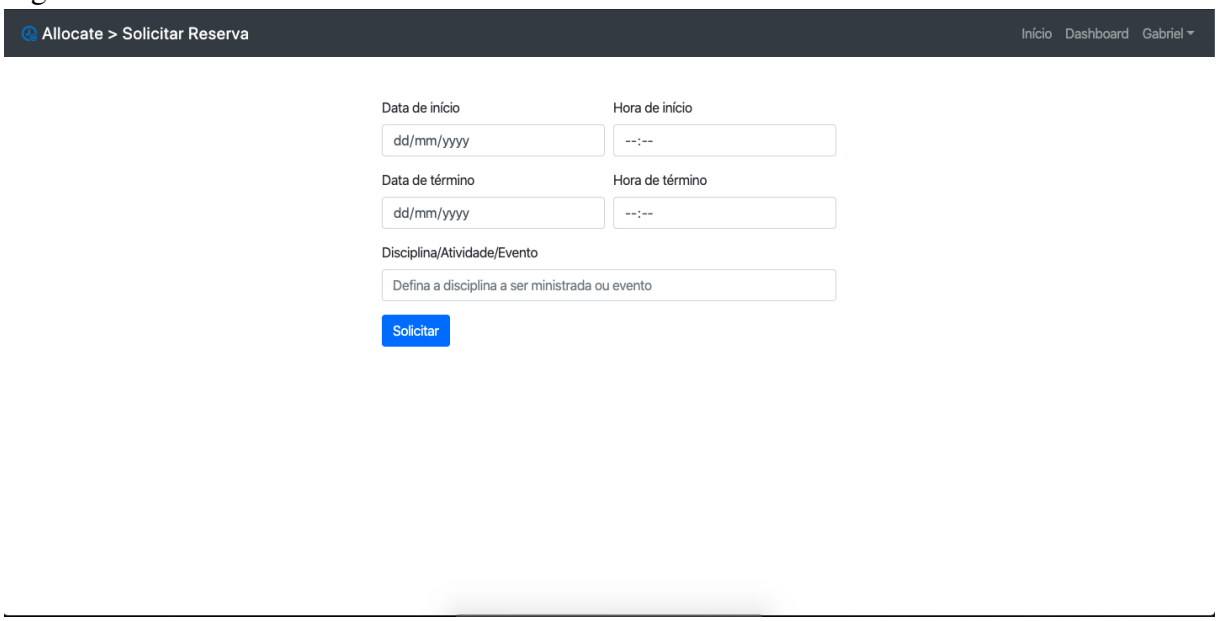
Figura 12 – Tela 7



Fonte: Elaborada pelo autor

Na Figura 13 é apresentada a tela de solicitação de reserva, sendo esta de acesso do professor/servidor ou aluno. Após realizar a solicitação, a mesma será enviada diretamente para o admin, onde o admin poderá aprovar caso seja possível fazer a reserva.

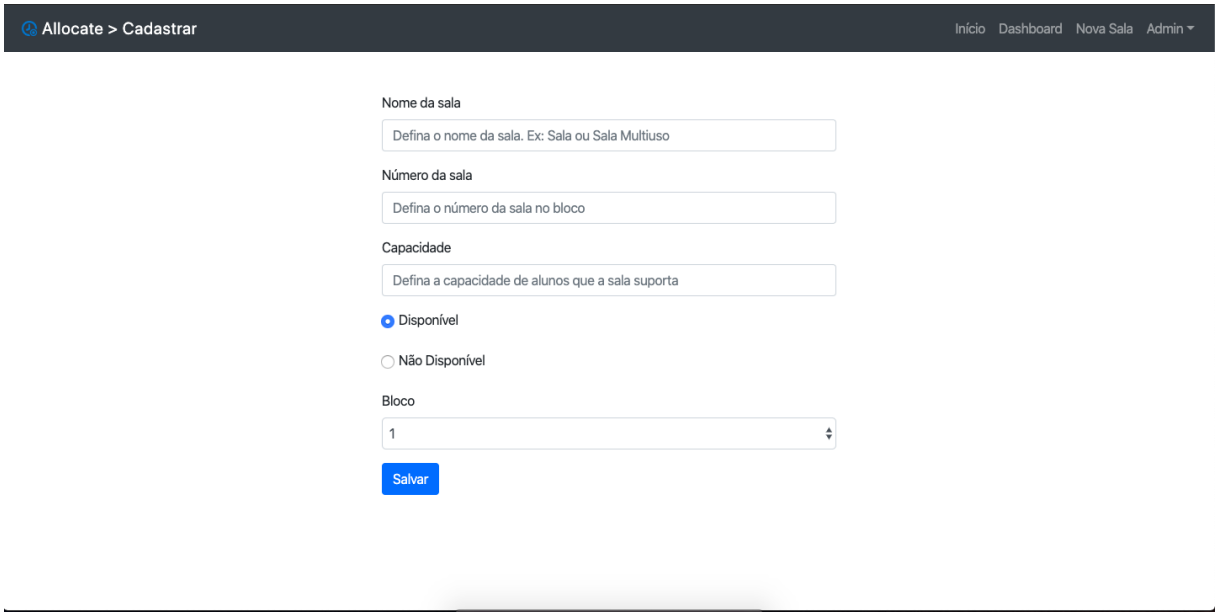
Figura 13 – Tela 8



Fonte: Elaborada pelo autor

Na Figura 14 é apresentada a tela para cadastrar uma nova sala, na qual somente o admin tem permissão. Possui campos para nome da sala, número da sala no bloco, capacidade de alunos que suporta, status de disponibilidade e bloco onde está localizada.

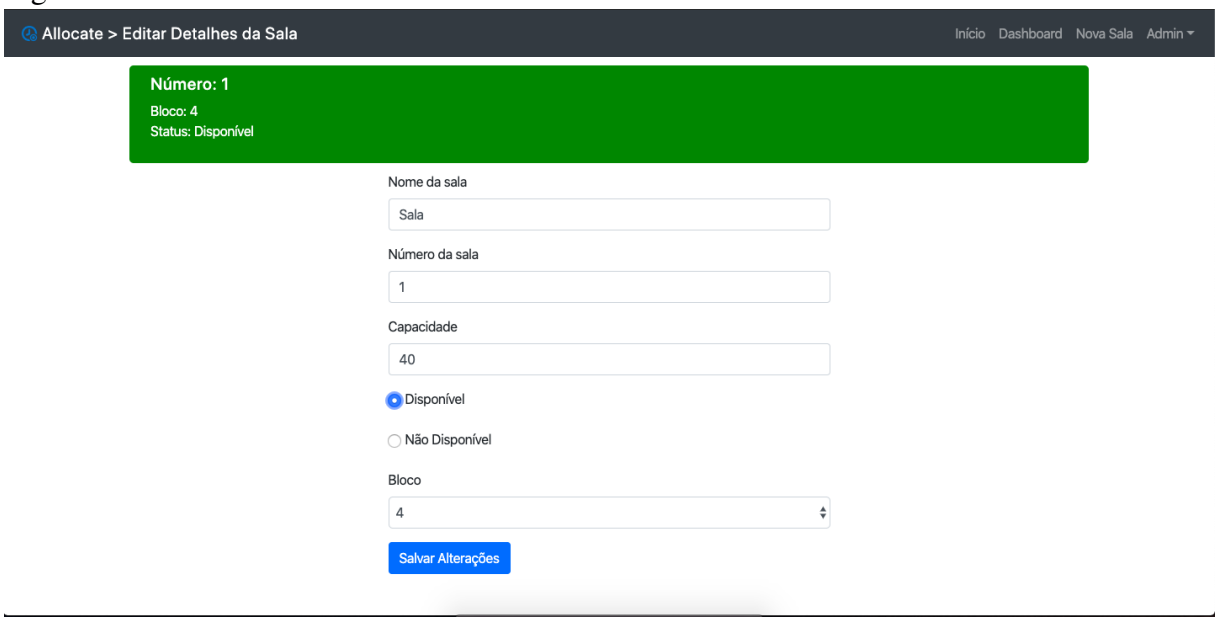
Figura 14 – Tela 9



Fonte: Elaborada pelo autor

Na Figura 15 é apresentada a tela de edição dos dados da sala, de acesso do admin. Onde é possível alterar dados como diponibilidade, bloco, capacidade, número e nome.

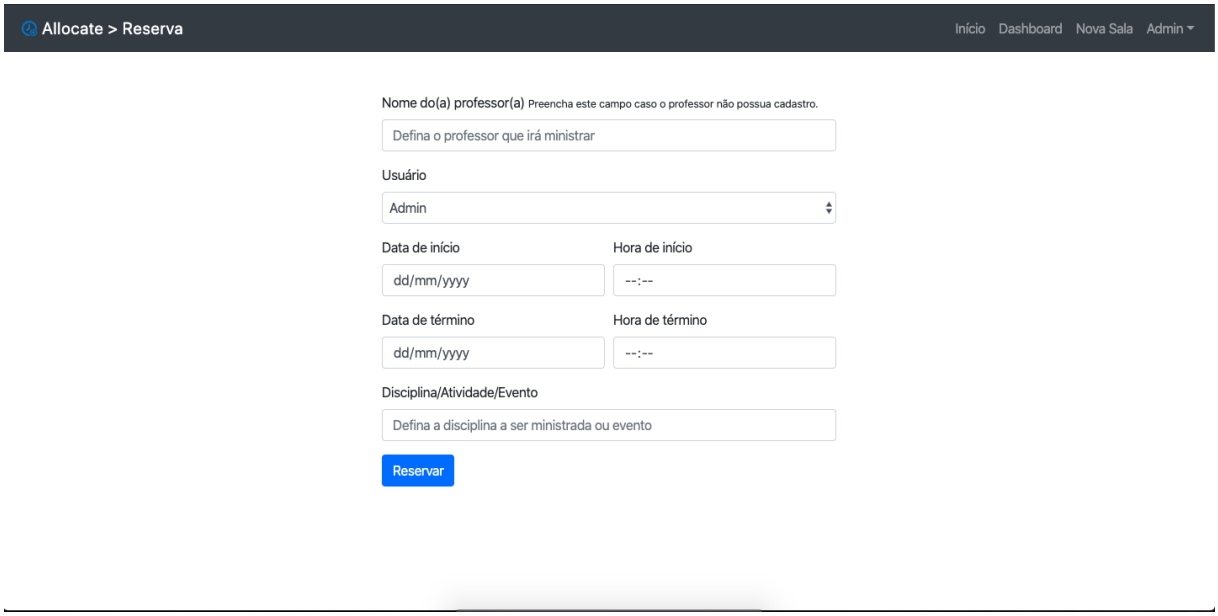
Figura 15 – Tela 10



Fonte: Elaborada pelo autor

Na Figura 16 é apresentada a tela de reserva, de acesso do admin, onde o mesmo pode fazer uma reserva diretamente. A reserva aparecerá diretamente na tela de reservas do admin e na tela inicial da aplicação.

Figura 16 – Tela 11

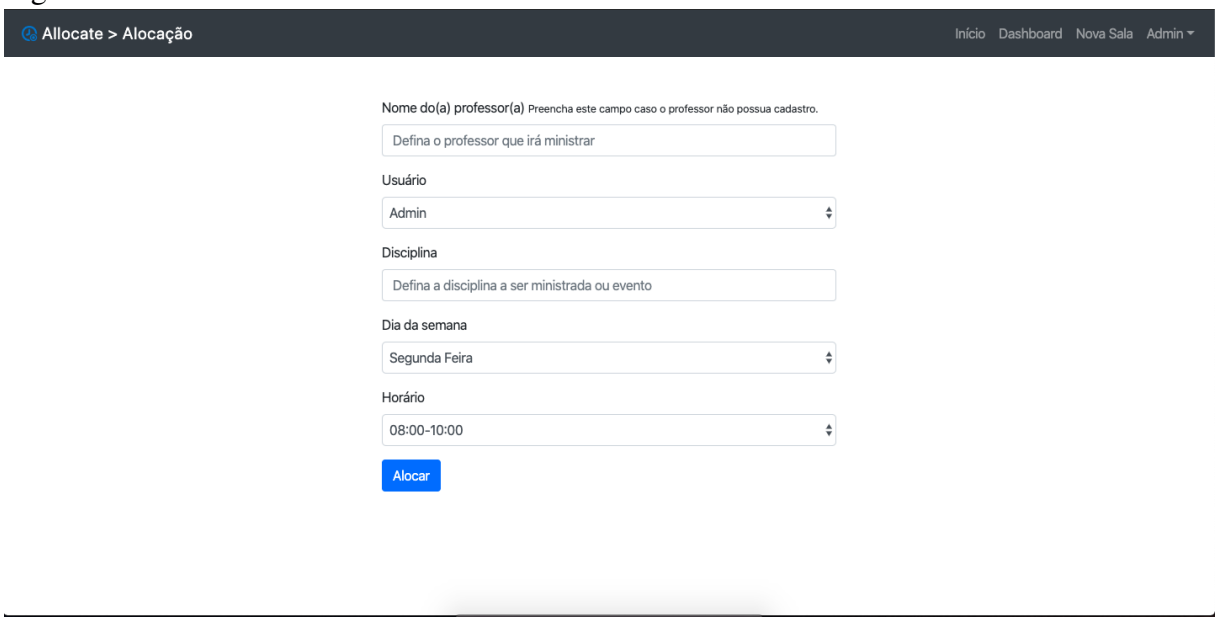


Fonte: Elaborada pelo autor

Na Figura 17 é apresentada a tela de alocação, semelhante à de reserva, porém com

alguns campos distintos. De acesso exclusivo do admin. Nesta tela o admin tem disponível os horários pre-definidos de acordo o da universidade, e possui pre-definidos os dias da semana, então alocações podem ser feitas somente de segunda a sexta.

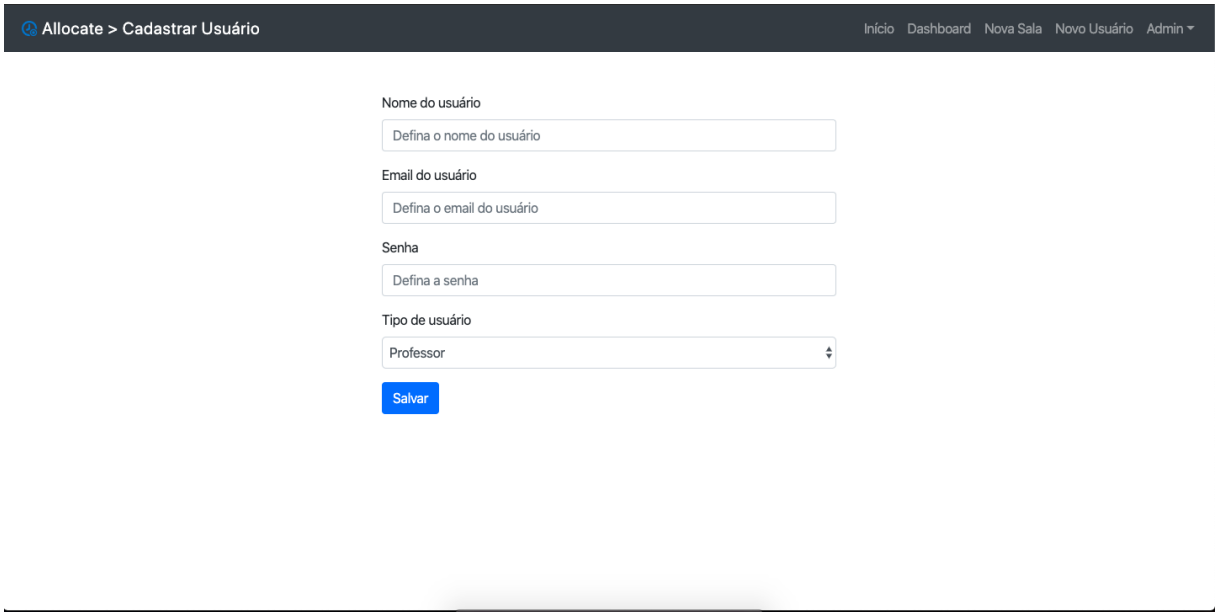
Figura 17 – Tela 12



Fonte: Elaborada pelo autor

Na figura 18 é apresentada a tela onde o admin poderá realizar o cadastro de um usuário, dos tipos professor ou aluno.

Figura 18 – Tela 13



Fonte: Elaborada pelo autor

5.3 Testes de verificação

Ao fim da implementação, ou seja, logo após a finalização da integração entre back-end e front-end foram realizados alguns testes com o servidor responsável pela atividade de reserva de salas.

5.3.1 Realizar login (admin e professor)

Foram fornecidos as credenciais de acesso de admin e de professor teste, para que fosse possível realizar os demais testes. O servidor conseguiu fazer login de forma rápida, cerca de 10 segundos, tanto para admin quanto para professor.

5.3.2 Como admin, cadastrar uma sala

Foi solicitado que o servidor fizesse um cadastro de uma sala, os dados ficaram a critério do mesmo, deixando-o livre para testar quaisquer entradas.

Houve um ponto em que o mesmo ficou em dúvida, que é no campo nome nesta tela, pois aparece nome e o mesmo tentou digitar o nome da sala já com o número. Porém abaixo do campo nome há o campo para número da sala. Isso foi explicado ao servidor e o mesmo entendeu como funciona.

5.3.3 Como admin, fazer uma reserva na sala

Assim que realizasse o cadastro o servidor fez uma reserva da sala que havia cadastrado.

5.3.4 Como admin, fazer uma alocação na sala

Foi solicitado ao servidor, realizar a alocação da sala que cadastrou para um horário de sua escolha.

5.3.5 Como admin, alterar os dados da sala

Foi solicitado ao servidor, fazer a alteração dos dados da sala que havia cadastrado.

5.3.6 Como admin, cancelar uma reserva

Por ser um processo simples, o servidor precisa apenas de clicar em um botão para cancelar uma reserva que havia feito. Relatou que não encontrou dificuldades, pois o significado da opção ficou claro para o mesmo.

5.3.7 Como admin, cancelar uma alocação

É uma atividade bem semelhante à de cancelar uma reserva, o relatou que não encontrou dificuldades para executar a ação.

5.3.8 Como professor, solicitar uma reserva

Após realizar login como professor, o servidor foi instruido a fazer uma solicitação de reserva como professor para verificar como funciona esse processo de solicitar/aprovar/cancelar.

5.3.9 Como admin, aprovar uma reserva feita por um professor

Foi solicitado ao servidor, aprovar uma reserva feita por um professor, no caso na tarefa anterior, onde fez uma solicitação.

5.3.10 Considerações

O servidor relatou que ainda sentiria um pouco de dificuldade para se adaptar à nova forma de realizar reservas, mas que isso se deve pelo fato de já está acostumado com a planilha.

5.3.11 Perguntas realizadas após os testes

- O que você achou da tela de login? O mesmo relatou que a tela é bem intuitiva e simples, tendo apenas os dados necessários, e exclamou que login é bastante rápido, cerca de 10s.
- O que você achou da tela inicial? Relatou que está bem dividida pois mostra as alocações e reservas descritas.
- O que você achou da tela de dashboard? O mesmo relatou que conseguiu aprovar normalmente e já visualizar a reserva que foi cadastrada. Não encontrou dificuldades.

- O que você achou da tela de solicitação de reserva? Relatou que não encontrou dificuldades ao realizar a tarefa e que a tela tem seu propósito bem claro, pois apresenta somente os campos necessários.
- O que você achou da tela de reserva? O mesmo relatou que achou a tela bem intuitiva e que a reserva é bastante rápida, a operação leva cerca de 1s para concluir.
- O que você achou da tela de alocação? Por ser um processo semelhante ao de reserva, o mesmo não encontrou dificuldades para finalizar. Relatou que a interface está bem intuitiva.
- O que você achou da tela cadastrar sala? Relatou que o cadastro é bem intuitivo e rápido para ser feito, cerca de 30s.
- O que você achou da tela de editar detalhes da sala? Relatou que a tela é bem intuitiva, mostra os campos preenchidos com os dados atuais da sala. O mesmo não encontrou dificuldades para realizar a tarefa.
- O que você achou da tela de cadastrar usuário? Fácil de usar e simples no que deve ser preenchido.

6 CONCLUSÕES E TRABALHOS FUTUROS

O processo de pesquisa e desenvolvimento de um software que atenda uma necessidade de um usuário, tornando-o capaz de realizar suas atividades de forma digital requer um grande esforço para se obtenha um software que atenda minimamente as expectativas do usuário.

O projeto desse tipo requer uma boa base de conhecimentos de tecnologias front-end e back-end para se possa desenvolver as duas aplicações separadas, uma aplicação front-end e uma aplicação back-end.

Como todo software, sempre há evoluções a serem consideradas e realizadas posteriormente. Evoluções estas que podem não estar no escopo do desenvolvimento deste trabalho, mas que deixariam o software melhor.

Durante os testes, o servidor sugeriu algumas coisas que poderiam ser adicionadas na aplicação para torná-la melhor, segue abaixo as considerações do mesmo.

- No dashboard do admin, possibilitar o fitro das salas por bloco e por dia.
- Na tela de reserva/solicitação, bloquear a seleção de uma data anterior a do dia em questão.
- Enviar um email para o professor assim que a solicitação de reserva for aceita.
- Possibilitar adicionar mais informações da sala como, se tem ar, projetor, etc.

Esses pontos destacados poderam ser realizados futuramente para a evolução deste trabalho.

REFERÊNCIAS

ALMEIDA, R. **Sistema de reserva e acompanhamento de salas**. [*S. l.*]: - UFC, 2016. Disponível em: http://www.quixada.ufc.br/wp-content/uploads/2015/06/Encontros-Universitários-2016_Quixadá.pdf. Acesso em: 01 dez. 2019.

BUNA, S. **Yes, React is taking over front-end development**. [*S. l.*]: - Medium, 2017. Disponível em: https://medium.freecodecamp.org/yes-react-is-taking-over-front-end-development-the-question-is-why-40837af8ab76. Acesso em: 01 dez. 2019.

ECMA-INTERNATIONAL. **ECMAScript 2015 Language Specification**. [*S. l.*]: - ECMAScript, 2015. Disponível em: https://imasters.com.br/desenvolvimento/o-guia-completo-do-react-e-o-seu-ecossistema. Acesso em: 01 dez. 2019.

FACEBOOK. **Arquitetura Flux**. [*S. l.*]: - Facebook, 2014. Disponível em: https://github.com/facebook/flux. Acesso em: 06 maio. 2019.

FACEBOOK-INC. **In-Depth Overview**. [*S. l.*]: - Flux, 2014. Disponível em: https://facebook.github.io/flux/docs/in-depth-overview. Acesso em: 01 dez. 2019.

FERNANDES, D. **NodeJS: Vale a pena?** [*S. l.*]: - RocketSeat, 2017. Disponível em: https://blog.rocketseat.com.br/nodejs-vale-a-pena-vantagens/. Acesso em: 04 dez. 2019.

FERNANDES, D. **AdonisJS vs ExpressJS: Quando utilizar cada um?** [*S. l.*]: - RocketSeat, 2018. Disponível em: https://blog.rocketseat.com.br/adonis-vs-express/. Acesso em: 04 dez. 2019.

GOEL, A. **10 Best JavaScript Frameworks to Use in 2019**. [*S. l.*]: - Hack.io, 2019. Disponível em: https://hackr.io/blog/10-best-javascript-frameworks-2019. Acesso em: 02 dez. 2019.

KHUAT, T. **Developing a frontend application using ReactJS and Redux**. Dissertação (Degree Programme in Business Information Technology Bachelor's) — Laurea University of Applied Sciences, Leppävaara, 2018. Disponível em: https://www.theseus.fi/bitstream/handle/10024/150837/Tung_Khuat_1301747_Thesis.pdf?sequence=1. Acesso em: 01 dez. 2019.

LIMA, M. **O guia completo do React e o seu ecossistema**. [*S. l.*]: - Imasters, 2017. Disponível em: https://imasters.com.br/desenvolvimento/o-guia-completo-do-react-e-o-seu-ecossistema. Acesso em: 03 dez. 2019.

REACTJS.ORG. **Introduzindo JSX**. [*S. l.*]: - ReactJS, 2014. Disponível em: https://pt-br.reactjs.org/docs/introducing-jsx.html. Acesso em: 01 dez. 2019.

RODRIGUEZ, A. **Serviços da web RESTful: informações básicas**. [*S. l.*]: - IBM, 2015. Disponível em: https://www.ibm.com/developerworks/br/library/ws-restful/index.html. Acesso em: 01 dez. 2019.

SILVA, J. **Alocação de disciplinas maximizando opções de matrícula**. [*S. l.*]: - UFC, 2016. Disponível em: http://www.quixada.ufc.br/wp-content/uploads/2015/06/Encontros-Universitários-2016_Quixadá.pdf. Acesso em: 02 dez. 2019.

VIANA, B. **SIGAT (Sistema de gestão e agendamento de TCCs)**. [*S. l.*]: - UFC, 2017. Disponível em: https://www.quixada.ufc.br/wp-content/uploads/2015/06/Encontros-Universitários-2017.pdf. Acesso em: 01 dez. 2019.