CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA
FUNDAÇÃO DE ENSINO "EURÍPIDES SOARES DA ROCHA"
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Ferramenta Didática WEB Colaborativa para Criação e Simulação de Circuitos Lógicos Digitais

FERNANDO HENRIQUE ALVES FRANCO

Marília, 2016

CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA
FUNDAÇÃO DE ENSINO "EURÍPIDES SOARES DA ROCHA"
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Ferramenta Didática WEB Colaborativa para Criação e Simulação de Circuitos Lógicos Digitais

Monografia apresentada ao Centro Universitário Eurípedes de Marília como parte dos requisitos necessários para obtenção do título de Bacharel em Ciência da Computação
Orientador: Prof. Me. Rodolfo Barros Chiaramonte

Marília, 2016

CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA - UNIVEM
MANTIDO PELA FUNDAÇÃO DE ENSINO "EURÍPIDES SOARES DA ROCHA"

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Fernando Henrique Alves Franco

Ferramenta WEB didática colaborativa para criação e simulação de circuitos lógicos digitais.

Banca examinadora da monografia apresentada ao Curso de Bacharelado em Ciência da Computação do UNIVEM/F.E.E.S.R., para obtenção do Título de Bacharel em Ciência da Computação.

Nota: 10,0 (Dez)

Orientador:Rodolfo Barros Chiaramonte

1º.Examinador:Fábio Dacêncio Pereira

2º.Examinador:Fabio Piola Navarro

Marília, 07 de dezembro de 2016.

AGRADECIMENTOS

A minha mãe Maria, a minha tia Carmen, a minha namorada Suzielly.

A meus amigos, em especial ao Bruno que sempre esteve presente nos piores momentos para me acolher.

Aos meus colegas de sala Rafael, Cristiano, César, Claudio, Vitória, Lucas, Melissa e Anderson.

Ao meu orientador Prof. Me. Rodolfo Barros Chiaramonte, pela amizade, auxílio e orientação durante a produção do trabalho, e principalmente por acreditar neste trabalho até o último minuto apesar dos problemas ocorridos.

A todos os professores e profissionais do Centro Universitário Eurípedes de Marília que de alguma forma contribuíram com minha graduação.

RESUMO

A Ferramenta Didática Colaborativa para Criação e Simulação de Circuitos Lógicos Digitais é uma aplicação WEB, onde o usuário pode criar, editar e simular projetos de circuitos lógicos digitais utilizando de bibliotecas de componentes, como portas lógicas digitais, junto com ferramentas para realizar ligações de fios, inputs, outputs, entre outras. Tendo o foco principal para a utilização entre multiusuários, onde a edição do projeto acontece de forma simultânea para todos os envolvidos conectados no momento, assim alunos e professores possam criar, editar e visualizar resultados de forma síncrona e dinâmica, auxiliando no aprendizado e na realização de projetos em grupos.

Palavras-chave: Ferramenta Didática, Circuitos Lógicos Digitais, Colaborativa, Multiusuário.

ABSTRACT

The Collaborative Didactic Tool for Creating and Simulating Digital Logic Circuits is a WEB application where the user can create, edit and simulate digital logic circuits projects using component libraries, such as digital logic gates, along with tools to make wire connections, inputs, outputs, and others. Having the focus for use among multiusers, where project editing happens simultaneously for all those involved now, so that students and teachers can create, edit and visualize results in a synchronous and dynamic way, aiding in learning and performing of projects in groups.

Key words: Didactic Tool, Digital Logic Circuits, Collaborative, Multiuser.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

Sumário

Introdução

Atualmente em cursos como Ciência da Computação, Sistemas de Informação, entre outros, existem matérias que envolvem a aprendizagem de Circuitos Lógicos Digitais, para que então o aluno possa assimilar o conhecimento primordial da computação e seus componentes.

Nestas matérias, após a apresentação de seus fundamentos, aplicações e teorias básicas, são realizadas aulas práticas para um melhor entendimento do aluno sobre o conteúdo estudado e obtido até o momento. Para isso, existe a necessidade de uso de ferramentas que possam construir e simular circuitos lógicos digitais, que são apresentadas aos alunos por seus tutores, onde receberão aulas sobre seu funcionamento, tanto para a construção dos circuitos desejados, quanto para a simulação de seus projetos construídos.

Porém, por muitas vezes, tais ferramentas são privadas, o que leva as entidades educacionais, e/ou ao aluno, o custo financeiro para compra e uso da ferramenta. Outro problema visível neste meio é que, geralmente, essas ferramentas são utilizadas no mercado de trabalho, ou seja, as ferramentas são preparadas para um único usuário final, onde já deve existir um conhecimento prévio e/ou treinamentos para o uso da ferramenta. Isso pode acarretar na falta de recursos que auxiliem aos alunos no aprendizado e no uso da aplicação para eles apresentada, como o uso da ferramenta de forma colaborativa entre multiusuários que poderia auxiliar grupos de alunos interagir com o mesmo projeto de forma colaborativa e simultânea.

Deste modo, propõe-se uma ferramenta para a Criação e Edição de Circuitos Lógicos Digitais sendo colaborativa entre multiusuários, sem a necessidade de softwares de terceiros, para que o mesmo possa ser executado em qualquer navegador moderno como Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, entre outros, sem a necessidade de instalação de máquinas virtuais ou extensões extras ou externas.

Objetivos

Este trabalho propõe uma ferramenta didática para criação e simulação de circuitos lógicos digitais, sendo uma aplicação WEB voltada para a colaboração entre multiusuários.

Tem como objetivos específicos:

- Definição das tecnologias a serem utilizadas do lado cliente e servidor da aplicação.
- Definição da arquitetura para a comunicação entre multiusuários.
- Implementação da ferramenta para criação de simulação de circuitos lógicos digitais.
- Implementação da arquitetura utilizando tecnologia definida para comunicação entre cliente/cliente.
- Testes de avaliação da comunicação entre usuários utilizando o módulo criado.

Metodologia

O trabalho está dividido em cinco partes principais:

- Levantamento bibliográfico e pesquisa de trabalhos correlatos e tecnologias:
    - Foi realizada a busca bibliográfica sobre temas como: ferramenta didática, web socket, criação e simulação de circuitos lógicos digitais, entre outros.
- Definição das tecnologias a serem utilizadas do lado cliente e servidor da aplicação.
    - Estudo de possíveis tecnologias.
    - Definição das tecnologias que melhor se adequarem para a situação.
- Definição da arquitetura para a comunicação entre multiusuários.
    - Estudo de possíveis arquiteturas.
    - Definição da arquitetura que melhor se adequa para a situação.
- Implementação da ferramenta para criação e simulação de circuitos lógicos digitais.
    - Desenvolvimento da ferramenta utilizando *"Javascript5"*, *"AngularJS"* e *"CSS3*
    - Desenvolvimento do workspace utilizando *"Canvas"*.
- Implementação da arquitetura utilizando tecnologia de websocket e linguagem javascript.
    - Desenvolvimento da arquitetura utilizando *"Node.js®"*.
    - Implementação da comunicação utilizando *"socket.io"*.
        - Comunicação de edição em projeto.
- Testes de avaliação da comunicação entre usuários utilizando o módulo criado.
    - Teste de edição de componentes no projeto de circuitos lógicos digitais.

Trabalhos Correlatos

Considerando as dificuldades apresentadas, (KAYAL; STEFANOVIC; PASTRE, 2004) propôs uma ferramenta educacional independente que é dedicada a compreensão do comportamento de transistor MOS, para o ponto de vista eletrônico, com ênfase em células analógicas básicas. Apesar de ser baseada para fins educacionais, ela é voltada para circuitos analógicos, enquanto este trabalho tem como foco, Circuitos Lógicos Digitais, que "é aquele em que estão presentes somente dois valores lógicos" (TANENBAUM, 2008).

(KIM et al., 2012) também propôs uma ferramenta educacional, sendo uma aplicação completa para desenvolvimento e simulação de Circuitos Lógicos Digitais, com recursos como simulação de laboratório para componentes elétricos e ainda sendo uma ferramenta WEB. Porém o trabalho citado utiliza de Java Applet, que pode consumir mais recursos do navegador, pela necessidade de uma máquina virtual Java e alguns navegadores hoje já não dão mais suporte ao Java Applet, e não existe um meio para uma comunicação colaborativa da ferramenta entre multiusuários.

Outra ferramenta também dependente de plug-in externo oferecido por terceiros para serem instalados em navegadores é a oferecida por (LUCENA, 2013), onde foi utilizada a linguagem Action Script 3.0 que depende do plug-in para Adobe Flash Player nos navegadores.

Logisim é uma ferramenta com a finalidade de facilitar a aprendizagem dos conceitos básicos de circuitos lógicos digitais. Nela são possíveis a criação e a simulação de circuitos, porem sendo a simulação de forma visual pelo próprio circuito criado pela coloração de seus fios e numerais em suas entradas e saídas. Apesar de ser desenvolvida em Java, e com isso ter a compatibilidade entre plataformas diferentes, o usuário ainda se prende a softwares de terceiros para ser executada. Também não é possível uma edição colaborativa entre multiusuários.

Assim como Logisim, o LogicCircuit oferece ferramentas para criação e simulação de circuitos lógicos digitais, porem com um Osciloscópio, onde o usuário pode realizar suas simulações verificando suas entradas e saídas de modo a observar as bordas de subida e descida de cada componente do circuito criado. LogicCircuit também depende de um software instalado na máquina do usuário, porém neste caso sendo o.NET Framework 4.5 ou superior, e ainda sem uma função para comunicação entre multiusuários para a criação dos circuitos desejados.

Uma ferramenta mais completa, com possibilidades de criação de macros, simulação com pontos de prova ou ainda visualizando as bordas de subida e descida, assim

como ferramentas avançadas para criação dos circuitos, é o schematic editor, que é um editor interno do project manager oferecido pela Xilinx. Porém, apesar de seus prós, é uma ferramenta totalmente dependente do Sistema Operacional, onde o mesmo deve ser Windows, e ainda não oferece aos usuários uma plataforma colaborativa para a dição de projetos.

Ferramentas para criação de diagramas como draw.io, oferecem aos usuários, uma forma para criar diagramas de circuitos lógicos digitais, ainda com a plataforma para colaboração desta criação, e ainda sendo disponibilizada via WEB, ou seja, sendo uma aplicação WEB e independente de plug-ins pois foi desenvolvida com Canvas do HTML5. Porém a ferramenta não possui de recursos para simulação do circuito montado

Outra ferramenta WEB é a logic.ly, que assim como draw.io, existe a possibilidade da criação de circuitos, e neste caso também a possibilidade da simulação do circuito criado. Infelizmente esta logic.ly não tem a plataforma de colaboração como draw.io, e ainda foi desenvolvida com base no Adobe Flash Player, o que a deixa totalmente dependente de softwares e plug-ins de terceiros para que a mesma possa ser executada em um navegador.

Uma ferramenta WEB em destaque é a simulator.io, que oferece ao usuário a possibilidade de criação e edição de circuitos lógicos digitais, podendo compartilha-los de forma colaborativa, e desenvolvido com Canvas do HTML5 que a deixa sem dependências de softwares de terceiros. Simulator.io também oferece simulação da ferramenta, porém sem visualização de bordas de subida e descida, e ainda não conta com ferramentas para importação de bibliotecas externas para uso de novas componentes durante a criação de um circuito.

Para realizar uma comparação entre as ferramentas citadas acima, pode-se observar a tabela 1, onde foi exposto os principais recursos, tanto para a criação, quanto para a simulação de circuitos lógicos digitais. Também foi levantado informações sobre a ferramenta ser ou não utilizada via WEB, assim como também o uso de software de terceiros para que a mesma possa ser executada.

Tabela 1 - Comparação de trabalhos correlatos

Trabalho	Ferramenta educacional	Criação de circuitos lógicos digitais	Simulação de circuitos lógicos digitais	Visualização de sinais transmitidos por coloração em fios	Visualização de bordas de subida e descida das entradas/saídas	Aplicação WEB	Dados e/ou projetos persistidos na nuvem	Dependência de softwares e/ou plug-ins de terceiros	Utilização de bibliotecas para gerenciar componentes	Ferramenta colaborativa	Criação de macros/componentes para utilização posterior
Trabalho	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Não
Kayal	Sim	Não	Não	Sim	Não	Não	Não	Sim	Não	Não	Não
Kim	Sim	Sim	Sim	Sim	Sim	Sim	Não	Sim	Não	Não	Sim
Lucena	Sim	Sim	Sim	Sim	Sim	Não	Não	Sim	Não	Não	Sim
Logisim	Sim	Sim	Sim	Sim	Não	Não	Não	Sim	Não	Não	Sim
LogicCircuit	Sim	Sim	Sim	Sim	Não	Não	Não	Sim	Não	Não	Sim
Xilinx	Não	Sim	Sim	Sim	Sim	Não	Não	Sim	Sim	Não	Sim
Draw.io	Não	Sim	Não	Não	Não	Sim	Sim	Não	Sim	Sim	Não
Logic.ly	Não	Sim	Sim	Sim	Não	Sim	Sim	Sim	Não	Não	Não
Simulator.io	Não	Sim	Sim	Sim	Não	Sim	Sim	Não	Não	Sim	Não

1. Fundamentação Teórica

Para a criação da ferramenta proposta neste trabalho, é necessário o conhecimento sobre os principais temas nela abrangidos, sendo um deles os próprios circuitos lógicos digitais e como eles se comportam, e outro sobre as ferramentas didáticas e softwares educacionais. Assim contextualizando a teoria utilizada para o desenvolvimento deste trabalho e enfatizando fatos importantes para que o mesmo pudesse ser criado.

1.1.Nível Lógico Digital

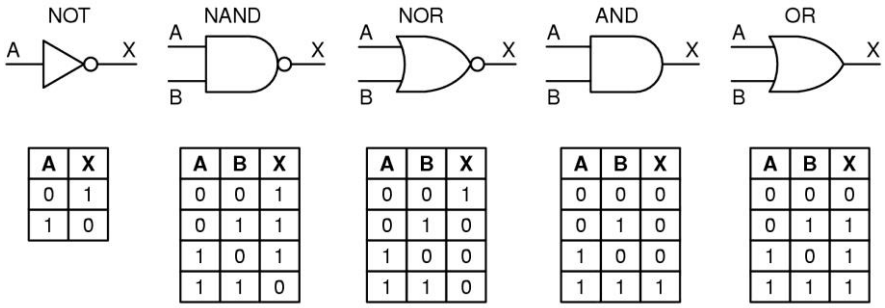
Nível Lógico Digital é o nível mais baixo da arquitetura, onde observa-se o fato de que circuitos digitais utilizam de sinais elétricos para admitir a presença de dois valores, utilizando da voltagem de cada sinal para representar os valores um e zero para definir a representação de valores binários. “O normal é que um sinal entre 0 e 1 volt represente um valor (por exemplo, 0 binário) e um sinal entre 2 e 5 volts represente o outro valor (por exemplo, 1 binário)” (TANENBAUM, 2008)

1.1.1. Portas Lógicas

Portas lógicas são minúsculos dispositivos eletrônicos que podem calcular funções utilizando de valores binários, e são a base de todo o hardware de um computador digital.

Baseando-se no fato de que todo transistor pode funcionar como um comutador binário, os mesmos podem ser utilizados junto a três ou mais conexões como coletores, base e emissores para realizarem comunicações necessárias com o mundo exterior, recebendo e/ou transmitindo sinais elétricos. Utilizando destas conexões podemos então representar as “três portas mais simples que são denominadas portas NOT, NAND e NOR” (TANENBAUM, 2008), e posteriormente com a utilização da combinação de duas portas, pode-se então gerar as portas AND (utilizando NAND e NOT) e OR (utilizando NOR e NOT), assim obtendo os cinco principais elementos de construção do nível lógico digital exibidos com suas anotações na figura 1 onde é adicionado nas portas NAND e NOR uma bolha de inversão, que também pode ser utilizada em outros contextos indicando um sinal invertido.

Figura 1 - Os símbolos e o comportamento funcional das 5 portas lógicas básicas



A	X
0	1
1	0

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

Fonte: (TANENBAUM, 2008)

Observa-se então que as portas AND e OR são a junção de duas portas lógicas mais simples, logo, essas portas consomem mais recursos que as portas NAND e NOR, pois cada porta utiliza então de 3 transistores, sendo 2 de cada porta simples NAND e NOR e 1 transistor da porta NOT para a inversão de seu valor. Tal consumo de recurso influencia diretamente na construção de computadores digitais, onde muitos computadores acabam sendo baseados nas portas simples NAND e NOR invés de utilizar as portas AND e OR.

(TANENBAUM, 2008) também indica a importancia de citar que cada porta pode receber mais de duas entradas como observado até o momento, porem na prática não é algo comum encontrar portas lógicas com mais de 8 entradas.

1.1.2. Álgebra Booleana

Álgebra Booleana ou Álgebra de Boole é um "sistema matemático composto por operadores, regras, postulados e teoremas" (LUCENA, 2013) na base 2. Ainda pode-se comprar a álgebra booleana com a álgebra ordinária, pois assim como em uma a outra também possui funções. "Uma função booleana tem uma ou mais variáveis de entrada e produz um resultado que depende somente dos valores dessas variáveis" (TANENBAUM, 2008). Como exemplo de função para a porta lógica NOT, se obtêm $f(A) = 1$ se A for 0 e $f(A) = 0$ se $A = 1$.

Como uma função booleana teremos sempre n entradas com máximo de 2^n combinações possíveis dos valores de entrada, logo uma função pode ser descrita através de uma tabela com 2^n linhas e $n + 1$ colunas, sendo a última coluna o valor de saída da função. Todas as tabelas apresentadas na figura 1 são exemplos que mostram as entradas e os resultados para cada porta lógica digital indicada anteriormente, e tais tabelas podem ser chamadas de tabela verdade.

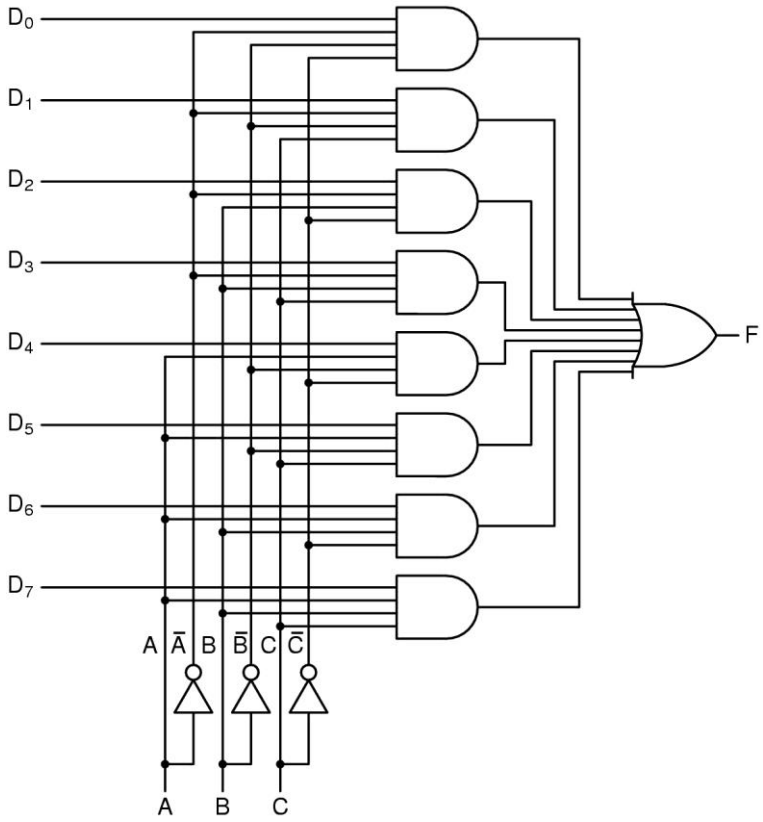
1.1.3. Circuitos Lógicos Digitais Combinacionais

Um circuito combinacional é um circuito formado por combinações de portas lógicas, utilizando de fios junto suas entradas e saídas para interligar cada porta e assim criar funções complexas para obter um resultado elaborado para cada combinação utilizada nas entradas principais do circuito. “Um circuito que esteja implementando uma tabela-verdade é um exemplo típico de um circuito combinacional” (TANENBAUM, 2008).

Quando a tensão da entrada de uma porta de entrada do circuito é alterada, sua saída também será alterada, passando pela ligação e entrando na entrada de uma porta ligada a esta anterior, alterando seu valor e propagando seu novo resultado para sua saída e assim por diante.

Exemplos comuns de circuitos combinacionais são os multiplexadores, decodificadores, comparadores, arranjos lógicos programáveis entre outros. Na figura 2 mostra-se o exemplo de um multiplexador utilizando das combinações de várias portas lógicas para a criação de um circuito lógico combinacional.

Figura 2 - Multiplexador



Fonte: (TANENBAUM, 2008)

Contudo, existem alguns detalhes que devem ser observados. Um deles é o tempo de propagação, necessário para que um circuito possa efetivar sua saída, assim garantindo o resultado do mesmo, pois um sinal emitido para uma entrada de um circuito deverá ser propagado até sua saída, consumindo uma fração de nano segundos passando de porta em porta. Esse procedimento deve ser levado em consideração em projetos de circuitos, pois em casos onde um circuito é composto por vários circuitos ligados entre si, e que a necessidade de uma resposta que dependa a propagação de um sinal por todos eles, tal saída pode então demorar um certo tempo (tempo de propagação) para ser atualizada definitivamente, ou ainda, apresentar resultados intermediários antes de um resultado efetivo.

Outro ponto a se observar é o limite de tensão que um circuito pode receber ou emitir, pois seus circuitos internos possuem limites operacionais de condução de corrente, que em caso de serem ultrapassados, pode acarretar em danos ao circuito. Logo uma entrada que possui uma especificação de tensão para manter seu estado binário, seu circuito deve ter a capacidade de transmitir também uma tensão apropriada para sua (s) saída (s), para consequentemente ser capaz de alimentar uma ligação com um novo circuito.

Contudo, este projeto consiste na criação e simulação simplificada dos circuitos lógicos criados, onde apesar de se levar em consideração a utilização da tensão representada por emissão de um sinal 0 ou 1 sendo um valor booleana true ou false, disparado pelas entradas do circuito criado, e com tempo de propagação sendo contabilizado apenas por seus componentes e não por seus fios.

Tais cuidados possibilitam a criação de circuitos não apenas combinacionais, onde a saída de cada porta lógica, componente ou circuito criado, dependeriam apenas de suas variáveis de entrada, o que diminuiria a capacidade do projeto de satisfazer a necessidade educacional exigida por um software como o aqui proposto. Porem a não contabilização da propagação de um sinal por um fio, caracteriza em uma simulação mais distante de um ambiente real.

1.2.Ferramentas Didáticas

Softwares educacionais segundo (MORAIS, 2003), diferem em duas classes para o processo educacional, sendo eles software educativo e software aplicativo. Este documento se focará apenas em software educativo, que tem por objetivo principal, facilitar o processo de aprendizado do aluno, fazendo-o com que ele construa determinado conhecimento baseando-se no conteúdo didático que a ferramenta está disponível a oferecer.

1.2.1. Softwares Educativos

Softwares educativos possuem algumas características que os diferenciam de demais softwares relacionais à educação, sendo elas de acordo com (MORAIS, 2003):

Definição e presença de uma fundamentação pedagógica que permeie todo o seu desenvolvimento;

Finalidade didática, por levar o aluno a "construir" conhecimento relacionado com seu currículo escolar;

Interação de uso, uma vez que não se devem exigir do aluno conhecimentos computacionais prévios, mas permitir que qualquer aluno, mesmo que em um primeiro contato com a máquina, seja capaz de desenvolver suas atividades;

Atualização quanto ao estado da arte, ou seja, o uso de novas técnicas para o trabalho com imagens e sons cativando cada vez mais o interesse do aluno pelo software.

Tais softwares também podem ser subdivididos em 3 categorias se levado em consideração suas características específicas como SE com ênfase na lógica do conteúdo, onde o aluno é limitado com resposta pré-programadas e caso o mesmo saiba todas as respostas, nada do conteúdo fornecido será adquirido como um novo conhecimento. Outra categoria seriam SE inteligentes, que de certa forma, utilizaria da capacidade de aprendizado de novas informações para realizar uma iteração progressiva com o usuário, porem esse tipo de software deveria utilizar de Inteligência Artificial (IA), para que o mesmo tenha a capacidade necessária do aprendizado e manter o relacionamento necessário com o aluno para esse tipo de aprendizagem, o que não é o foco deste documento. Por fim uma terceira subcategoria também é citada por (MORAIS, 2003), onde se incluem softwares com perspectiva construtivista, na forma de simulações, desafios e jogos, onde será focado para este trabalho os softwares para simulações.

1.2.2. Software Educativo utilizando de Simulações

Com softwares de simulação, um aluno que tenha recebido o estudo inicial contendo a teoria do problema a ser solucionado, o mesmo terá a possibilidade de fazer simulações utilizando de várias entradas para um determinado problema, e consequentemente obtendo as soluções para esses valores em tempo real, que fará com que o aluno perceba o que cada ação realizada pode ou não influenciar no resultado do problema em questão. Como consequência a esta ação e reação do aluno sobre o software, o mesmo irá obter então o

conhecimento sobre o conteúdo estudado utilizando da ferramenta, assimilando o que foi explicado na teoria com o que foi realizado na prática.

A motivação dos usuários/alunos em aprender utilizando-se deste tipo de ferramentas é muito positiva, pois o aluno deixa de ser ocioso na construção do conhecimento e entende como se procede para solucionar um problema do mesmo tipo do conteúdo abordado pelo software (MORAIS, 2003).

Sendo então o tipo de software que corresponde a este trabalho, seguinte de simulações da criação de circuitos lógicos combinacionais e utilizando de ferramentas para testes do circuito criado em relações as várias possibilidades de entradas para que o aluno possa então entender o circuito montado e identificar possíveis erros e/ou problemas que podem ocorrer no mesmo, assim como também poderá verificar as alterações nos vários estágios durante uma simulação, identificando seu verdadeiro processo para obter um resultado final.

2. Tecnologias Empregadas

Sendo este um projeto WEB, e com a intensão de que o mesmo possa ser utilizado por qualquer navegador moderno, e ainda, sem a necessidade de instalações de softwares de terceiros para a execução da mesma, como por exemplo *"Java Applet Plugin"* (Java) e *"Adobe Flash Player"* (Action Script). Tais softwares são plug-ins para rodar linguagens de programação que geralmente não são padrões dos navegadores, e, portanto, correm o risco da perda de compatibilidade ou continuidade da mesma, ou ainda ser dependente de versões de softwares instalados no cliente que utilizará a aplicação final para criação de seus projetos.

Portanto, para que a ferramenta proposta seja compatível com os navegadores mais recentes, e diminuindo o risco de incompatibilidade com os mesmos, a ferramenta será desenvolvida com *HTML5*, *CSS3* e JavaScript. Assim então sendo independente de linguagens e plug-ins de terceiros, e utilizando apenas linguagens e textos de marcação padrões de navegadores WEB modernos, como por exemplo, o Google Chrome, o Mozilla Firefox, o Safari, o Microsoft Edge, entre outros.

Figura 3 - Logos dos Navegadores Modernos

Fonte: Elaborada pelo autor

O produto final deste trabalho é uma aplicação cliente/servidor, ou seja, dois projetos que se comunicam entre si, sendo eles o projeto do lado servidor e o projeto do lado cliente. O lado servidor deve manter todos os dados das funcionalidades que necessitam de autenticação, comunicação e persistência. O lado cliente que tem a responsabilidade da interação com o usuário final, onde poderá ser criado e simulado os circuitos lógicos digitais.

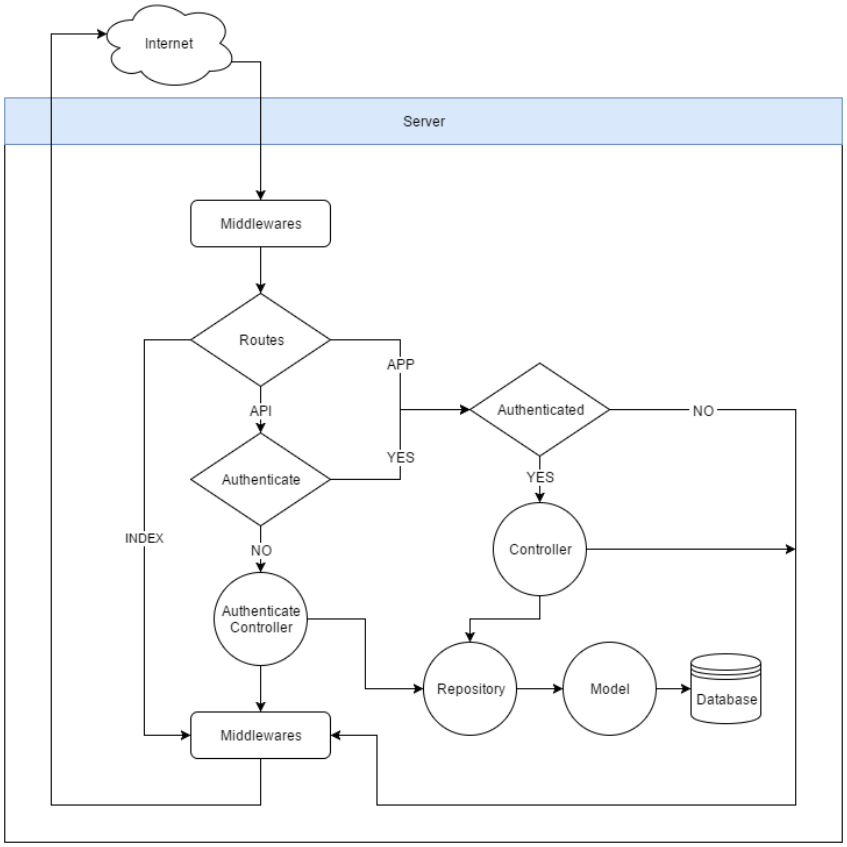
Figura 4 - Cliente / Servidor

Fonte: Elaborada pelo autor

2.1.Lado Servidor

O servidor foi projetado e desenvolvido sobre *NodeJs* que é uma plataforma para desenvolvimento de aplicações *"server-side"* baseadas em rede utilizando Javascript e a engine *Chrome's V8 JavaScript*, junto com o framework *Express* utilizando do padrão *middleware*. A plataforma foi escolhida baseando-se em sua linguagem de programação *Javascript*, para que assim as aplicações, tanto servidor quanto cliente sejam desenvolvidas com uma mesma linguagem de programação, assim facilitando e agilizando o processo de criação e manutenção do software.

Figura 5 - Lado Servidor



Fonte: Elaborada pelo autor

2.1.1. Persistência

A persistência utilizará de um ORM (Object Relational Mapper) para realizar a mesma, ou seja, uma técnica para mapeamento de modelos, que são objetos/classes da

aplicação, com os modelos relacionais do banco de dados, assim abstraindo o acesso ao banco de dados e agilizando o processo de desenvolvimento de um trabalho.

Sendo o lado servidor responsável por autenticação e comunicação entre a aplicação e o usuário final, o mesmo deve então persistir os dados tanto de usuários quanto de aplicação, sendo assim necessário a utilização de um banco de dados para tal persistência. Com a intenção de aumentar a produtividade no desenvolvimento como também para que a aplicação não seja dependente de um banco de dados específico, foi optado pela a utilização de um *ORM* para manter tais dados. Sendo assim a escolha que mais se adequou as necessidades que o software apresentou foi o ORM *Sequelize*, pois o mesmo tem suporte para "*PostgreSQL, MySQL, MariaDB, SQLite e MSSQL"* ("Sequelize", 2016), e assim diminuindo a responsabilidade e cuidados com o banco de dados a ser utilizado.

2.1.2. Autenticação

A autenticação do servidor será realizada de forma a se utilizar JWT (JSON Web Token), onde após a verificação do email e a senha do usuário, é retornado para o consumidor um token de acesso, e o mesmo por sua vez deve ser retornado em todo e qualquer request ao servidor onde a autenticação do usuário é necessária ou obrigatória, tendo também um tempo de vida, onde quando o mesmo é expirado, o token de acesso do usuário se torna inválido, e então o cliente é obrigado a realizar uma nova autenticação no servidor para que seja então gerado um novo token de acesso para continuar a comunicação.

Sua implementação foi realizada junto a biblioteca *jsonwebtoken* que em paralelo com o *Express* e métodos criados para a pesquisa dos dados do usuário pelo o ORM realizem os processos para gerar e verificar os tokens de acesso de um usuário, assim então fazendo com que a aplicação seja capaz de identificar e retornar de forma correta a um request, sendo com os dados solicitados ou com o status "401 Unauthorized" (W3C, 2016a).

Para que tais verificações sejam realizadas, foi programado então middlewares, que devem estar presentes em toda rota da aplicação que a autenticação é necessária ou obrigatória, assim agilizando o processo do retorno para o consumidor.

2.1.3. Comunicação cliente/cliente

Como este projeto propõe ser um software colaborativo, o mesmo necessita de uma comunicação cliente/cliente, onde os dados são sincronizados através de um socket, utilizando

protocolos como *web-socket* ou *long-pulling* por exemplo. Neste caso, se mantem uma comunicação do servidor com o cliente, diferente de um request normal na internet onde apenas o cliente solicita dados ao servidor, o servidor também poderá solicitar ou enviar novos dados ao cliente, sem que o mesmo solicite, assim possibilitando a transmissão dos dados de um cliente para o servidor, e do servidor para os outros clientes.

Contudo, foi adotado a biblioteca *socket.io*, que "*permite a comunicação bidirecional baseada em eventos em tempo real*" ("Socket.io", 2016). A biblioteca já implementa toda a comunicação do lado servidor quando do lado cliente com a linguagem *JavaScript*, utilizando de emissão ou broadcast para transmissão dos dados desejados e ainda a utilização de salas, onde um broadcast é transmitido apenas para aqueles usuários que entraram na sala em questão.

Apesar da biblioteca auxiliar no desenvolvimento, foi também elaborado uma série de módulos para a utilização de padrões e métodos como middlewares, constrollers, factories, providers e services, assim subdividindo as necessidades da comunicação e transmissão de dados em pequenas partes mais simples, diminuindo então a responsabilidade de cada subdivisão deste serviço, e separando as mesmas de forma a simplificar a manutenção do projeto por parte do desenvolvedor.

2.1.4. Gerenciador de pacotes

"NPM é o gerenciador de pacotes para JavaScript" (NPM, 2016) utilizando para gerenciar os pacotes JavaScript contidos neste projeto, mantendo as versões necessárias de cada pacote, o gerenciador é simplificado e utilizado a partir de comandos simples pelo terminal e/ou prompt de comando, como "npm install", "npm update", entre outros.

O mesmo também mantem o arquivo de gerenciamento de pacotes "pakage.json", adicionando, removendo e atualizando as dependências do projeto. Escrito em JSON (JavaScript Object Notation) que é uma forma leve para comunicação de dados computacionais, utilizando de chaves e valores para o mesmo. O arquivo pakage.json contém também informações sobre o projeto, como nome, descrição, versão, autor, etc, que por muitas vezes são utilizados durante sua publicação em hospedagens para indicar, por exemplo, arquivos de entrada da aplicação, e configurações de produção.

2.2.Lado Cliente

O lado cliente desta aplicação foi desenvolvido utilizando de tecnologias WEB, sendo elas o HTML5, CSS3 e JavaScript. Apesar de que apenas estes recursos sejam o suficiente para criar os elementos visuais de um site e/ou uma aplicação, este projeto utilizou de alguns frameworks para agilizar a criação de componentes e realizar binds dos elementos HTML com os dados mantidos pelo JavaScript, assim como também para realizar comunicações sendo um consumidor do lado cliente do projeto, fornecendo e recebendo dados de usuários e circuitos criados com esta ferramenta.

2.2.1. AngularJS

Um dos frameworks utilizados foi o AngularJS, que oferece recursos como data bindings, controllers, deep linking, validação de forms, comunicação com servidores, diretivas, injeção de dependências, entre outros, para auxiliar o desenvolvedor durante o processo de criação de uma aplicação WEB em JavaScript.

Outro recurso importante fornecido pelo AngularJS utilizado na aplicação, foi o controle de rotas, que por sua parte fez com que, a ferramenta se tornasse uma aplicação single page, ou seja, para atualizar seu conteúdo, a mesma não depende de redirecionamento para a mesma ou outras páginas, mas sim, por meio de requests via *Ajax*, recendo seu conteúdo por meio de um call-back realizado utilizando XMLHttpRequest fornecido pelo próprio JavaScript.

Este tipo de request também é utilizado para a comunicação com a *api* deste projeto que fica no lado servidor da aplicação, para fazer consultados a dados sem que a página sofra um refresh a todo momento que um novo dado for requisitado ao servidor. Dentre estes dados se destacam os dados de cadastro e login de usuário, dados de projetos assim como seu compartilhamento, e controle de amizades entre usuários da ferramenta.

2.2.2. Lumx

Para a construção dos elementos visuais, foi utilizado um pacote chamado *lumx*, que é "*um framework front-end baseado no Google Material Design*" (LUMAPPS, 2016). Google Material Design "*é um ambiente tridimensional contendo luz, materiais e sombras fundidas"* ("Material Design", 2016).

O framework contém componentes como toolbar, checkbox, text fields, tabs, buttons, dialogs, entre outros utilizados durante o desenvolvimento do visual da ferramenta, agilizando o processo e diminuindo a preocupação com a aparência e design do mesmo.

2.2.3. Canvas

Canvas é um elemento HTML utilizando para renderizar/desenhar gráficos em páginas WEB, sendo tais gráficos gerados a partir de scripts escritos em JavaScript utilizando de um contexto 2D que "*fornece objetos, métodos e propriedades para desenhar e manipular gráficos em uma superfície*" (W3C, 2016b).

Neste projeto o elemento foi utilizado para desenhar todo o workspace do usuário, desde marcações de grid, até componentes, portas lógicas digitais, fios, inputs, outputs, entre outros. Também é o elemento que recebe as ações realizadas com o mouse, propagando-as para outros serviços com responsabilidades distintas.

Figura 6 - Utilização de canvas no projeto

```
<canvas id="{{workspace.canvasId}}" class="bgc-white-1 z-depth1 workspace-bg"
        dnd-drop="workspace.addSource($event, $resource)"
        ng-mousemove="workspace.mouseMove($event);"
        ng-click="workspace.click($event)"
        ng-rightclick="workspace.rightClick($event)"
        ng-dblclick="workspace.dblClick($event)"
        ng-mousedown="workspace.mouseDown($event)"
        ng-mouseup="workspace.mouseUp($event)"
        ng-mouseleave="workspace.mouseLeave($event)"
        ng-class="{'draggable': workspace.compHover()}"
        ng-style="{cursor: workspace.getCursor()}"></canvas>
```

Fonte: Elaborada pelo autor

3. Desenvolvimento

Utilizando as tecnologias citadas anteriormente, este projeto foi desenvolvimento exclusivamente com a linguagem de programação JavaScript, sendo seu lado servidor sendo executando em cima do NodeJS, utilizando pacote Express para o auxílio no controle de rotas e middlewares.

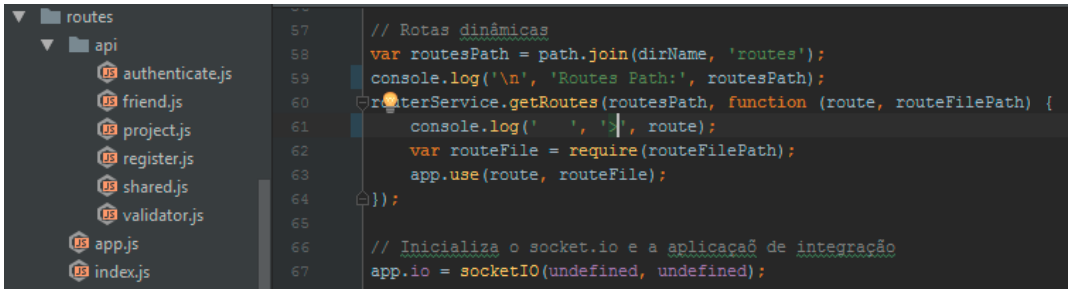
3.1. Rotas da aplicação

Apesar de que o *Express-Generator* que cria uma estrutura inicial para um projeto baseado em Express o prepare para casos mais genéricos, e de cadastros manuais de rotas, este projeto tente a ser mais dinâmico, assim obtendo a necessidade da criação de um modulo para que as rotas sejam geradas dinamicamente, baseando apenas sua hierarquia de pastas, sendo assim, apenas com a necessidade de oferecer ao módulo o ponto inicial das rotas.

Tal módulo, criado como um serviço da aplicação recebe o caminho inicial das rotas, ou seja, a pasta onde contém as rotas da aplicação, identificando qualquer arquivo com extensão js (*.js) como uma rota válida, onde um arquivo *index.js* é identificado como uma entrada padrão, e seu valor para uma rota é vazio. Para tal recurso foi criado então um novo serviço para percorrer as pastas encontradas e assim oferecendo um recurso completo para gerenciar rotas, já que com o serviço de *FileWalker* junto com o serviço de *RouterService*, o desenvolvedor pode criar uma hierarquia de pastas onde os nomes das pastas serão prefixos das rotas identificadas pelo nome os arquivos de extensão js.

Por exemplo, na figura 7, temos as rotas da aplicação, sendo que para acesso a rota de app.js, a url que deverá ser acessada será "*http://host/app*" e para acesso a rota authenticate.js, a url deverá ser "http://host/api/authenticate".

Figura 7 - Rotas da aplicação



Fonte: Elaborada pelo autor

3.2. Autenticação

A autenticação será realizada com a utilização de JWT (JSON WEB Token), que é uma forma de controlar o acesso de um usuário em uma aplicação WEB, de modo que o usuário fornecendo seus dados de acesso, o servidor gera um token com os dados do mesmo, e o devolve para o usuário. Assim a todo novo request que o usuário realizar ao servidor novamente, o mesmo deverá fornecer o token recebido, para fins de verificação de autenticação do usuário em determinadas áreas e/ou páginas da aplicação WEB.

Para a implementação de autenticação baseada em JWT, foi utilizado dois pacotes em conjunto com as rotas, middlewares e o ORM Sequelize, sendo eles o *Express-Session* para controle de sessões em aplicações Express, e *Jsonwebtoken* para gerar e gerenciar os tokens de sessão de usuários.

Tais tokens são gerados a partir de um requeste de autenticação, onde com o Sequelize é feio uma consulta ao banco de dados para verificar se os dados fornecidos são de um usuário cadastrado e se sua senha corresponde a senha enviada no request. A partir deste momento, o lado cliente recebe um token de acesso, onde o mesmo deverá ser então enviado de volta ao servidor em todos os novos requests a través de um cabeçalho customizado de chave "*x-access-token*", ou ainda por meio do próprio corpo de um request ou query string com nome de token, para que então o servidor identifique o token, verifique que o mesmo existe e ainda não tenha sido expirado, para então permitir ou recusar tal request.

O lado cliente é totalmente responsável por manter tal token, onde neste projeto foi utilizado o banco de sessão dos navegadores e/ou cookies de sessão para persistir o mesmo temporariamente, sendo isso realizado com o auxílio do AngularJS.

3.3. Comunicação cliente/cliente

Para a comunicação entre os clientes, foram implementados a estrutura descrita anteriormente, assim facilitando o consumo do pacote socket.io, onde cada rota indica ao desenvolvedor qual o evento a ser chamado para que um controller seja invocado durante a comunicação.

O controle para que cada usuário possa editar um circuito específico onde o mesmo deve ser único para todos os envolvidos na edição do mesmo teve sua persistência implementada então não apenas no cliente, mas também como um servido do servidor, assim

com uma comunicação aberta, e com usuários dentro de uma sala criada a partir do momento que um projeto é aberto para edição em compartilhamento, o serviço consegue ter então o controle da persistência do mesmo no Banco de Dados pelo ORM.

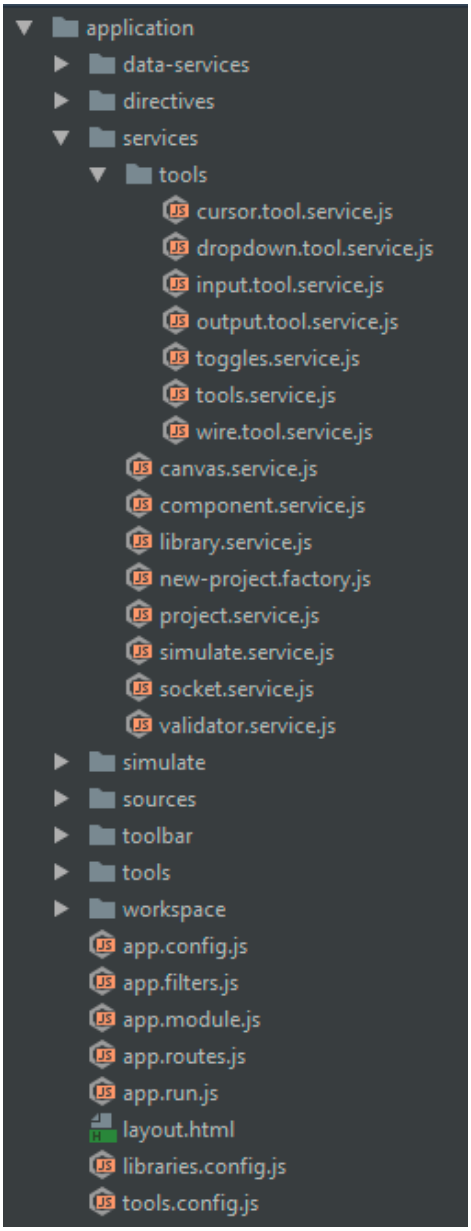
Tal atualização ocorre apenas em eventos específicos, como adição e remoção de componentes, adição e remoção de fios e por fim “drop” de um “drag-and-drop” de um componente, ou seja, quando um componente é movido de lugar, e solto pelo usuário, quando seu movimento para. Este controle possibilita uma persistência mais limpa do projeto editado em compartilhamento, alterando apenas ações finais de usuários, pois ações não finais como posição de um componente que ainda está sendo arrastado no workspace do projeto não é um item importante a ser persistido, mas sim apenas seu estado e/ou posição final.

3.4. Criação e Simulação de Circuitos Lógicos Digitais

A página de desenvolvimento de circuito lógicos digitais foi desenvolvida com AngularJS e o elemento canvas do HTML5 que é controlado por JavaScript. Para o controle do mesmo via AngularJS, foi necessário criar um serviço, para que uma alteração seja vista e aplicada no scope do framework, mantendo os data-binds em sincronismo.

Cada recurso da ferramenta foi desenvolvido separadamente, sendo cada uma um serviço distinto, ou um grupo de serviços como é mostrado na figura 8. Dentre eles se destacam os serviços de projeto, simulação, canvas, componentes, bibliotecas e ferramentas, que controlam as principais responsabilidades da aplicação.

Figura 8 - Estrutura da ferramenta em AngularJS



Fonte: Elaborada pelo autor

3.4.1. Serviço de Projeto

O Serviço de projeto é responsável por controlar toda e qualquer alteração ocorrida em um circuito que esteja sendo criado e/ou editado, assim como adição e exclusão de componentes e fios, e também os enviando para o serviço de comunicação de compartilhamento para que o mesmo possa ser enviado ao servidor, ou enviando para a api do servidor, para que o circuito possa ser persistido ou alterado no Banco de Dados.

3.4.2. Serviço de Simulação

Serviço de simulação é onde ocorre a interpretação de cada componente, e simulado o disparo de um sinal de entrada, onde o mesmo é repassado através de sua hierarquia de componentes, até que o mesmo encontre um componente de output, ou que o fio já contenha o mesmo valor, para que assim o mesmo pare de ser propagado.

Tais envios de sinal são realizados de forma assíncrona, assim evitando que o navegador seja congelado por uma ação demorada do script, e também para que cada componente possa simular seu tempo de propagação, por meio de um evento de "*timeout*" oferecido pelo JavaScript que por sua vez é consumido pelo serviço "$timeout" do AngularJS.

3.4.3. Serviço de Canvas

Para a renderização de cada componente e fio da aplicação, foi criado o serviço de canvas, onde o mesmo recebe uma lista de componentes e fios que devem ser desenhados por ele no elemento canvas. Assumindo cada parâmetro encontrado na configuração do componente para que o mesmo seja renderizado no workspace, ou assumindo as ações descritas para realizar o mesmo processo para os fios criados.

Cada parâmetro de configuração de um componente para um desenho customizado deve conter então um tamanho, posições de tamanhos de seus inputs e outpus, e um corpo que por sua vez deve conter as instruções a serem realizadas. Tais instruções são métodos fornecidos pelo Contexto-2D de um elemento canvas, assim como demonstrado na figura 9, recebendo seu método e seus parâmetros.

Figura 9 - Desenho customizado do componente NAND2x1

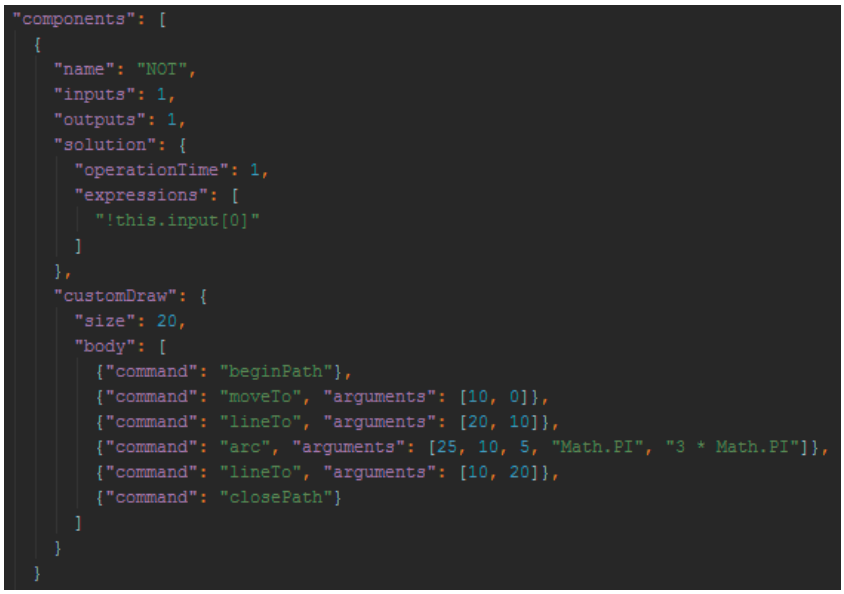
```
"customDraw": {
  "size": 50,
  "inputs": [
    {"x": 0, "y": 20},
    {"x": 0, "y": 40}
  ],
  "outputs": [
    {"x": 0, "y": 30}
  ],
  "body": [
    {"command": "beginPath"},
    {"command": "moveTo", "arguments": [10, 10]},
    {"command": "lineTo", "arguments": [30, 10]},
    {"command": "arc", "arguments": [30, 30, 20, "1.5 * Math.PI", "0 * Math.PI"]},
    {"command": "arc", "arguments": [55, 30, 5, "Math.PI", "3 * Math.PI"]},
    {"command": "arc", "arguments": [30, 30, 20, "2 * Math.PI", "0.5 * Math.PI"]},
    {"command": "lineTo", "arguments": [10, 50]},
    {"command": "closePath"}
  ]
}
```

Fonte: Elaborada pelo autor

3.4.4. Serviço de Componentes e Serviço de Biblioteca

O Serviço de componentes é responsável por controlar todo e qualquer componente de uma biblioteca, transformando suas configurações em métodos e objetos para que o mesmo possa ser desenhado e simulado corretamente. Cada componente é validado antes de que o mesmo possa ser utilizado na biblioteca por este serviço, que para tal ação seja bem-sucedida, todo e qualquer componentes deve ser escrito como apresentado na figura 10, contendo nome, quantidade de inputs, quantidade de outputs, tempo de operação, expressão de execução do componente, e opcionalmente um desenho customizado.

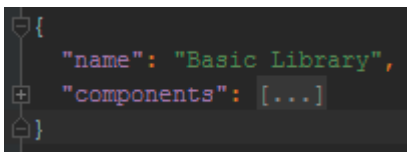
Figura 10 - Componente NOT



Fonte: Elaborada pelo autor

Assim como o serviço de componentes, o serviço de bibliotecas tem o controle e a validação de entradas de bibliotecas, que por sua vez, necessita apenas de um nome e uma lista de componentes como mostrado na figura 11.

Figura 11 - Biblioteca



Fonte: Elaborada pelo autor

3.4.5. Serviços de Ferramenta

São serviços de controlam as ferramentas disponibilizadas para os usuários, como cursor, fio, input e output. Sendo cada ferramenta controlada por um serviço geral de ferramentas "*tool.service.js*", onde cada entrada é recebida e enviada para a ferramenta atualmente selecionada.

Sendo cada ferramenta responsável por uma ação diferente, temos que a ferramenta cursor permite ao usuário selecionar e realizar drag-and-drop nos componentes dentro do workspace, assim como também possibilita a exclusão dos mesmos com o uso do clique do botão direito do mouse em cima do componente, onde surgirá um dropdown com as ações permitidas, ou ainda, se o mesmo clique for realizado em uma entrada ou saída do componente, um dropdown com ações de gatilho rápido surgirão, para que o mesmo possa criar inputs e outputs de maneira mais rápida.

A ferramenta de fio, é exclusiva para criar ligações entre componente, onde a mesma pode ser utilizada de duas formas, sendo por drag-and-drop, ou por meio de dois cliques, onde o primeiro cria o fio, e o segundo finaliza o mesmo. Nenhum fio pode ser criado sem uma ligação válida. E por fim, um fio é fixo, porem um duplo clique em um fio, gera um nó de fio, que pode ser movido.

As ferramentas de input e output são exclusivas para criar os componentes de mesmo nome, sendo elas criadas com um clique normal no workspace, ou um clique direto em uma entrada ou saída de um componente, assim criando também o fio de ligação entre a ferramenta e o componente desejado, agilizando o processo de criação do circuito.

4. CELtIC

CELtIC, acrônimo de Create and Simulate Logic Circuit, ou, traduzindo para o Português, Criar e Simular Circuitos Lógicos, é o nome da ferramenta proposta para este trabalho. Sendo este projeto, uma aplicação WEB que contém controle de usuários, controle de projetos, controle de amizades e compartilhamentos de projetos e principalmente a criação e edição de projetos de circuitos lógicos digitais.

Figura 12 - Logo CELtIC

Fonte: Elaborada pelo autor

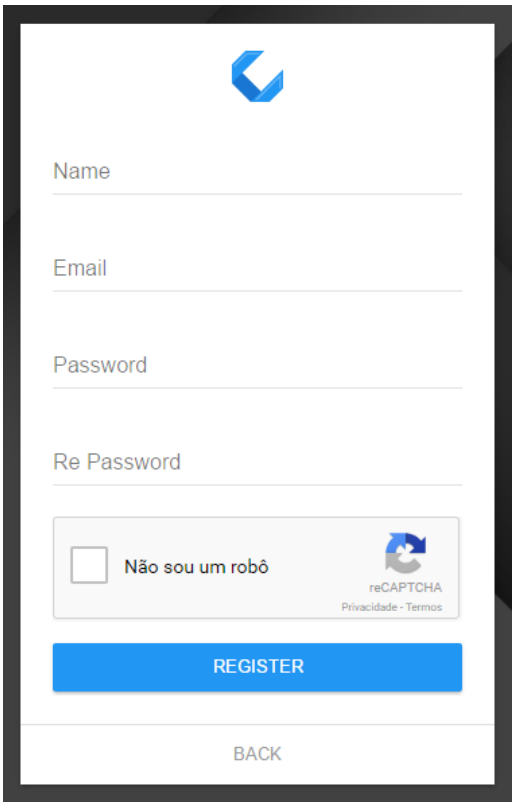
A utilização da ferramenta é baseada apenas do lado cliente da aplicação, assim desconsiderando de que o usuário tenha qualquer ação ou interação com o lado servidor de forma manual, sendo então, toda e qualquer comunicação entre os dois lados, transparente para um consumidor final do projeto.

A interação com o usuário consiste em quatro áreas distintas para controle desses recursos, sendo a área de cadastro, área de login, área de dashboard e área de aplicação, cada uma contendo seus respectivos conteúdos.

4.1. Cadastro

Para a utilização da aplicação é necessário que um novo usuário realize o seu cadastro na aplicação, fornecendo um nome, um email válido, uma senha, confirmar a senha fornecida e por fim confirmar que é um usuário real, assim evitando cadastros inválidos e/ou maliciosos. Após a confirmação dos dados fornecidos, o usuário receberá um email que será enviado para o email de seu cadastro, um link para confirmar que seu email é válido.

Figura 13 - Cadastro de novo Usuário

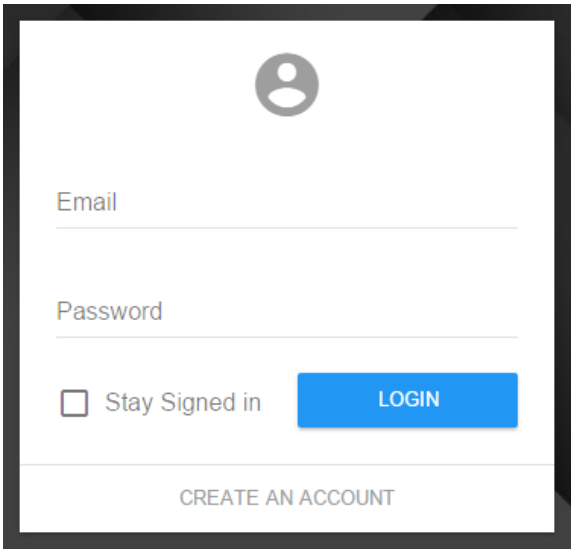


Fonte: Elaborada pelo autor

4.2.Login

Um usuário que possui um cadastro no site e também tem o mesmo já confirmado pelo link recebido em seu email de cadastro, pode então realizar o login na aplicação, para então ser redirecionado ao ser dashboard. Para realizar o login, o usuário deve fornecer alguns dados, sendo eles, o email e a senha cadastradas anteriormente, tendo também a opção para permanecer logado, o que faz com que a aplicação entenda que ele não deseja realizar o login quando a sessão do mesmo ser expirada. Para este projeto, permanecer logado apenas prolonga a vida útil do *JWT*, ou seja, o usuário apenas tem mais tempo para permanecer logado, e não apenas com uma sessão que é expirada normalmente conforme as regras de sessão do navegador utilizado.

Figura 14 - Login do Usuário

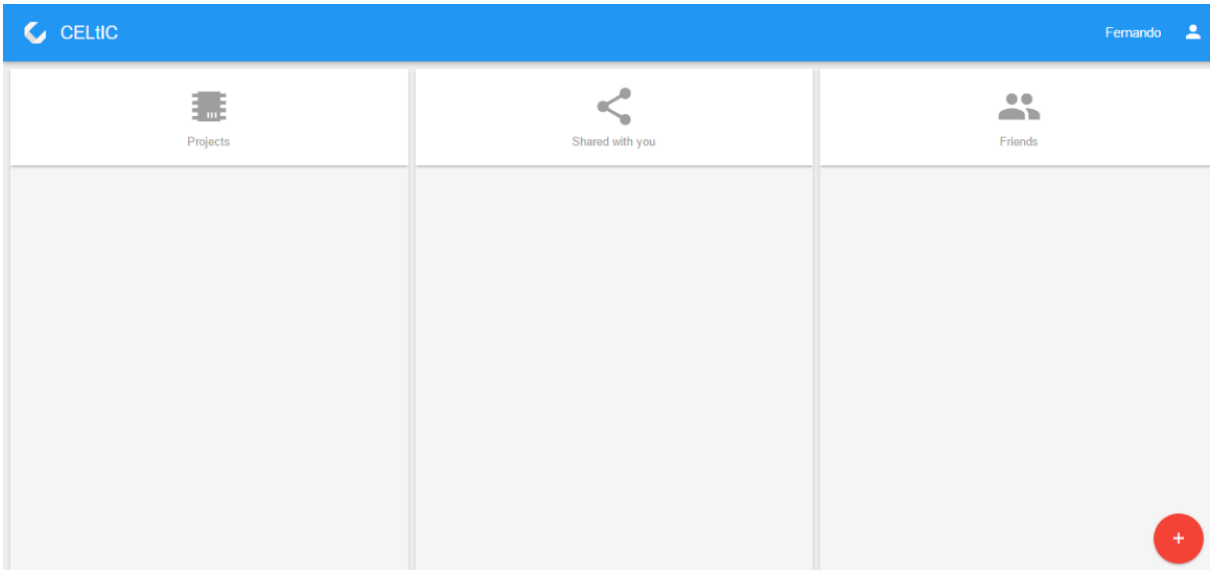


Fonte: Elaborada pelo autor

4.3.Dashboard

O Dashboard mostra ao usuário, controles para que ele possa manter seus projetos, manter os projetos que foram compartilhados com ele, e por fim, manter suas amizades.

Figura 15 - Dashboard

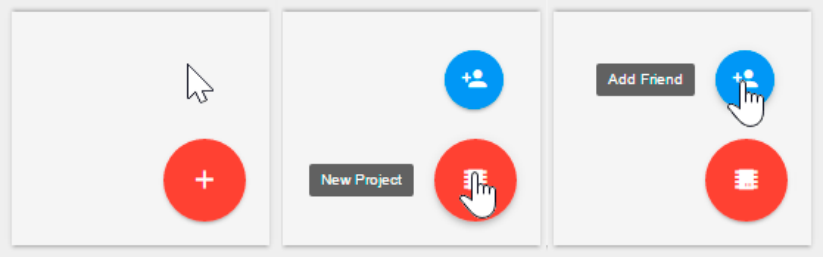


Fonte: Elaborada pelo autor

4.4.Projetos

Para criar um novo projeto o usuário pode ir com o mouse sobre o botão avermelhado em formato de círculo com um ícone de “+”, o que abrirá as possibilidades de adição do dashboard, que são novo projeto e nova amizade.

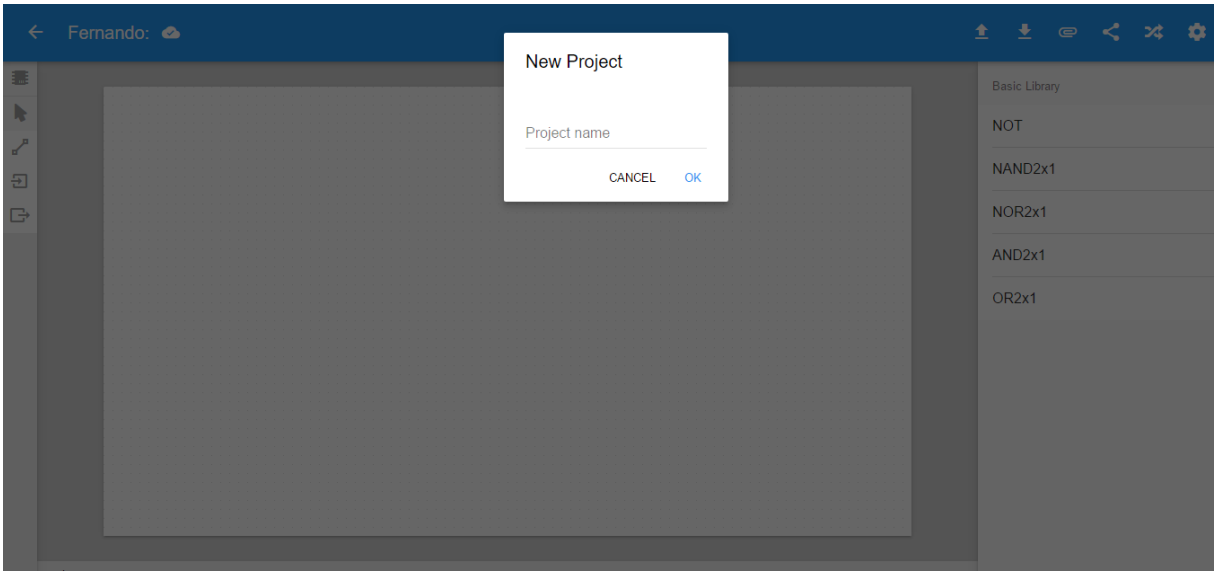
Figura 16 - Botão Novo



Fonte: Elaborada pelo autor

Ao criar um novo projeto, o usuário é redirecionado para a página de aplicação, onde ele deverá nomear seu circuito, e posteriormente criar o mesmo. Caso o usuário cancele a nomeação do circuito, o mesmo será enviado de volta a página de dashboard.

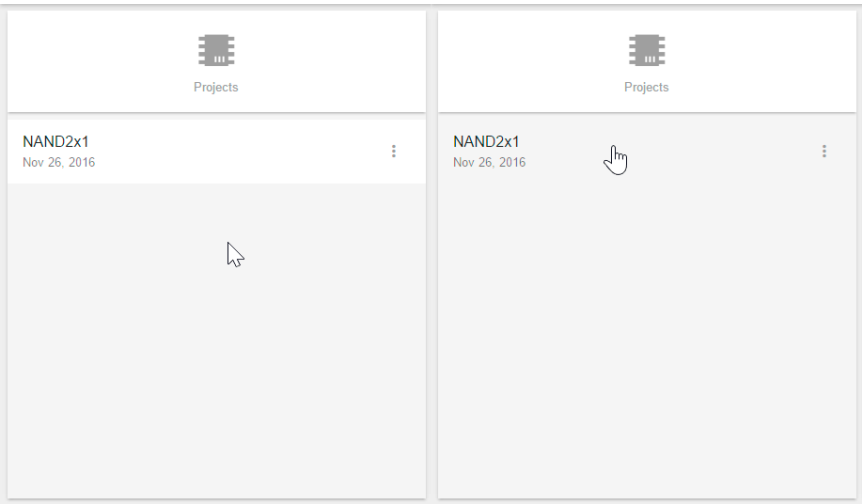
Figura 17 - Novo Projeto



Fonte: Elaborada pelo autor

Com um ou mais projetos criados, o dashboard mostrará uma lista de os mesmos, onde para que o usuário volte a editar um projeto específico, basta realizar um clique no item desejado quando uma indicação de mouse sobre o item estiver visível, que neste caso mudará a cor do item para cinza claro.

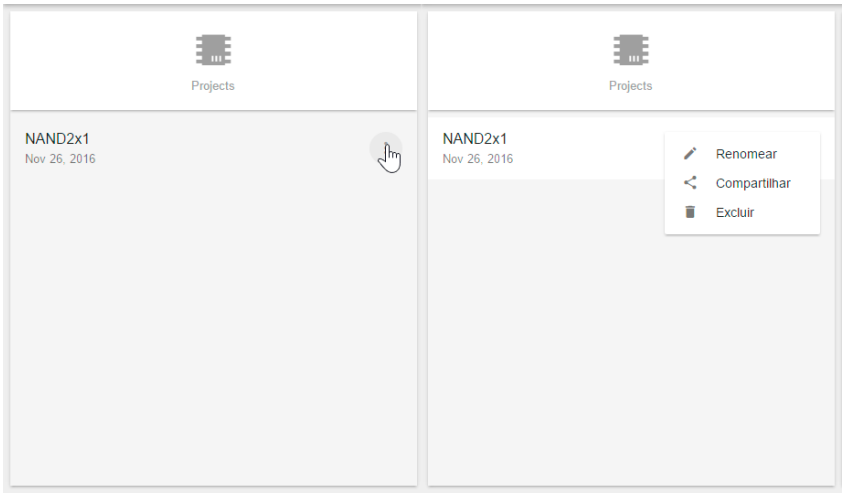
Figura 18 - Seleção de Projetos



Fonte: Elaborada pelo autor

O usuário também poderá ver um botão de overflow, que contem mais ações que possam ser realizadas para os projetos listados, assim como renomear, compartilhar e excluir o projeto, cada ação abrindo seu respectivo modal ou dialog para controle do mesmo.

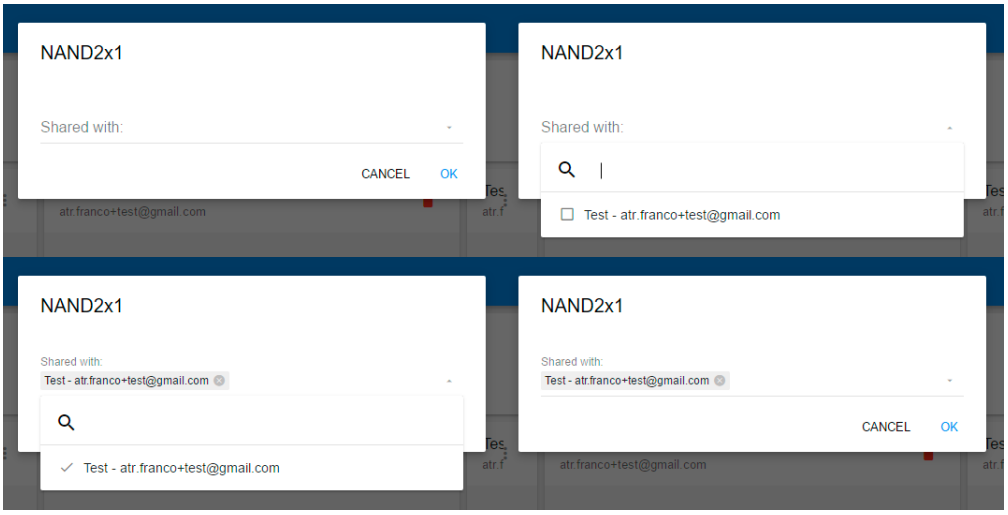
Figura 19 - Botão de Overflow



Fonte: Elaborada pelo autor

No modal de compartilhar um projeto, é apresentado ao usuário, um componente lumx para seleção e pesquisa de itens, onde ao se clicar, é exibida um input para filtro de amizades, e uma lista de amizades do usuário, onde o mesmo pode escolher os usuários que poderão realizar alterações em seu projeto (compartilhar o projeto com a amizade).

Figura 20 - Modal de compartilhamento

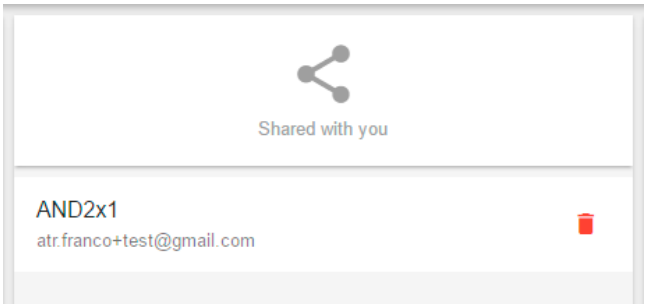


Fonte: Elaborada pelo autor

4.5.Projetos Compartilhados

Assim como os projetos criados pelos usuários, é exibida uma lista dos projetos que foram compartilhados com ele, porem nestes itens, ao excluir um projeto pelo botão de exclusão (ícone de uma lixeira), o usuário não excluirá o projeto real, mas sim, apenas cancelará sua participação no compartilhamento do projeto.

Figura 21 - Lista de projetos compartilhados



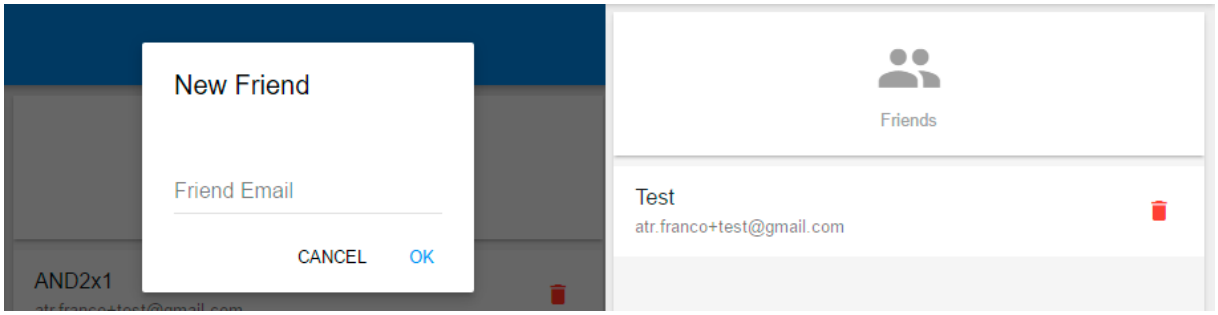
Fonte: Elaborada pelo autor

4.6. Amizades

As amizades são necessárias para que o usuário possa compartilhar seus projetos com seus amigos, assim tendo a possibilidade da edição do projeto em modo colaborativo, onde todos os usuários que o projeto está compartilhado, poderão editar e visualizar as edições de seus amigos simultaneamente, sem a necessidade de um refresh na página da aplicação.

O controle de amizade é simplificado, podendo apenas adicionar uma amizade, ou remove-la utilizando o botão de exclusão encontrado em um item na lista de amizades.

Figura 22 - Modal e Lista de Amizades



Fonte: Elaborada pelo autor

4.7. Aplicação

A aplicação é onde o usuário pode construir e simular seus circuitos, utilizando de bibliotecas e ferramentas disponibilizadas, e auxiliando com identificações sobre o estado de persistência do projeto atualmente sendo criado/editado. Os recursos descritos são divididos em quatro categorias, onde temos o toolbar, toolbox, sources e workspace.

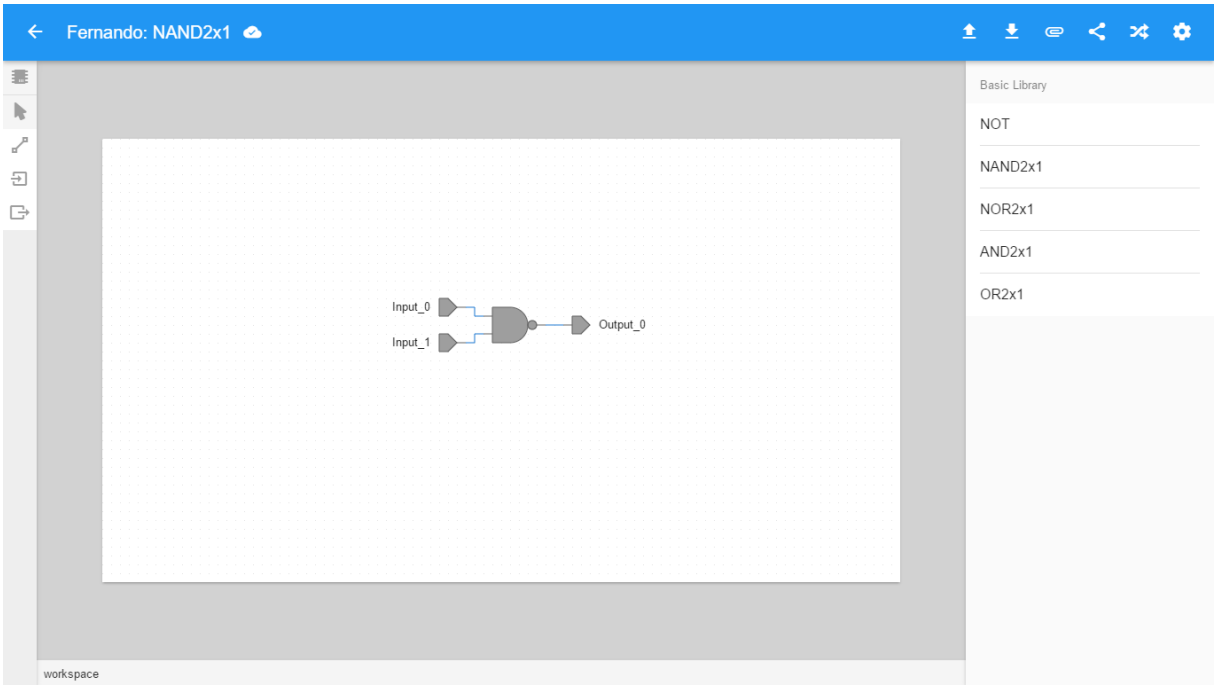
O Toolbar é responsável por ações como notificações de persistência, importação e exportação de projetos, importação de bibliotecas externas, compartilhamento, simulação e configuração de projeto.

Toolbox é uma lista de ferramentas que podem ser utilizadas para a edição de um projeto, ou seja, o responsável pelas possíveis ações que um usuário pode realizar com o circuito, como movimentar um componente, excluir componentes, criar e modificar fios, e adicionar inputs e outpus ao circuito.

Sources é uma lista de bibliotecas que contém componentes que podem ser adicionados ao circuito utilizando de dra-and-drop, que consiste em clicar e arrastar um componente do sources para o workspace.

Workspace é a área onde o usuário pode trabalhar, ou seja, onde ele criará seus projetos de circuitos lógicos digitais. A mesma é subdivida em grid representado por pontilhados em sua apresentação, assim auxiliando e facilitando a criação dos circuitos.

Figura 23 - Aplicação apresentando o circuito de uma porta NAND2x1

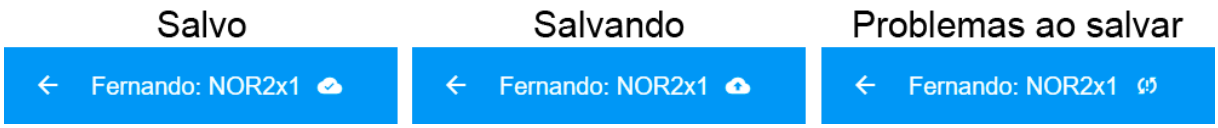


Fonte: Elaborada pelo autor

4.7.1. Indicador de persistência

Localizado no canto superior esquerdo, é responsável por indicar ao usuário o estado atual da persistência de seu projeto, sendo três estados que indicam respectivamente se o projeto está salvo, sendo salvo, ou se ocorreu algum problema não previsto ao salvar o projeto na nuvem. Cada estado é representado por um ícone como mostra a figura 24.

Figura 24 - Estados de persistência



Fonte: Elaborada pelo autor

4.7.2. Importar e exportar circuitos

É possível pela aplicação realizar a importação e exportação de circuitos por arquivos em formato JSON, que contém os detalhes de um projeto. Sendo que para cada ação, existe um botão específico.

4.7.3. Importar bibliotecas

O botão de importação de bibliotecas é usado para adicionar uma biblioteca externa à lista de bibliotecas da ferramenta. Tal biblioteca deve ser um arquivo JSON com um conteúdo e componentes válidos, caso contrário a mesma pode ser recusado pela ferramenta.

4.7.4. Compartilhar projetos

Assim como no dashboard, o botão de compartilhamento abrirá um modal para que o usuário selecione suas amizades que poderão editar o projeto de forma síncrona.

4.7.5. Simulação

O botão de simulação abre um modal, cujo conteúdo é dependente das entradas e saídas do circuito criado, onde também contém um toolbar com botões, sendo eles o botão de próximo passo (realiza um novo passo na simulação do circuito), de play (automatiza a simulação do circuito por uma velocidade configurável entre 500, 1000, 2000 ou 4000 milissegundos para cada passo), e de limpar (Limpa todos os estados e passos dados na simulação).

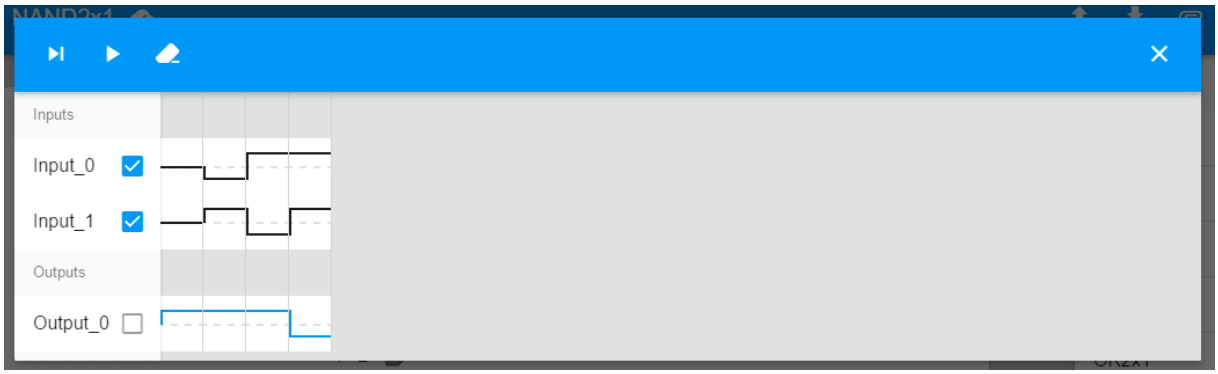
No canto esquerdo do modal é possível visualizar então seus inputs e outputs, onde os outputs não contém nenhum tipo de interação, e são apenas apresentados ao usuário de modo que o mesmo possa identificar o estado de cada saída do circuito em cada passo de simulação.

Os inputs possuem duas interações com o usuário, onde um clique em seu checkbox pode mudar seu estado para verdadeiro (equivalente ao sinal 1 em binário) ou falso (equivalente ao sinal 0 em binário), sendo indicado seu estado em cada passo.

Outra ação sendo o clique em qualquer outra posição do input para que o mesmo abra um dropdown contendo três ações, respectivamente para atribuir o valor do input para não definido, outra para indicar que o input é do tipo de entrada manual, e outra para indicar que o input é um clock com entrada automática, sendo alterada a cada passo de simulação para a inversão de seu valor atual, por exemplo, caso o input seja do tipo clock, e o mesmo tenha o valor true, no próximo passo de simulação o mesmo receberá o valor false, e no passo seguinte o valor true novamente, e assim respectivamente.

Cada passo de simulação é armazenado como um gráfico com bordas de subida e descida sendo respectivamente os valores true e false de cada entrada ou saída como demostrado na simulação de uma porta NAND com duas entradas e uma saída na figura 25.

Figura 25 - Simulação da porta NAND2x1



Fonte: Elaborada pelo autor

4.7.6. Configuração

O botão de configuração abre um modal com as configurações de tamanho do workspace em altura e largura, onde o usuário pode alterar e obter um workspace maior ou menor conforme sua necessidade.

4.7.7. Bibliotecas

O botão de bibliotecas é utilizado para esconder ou exibir as bibliotecas do projeto, assim possibilitando ao usuário ter um espaço maior e mais limpo para editar o circuito atual. Este botão é localizado no canto esquerdo, junto ao toolbox, acima da ferramenta de cursor.

4.7.8. Cursor

Ferramenta principal da ferramenta, onde o usuário pode editar a posição de componentes e fios, assim como também excluir os mesmo ao utilizar o botão direito em cima do item a ser removido, e ainda com a possibilidade de gatilho rápido para a criação de inputs e outputs em componentes que contem entradas e saídas, assim agilizando o processo de criação de circuitos dentro do projeto.

4.7.9. Fio

Utilizado para criar ligações entre componentes, é um recurso essencial para o projeto, onde o mesmo indicará para onde um input enviará seu sinal de entrada, já que o mesmo está relacionado a uma entrada de um componente posterior na hierarquia de componentes do circuito, assim o componente pode executar sua expressão, e envia para sua saída onde contém um ou mais fios conectados que passaram este sinal adiantes, e assim consecutivamente.

4.7.10. Input e Output

Utilizados da mesma maneira, os dois adicionam inputs e outputs ao projeto, sendo que o mesmo pode ser adicionado de forma genérica com um clique normal no workspace, assim como de forma dinâmica, onde o clique é realizado em uma entrada ou saída de um componente, assim criando um fio ligando o input/output com o componente desejado, agilizando então o processo de criação do circuito.

4.7.11. Biblioteca Básica

A biblioteca básica está presente em todo e qualquer circuito a ser editado, contendo as os componentes de portas lógicas NOT, NAND2x1, NOR2x1, AND2x1 e OR2x1. Sua

função é exibir tais componentes para que o usuário possa utilizá-los na criação de seus projetos de circuitos lógicos digitais, utilizando de drag-and-drop para adiciona-los ao workspace.

4.7.12. Workspace

O workspace é uma área onde o usuário irá criar e editar seus circuitos, utilizando de grid, e sendo possível alterar seu tamanho para melhor se adequar ao projeto, é a principal área da aplicação. Também contém uma barra de status, onde é quando o mouse não esteja sobre nenhum componente ou fio, é exibido a palavra workspace, porem com o mouse sobre um componente ou fio é exibido em seu canto esquerdo o ID do mesmo, e no canto direito seu nome, e em casos de componentes, quando o mouse está sobre uma entrada ou saída do componente, também é exibido o index da entrada/saída.

5. Teste de validação e conclusão

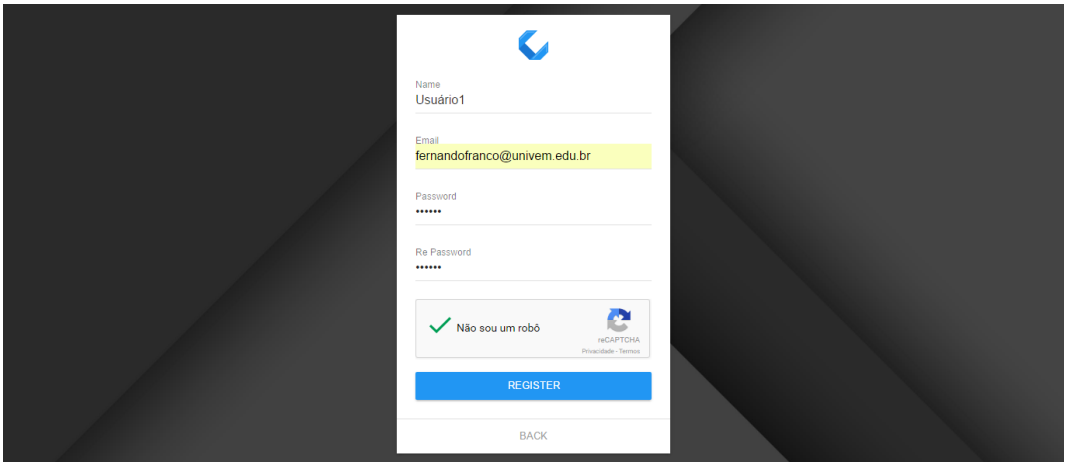
O teste de validação da ferramenta será realizado em um mesmo computador, onde para um teste de compartilhamento de projeto, serão necessários dois usuários, onde cada um será mantido em uma janela diferente do navegador. O teste seguirá os seguintes passos:

1. Cadastro do usuário1
2. Confirmação do castro do usuário1
3. Login do usuário1
4. Criação de um novo circuito que represente a porta AND2x1
5. Verificação de persistência do circuito
6. Simulação do circuito
7. Cadastro do usuári2
8. Confirmação do cadastro do usuário2
9. Login do usuário2
10. Adição de amizade do primeiro usuário com o segundo usuário
11. Compartilhamento do projeto
12. Edição do projeto simultaneamente com os dois usuários

5.1. Cadastro, confirmação de cadastro e login do usuário 1

O usuário1 será criado com o nome de Usuário1 e com o email: fernandofranco@univem.edu.br com a senha 123456 como mostra a figura 26.

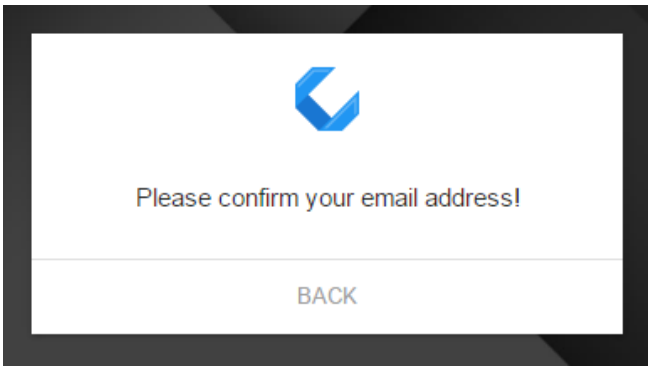
Figura 26 - Cadastro do usuário1



Fonte: Elaborada pelo autor

Uma mensagem pedindo para confirmar o email foi exibida após a solicitação do cadastro como mostra a figura 27.

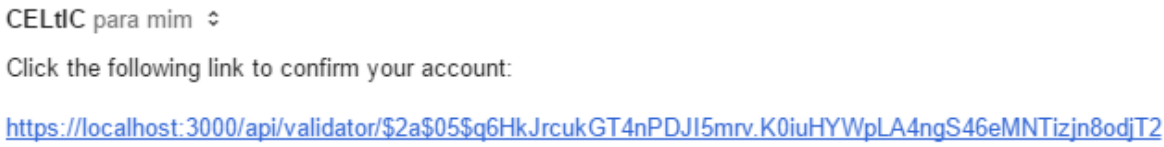
Figura 27 - Solicitação de confirmação de email



Fonte: Elaborada pelo autor

No email fornecido foi recebido um email com um link de confirmação do cadastro como é demostrado na figura 28.

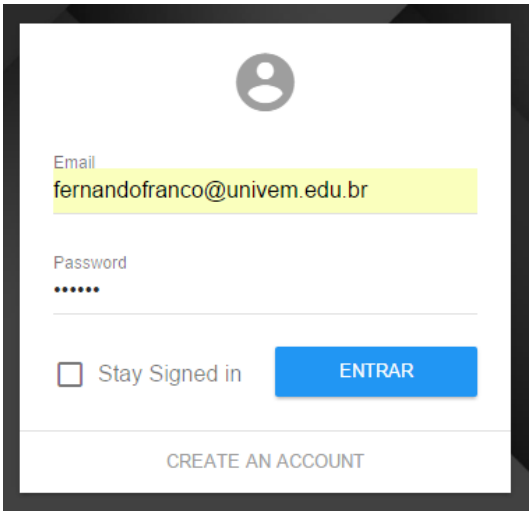
Figura 28 - Email com link de confirmação de cadastro



Fonte: Elaborada pelo autor

Após o clique no link de confirmação de cadastro, foi aberta uma nova aba no navegador com a página de login da ferramenta aberta. Então foi solicitado o login do usuário com os dados cadastrados, sendo o email fernandofranco@univem.edu.br e a senha 123456 como na figura 29.

Figura 29 - Login do usuário1

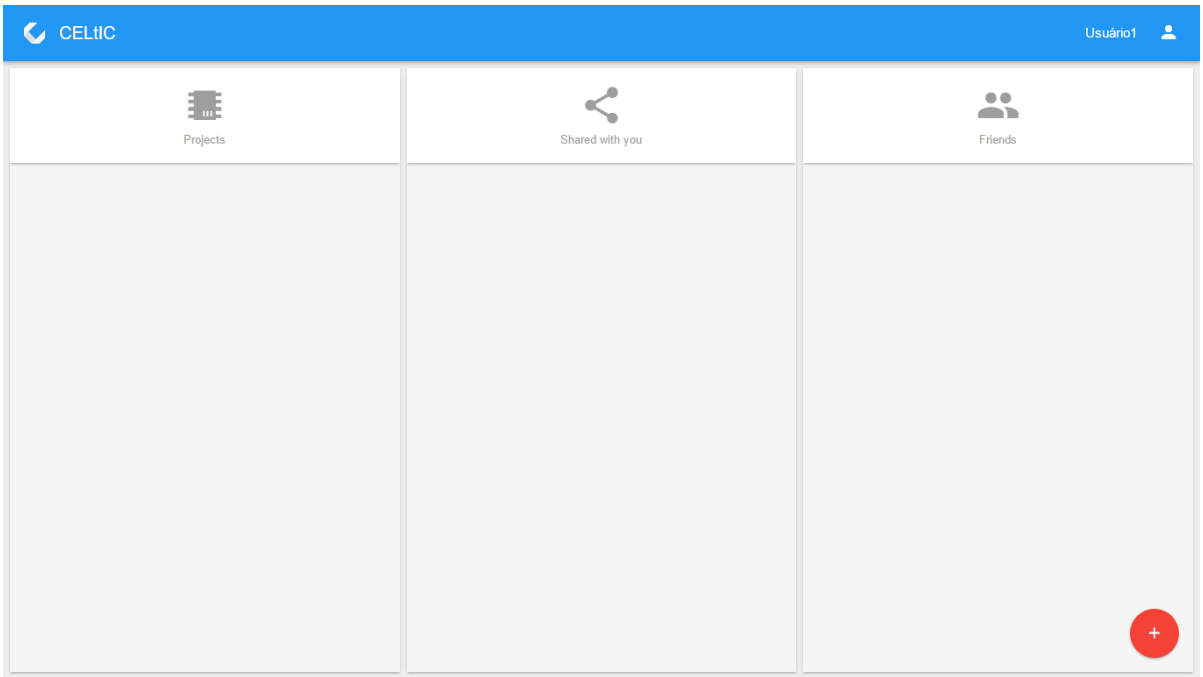


Fonte: Elaborada pelo autor

5.2. Criação do Projeto AND2x1

O dashboard foi exibido normalmente, sem itens em qualquer uma das listas como era previsto e exibido na figura 30.

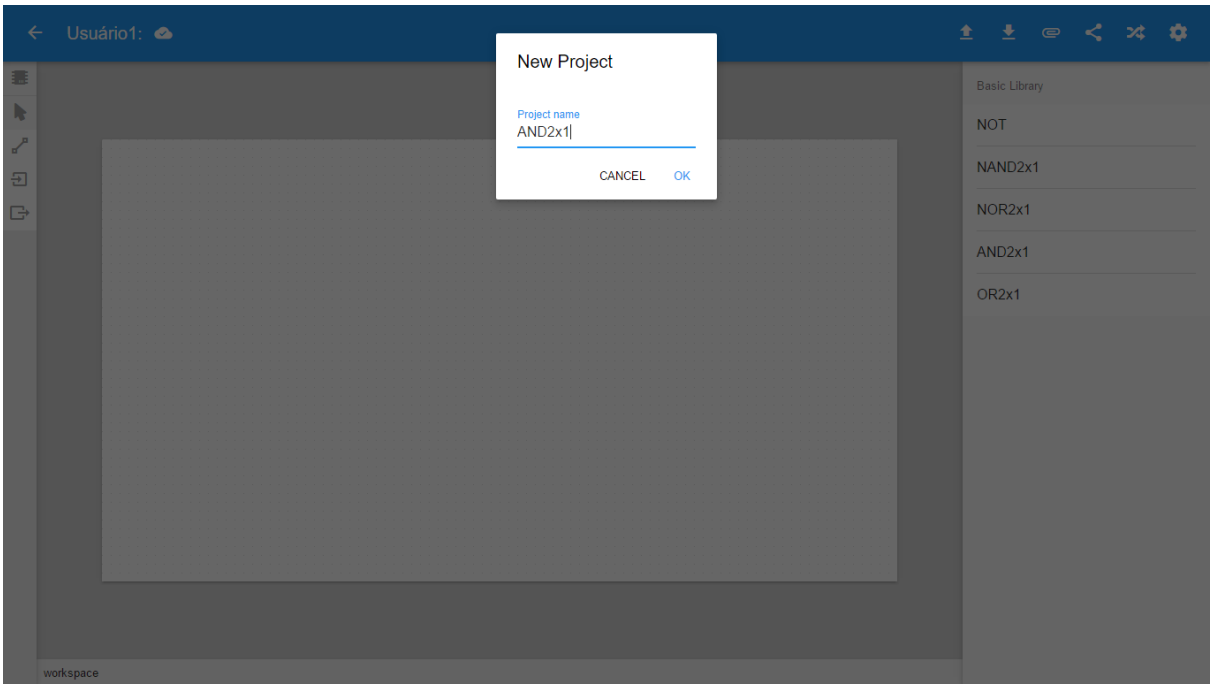
Figura 30 - Dashboard do usuário1



Fonte: Elaborada pelo autor

Então uma solicitação da criação de um novo projeto é enviada, e a aplicação é redirecionada para a ferramenta de criação e edição de circuitos lógicos digitais solicitando o nome do circuito que será criado como mostra a figura 31.

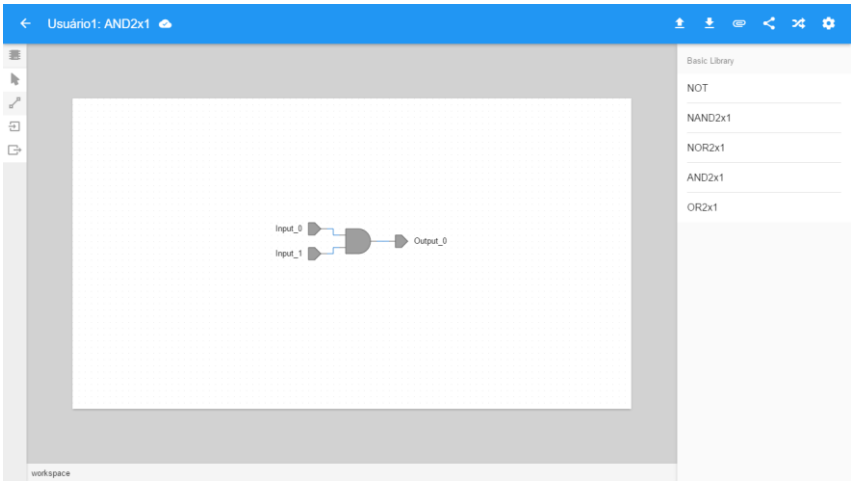
Figura 31 - Solicitação de nome do circuito



Fonte: Elaborada pelo autor

O nome foi fornecido e a aplicação já indicou um estado de persistência mostrando que o circuito está persistido na nuvem com sucesso. Então é adicionado uma porta AND2x1, junto com seus intpus e outpus, porem sendo os dois inputs adicionados pela ferramenta de inputs, sendo um diretamente no workspace e outro de forma dinâmica, já criando um fio de ligação entre o input e o componente. O output é adicionado com a ferramenta de cursor, sendo clicado com o botão direito em cima da saída do componente AND2x1 e selecionado a opção de "*Add Output*". Por fim a ferramenta de fio é utilizada para realizar a ligação entre o primeiro input adicionado ao workspace e o componente AND2x1 obtendo o resultado da figura 32.

Figura 32 - Adição de componente AND2x1 e suas entradas e saída



Fonte: Elaborada pelo autor

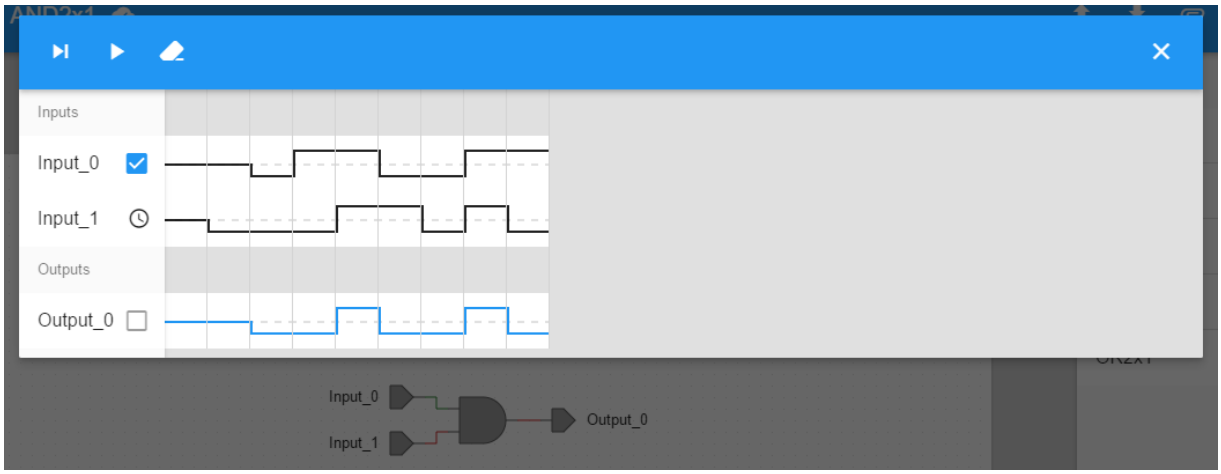
5.3.Simulação do Circuito AND2x1

Desta forma é então realizada a simulação do circuito, alterando os valores de entradas e utilizando os dois tipos de entradas, tanto como input (tipo manual) quando clock (tipo automático) com os passos realizados conforme a tabela 2, obtendo todos os resultados esperados, e a figura 33 mostrando todas as bordas se subida e descida da simulação.

Tabela 2 - Simulação de circuito AND2x1

Passo	Tipo de Entrada 0	Tipo de Entrada 1	Entrada 0	Entrada 1	Saída
1	Input	Input	-	-	-
2	Input	Input	-	0	-
3	Input	Input	0	0	0
4	Input	Input	1	0	0
5	Input	Input	1	1	1
6	Input	Clock	0	1	0
7	Input	Clock	0	0	0
8	Input	Clock	1	1	1
9	Input	Clock	1	0	0

Figura 33 - Simulação de circuito AND2x1

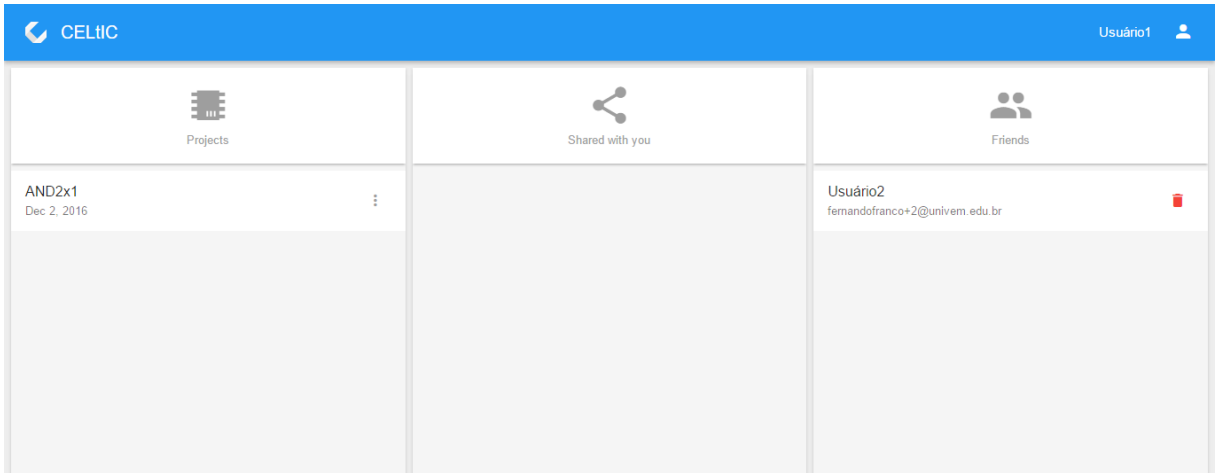


Fonte: Elaborada pelo autor

5.4. Compartilhamento de projeto

O cadastro, confirmação de cadastro e login do usuário2 assim como do usuário1 foram realizados com sucesso, porém utilizando o nome Usuário2 e o email fernandofranco+2@univem.edu.br, recebendo os mesmos dados, porem com um link de valores diferentes para a confirmação do cadastro. Portanto é então adicionado a amizade do usuário2 no usuário1 com sucesso, como mostra a figura 34.

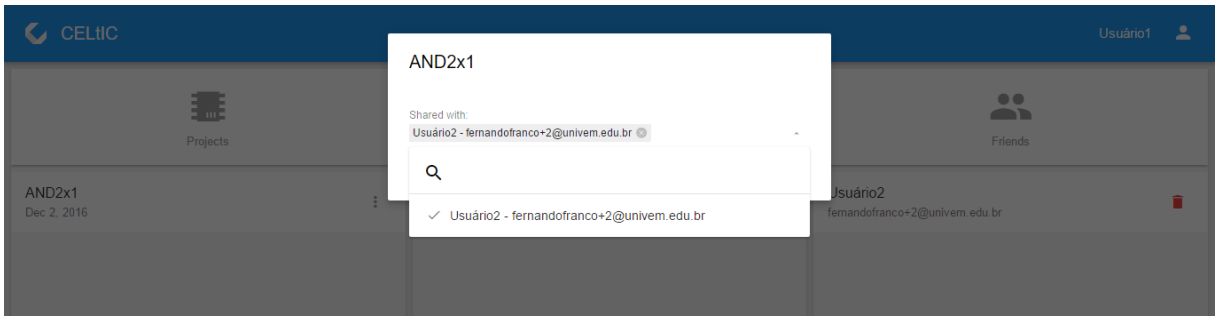
Figura 34 - Adição de amizade com do usuário2 no usuário1



Fonte: Elaborada pelo autor

O projeto então é compartilhado com o usuário2 para que o teste de sincronismo seja realizado, a figura 35 mostra a ação do compartilhamento.

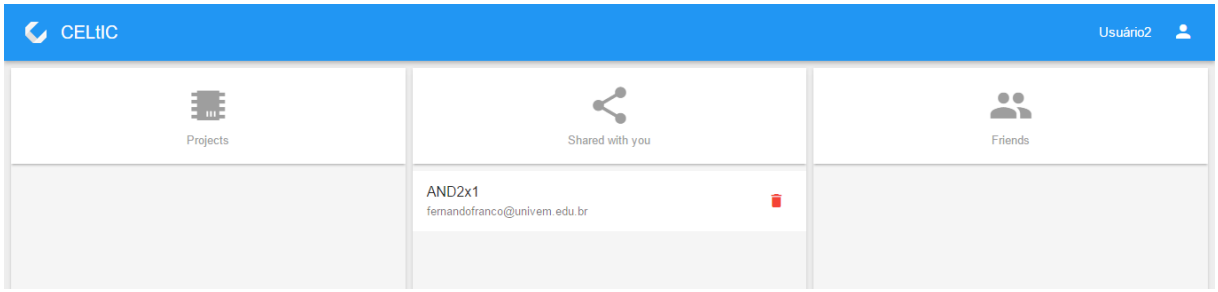
Figura 35 - Compartilhamento do circuito AND2x1 com o usuário 2



Fonte: Elaborada pelo autor

Com um refresh na página do dashboard do usuário 2, o projeto surge na lista de projetos compartilhados como é mostrado na figura 36.

Figura 36 - Projeto AND2x1 compartilhado com o usuário 2

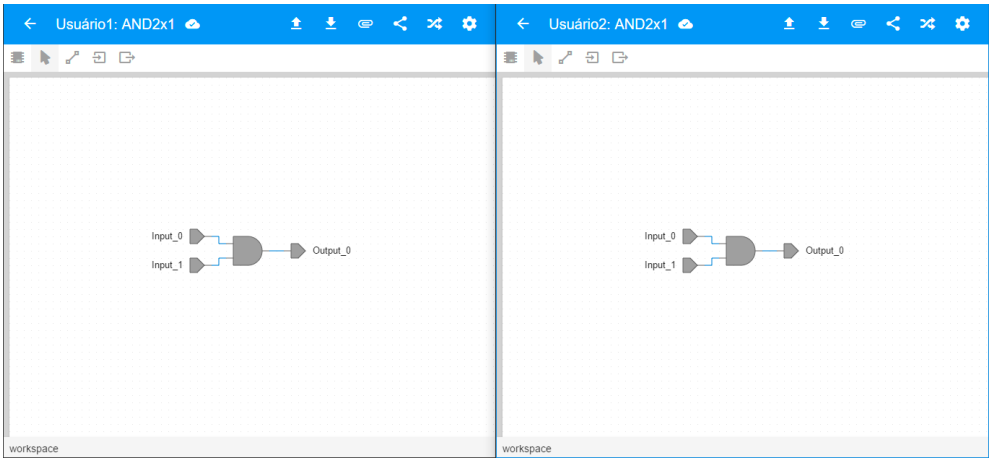


Fonte: Elaborada pelo autor

5.5.Edição síncrona do circuito

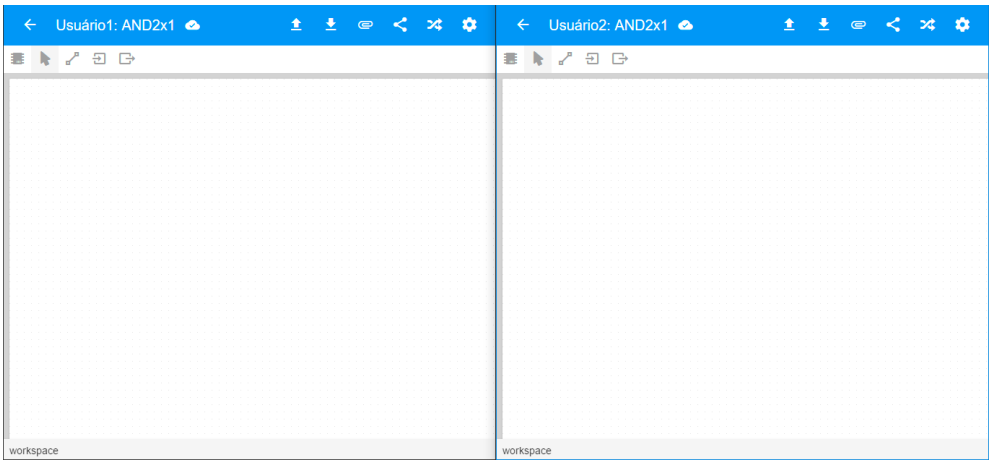
Então é aberto o projeto para que o mesmo possa ser editado em sincronismo entre os dois usuários, sendo que o teste realizado será a remoção do circuito atual e a adição do circuito OR2x1, que foi realizado com sucesso como é exibido nas figuras 37, 38 e 39, respectivamente sendo o projeto aberto no usuário2, a exclusão do circuito e a criação do circuito OR2x1, com as telas cortadas de cada usuários para que o circuito de cada um seja exibidos, sendo o usuário1 sempre a tela da esquerda e o usuário 2 sempre a tela da direita.

Figura 37 - Projeto aberto simultaneamente



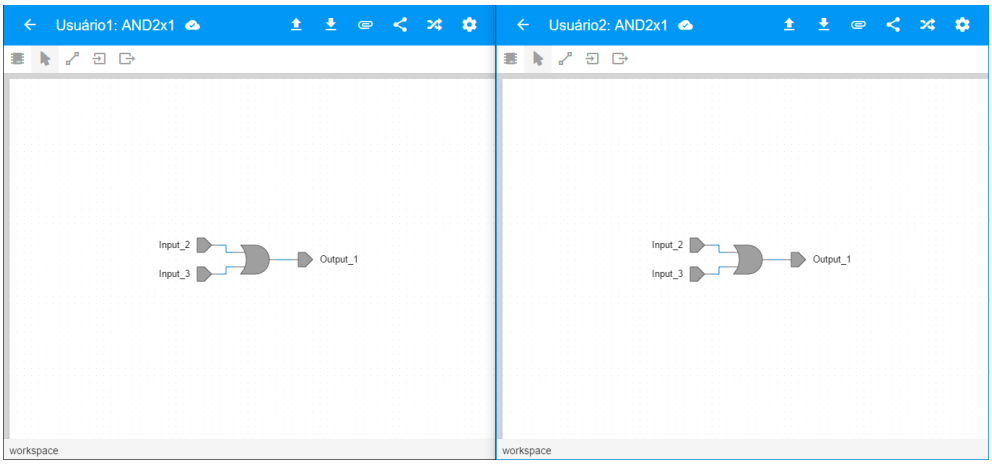
Fonte: Elaborada pelo autor

Figura 38 - Circuito excluído simultaneamente



Fonte: Elaborada pelo autor

Figura 39 - Criação do circuito OR2x1 simultaneamente



Fonte: Elaborada pelo autor

5.6. Conclusão

A proposta deste trabalho foi criar uma ferramenta didática para criação e simulação de circuitos lógicos digitais, sendo uma aplicação WEB voltada para a colaboração entre multiusuários.

Para o mesmo ser realizado, o projeto foi dividido em cinco partes, onde a primeira parte foi fazer o levantamento bibliográfico, pesquisa de trabalhos correlatos e tecnologias com termos como ferramenta didática, web socket, criação e simulação de circuitos lógicos digitais, entre outros.

A segunda parte foi a definição das tecnologias a serem utilizadas tanto do lado cliente quanto do lado servidor da aplicação, onde se destacam JavaScript, NodeJS, Express, Socket.io, AngujarJS, Canvas HTML5 e Lumx. Seguindo para a terceira parte, onde se definiu as estruturas a serem seguidas para o projeto, adotando as próprias estruturas indicadas pelos frameworks utilizados, com a adição de métodos próprios para agilizar processos.

A implementação da ferramenta sendo a quinta parte do desenvolvimento, utilizou de todas as tecnologias, para gerar um servidor com API de comunicação via REST, serviços de autenticação de usuários, persistência de dados, assim como também fornecer arquivos necessários para o download da ferramenta pelo sado cliente, e por fim, a comunicação via we-socket. Enquanto o lado cliente foi desenvolvido as páginas de cadastro e login que mantem usuários, dashboard que mantem projetos e amizades, e aplicação onde o usuário pode então realizar a criação, edição e simulação de circuitos lógicos digitais, utilizando de ferramentas e bibliotecas de componentes escritos em JSON, podendo importar e exportar projetos assim como compartilha-los para uma edição simultânea do mesmo.

Por fim, a quinta parte consiste em testar e validar a ferramenta criada, onde todos os testes realizados, passam sem ocasionar problemas ou erros durante o mesmo.

Porem a ferramenta necessita de um servidor para que a mesma possa ser executada, o que pode trazer futuros desconforto para usuários da mesma, pois sendo essa uma ferramenta gratuita, não possui um servidor próprio para que a mesma seja fornecida de forma gratuita para seus usuários.

Portanto a ferramenta proposta foi criada, e executada com sucesso, onde alguns itens que poderão ser adicionados em trabalhos futuros como melhoria na relação de usuários e amigos, assim como chat para comunicação entre os mesmos durante uma edição colaborativa de um projeto, e também a adição de criação de novos componentes e bibliotecas com os

circuitos criados na mesma para que assim a ferramenta seja mais robusta e intuitiva ao usuário, possibilitando.

Referências Bibliográficas

KAYAL, M.; STEFANOVIC, D.; PASTRE, M. CMOS Analog Circuits Design Educational Tool. In: **Microelectronics Education**. Dordrecht: Springer Netherlands, 2004. p. 133–138.

KIM, D. et al. Development of a Web-Based Educational Java Applet for Understanding Concepts and Principles of Digital Logic Circuits. In: [s.l: s.n.]. p. 135–150.

LUCENA, P. C. S. DE. Desenvolvimento de uma Ferramenta Computacional para Modelagem e Simulação de Circuitos Eletônicos Digitais. p. 153, 2013.

LUMAPPS. **Lumx**. Disponível em: <http://ui.lumapps.com/>. Acesso em: 1 dez. 2016.

Material Design. Disponível em: <https://material.google.com/material-design/environment.html>. Acesso em: 1 dez. 2016.

MORAIS, R. X. T. DE. Software Educacional: a Importância De Sua Avaliação E Do Seu Uso Nas Salas De Aula. **Flf.Edu.Br**, p. 1–52, 2003.

NPM. **NPM**. Disponível em: <https://www.npmjs.com/>. Acesso em: 1 dez. 2016.

Sequelize. Disponível em: <http://docs.sequelizejs.com/en/v3/>. Acesso em: 26 nov. 2016.

Socket.io. Disponível em: <http://socket.io/>. Acesso em: 1 dez. 2016.

TANENBAUM, A. S. **Organizacao Estruturada de Computadores**. 5. ed. [s.l.] Pearson Education - Br, 2008.

W3C. **Protocols**. Disponível em: <https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>. Acesso em: 30 nov. 2016a.

W3C. **HTML Canvas 2D Context**. Disponível em: <https://www.w3.org/TR/2dcontext/>. Acesso em: 1 dez. 2016b.