

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

ALISSON CHAGAS SILVA

Ambiente virtual para aplicação do projeto LoBoGames

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Renato Perez Ribas

Porto Alegre
2021

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitor: Prof. Carlos André Bulhões Mendes

Vice-Reitora: Prof^a Patrícia Helena Lucas Pranke

Pró-Reitoria de Ensino (Graduação e Pós-Graduação): Prof^a Cíntia Inês Boll

Diretora do Instituto de Informática: Prof^a Carla Maria Dal Sasso Freitas

Coordenador do Curso de Ciência da Computação: Prof. Rodrigo Machado

Bibliotecária-Chefe do Instituto de Informática: Beatriz Regina Bastos Haro

AGRADECIMENTOS

Primeiramente, gostaria de agradecer à UFRGS, professores, técnicos administrativos e terceirizados, por proporcionarem um ambiente de ensino, inovação e pesquisa que são excelência no Brasil e no mundo. Durante todos esses anos de graduação, a universidade foi o lugar que eu mais frequentei, que eu mais fiz amigos e que me fez crescer como profissional e como pessoa.

Gostaria de deixar um agradecimento mais que especial para os meus pais, Nilda Chagas Silva e José Carlos dos Santos Silva, jamais vou esquecer o esforço que vocês fizeram para pagar a minha inscrição no vestibular e proporcionar que eu pudesse me concentrar somente nos estudos, sei do privilégio que eu tenho por ter tido esse apoio. Se um dia eu for metade do ser humano que vocês são, eu já serei uma pessoa realizada.

Não posso deixar de citar aqui o meu irmão, Jocimar Chagas Silva, que sempre foi um espelho pra mim e um exemplo a ser seguido, minha namorada que sempre me deu apoio nessa trajetória e a todos os amigos e amigas que fizeram parte dessa minha jornada.

Ao meu orientador, Renato Perez Ribas, o meu muito obrigado, desenvolver esse trabalho foi um dos momentos mais gratificantes da minha vida, espero que esse projeto não acabe aqui e que possamos tornar o ambiente de ensino mais leve, prazeroso e altamente tecnológico.

RESUMO

O presente trabalho de conclusão do curso de Ciência da Computação tem como objetivo propor um ambiente virtual colaborativo para o ensino de jogos lógicos de tabuleiro, tendo como base o Programa de Extensão da UFRGS “Jogos Lógicos de Tabuleiro” (LoBoGames). O ambiente proposto aqui visa trazer para o mundo virtual todos os benefícios da gamificação do ensino que são obtidos no Projeto LoBoGames. Para isso, foi feita uma análise de quais são os elementos fundamentais de um ambiente virtual. Além disso, foi feito um estudo dos jogos lógicos de tabuleiro e a sua aplicação no ensino. Com isso, uma proposta de plataforma foi feita e, como resultado dessa proposta, foi desenvolvido um mínimo produto viável para que a viabilidade do sistema fosse discutida, tomando como base aplicações cliente-servidor que fazem trocas constantes de mensagens entre os clientes, como Mconf e Google Meet. O resultado desse mínimo produto viável mostrou que é possível ter uma plataforma que consiga mesclar os principais elementos de ambientes virtuais voltados para jogos com os ambientes virtuais de aprendizagem, trazendo assim uma alternativa de ensino para esse momento de distanciamento social.

Palavras-chave: Ambientes virtuais colaborativos. Jogos lógicos de tabuleiro. Plataformas virtuais de aprendizagem.

Virtual environment for application to the LoBoGames project

ABSTRACT

The present work of completion of the Computer Science course aims to propose a collaborative virtual environment for the teaching of logic board games, based on the UFRGS Extension Program “Jogos Lógicos de Board” (LoBoGames). The environment proposed here aims to bring to the virtual world all the benefits of the gamification of teaching that are obtained through the LoBoGames Project. For this, an analysis was made of what are the fundamental elements of a virtual environment. In addition, a study of the games was carried out and their adoption in the teaching process. With that, a platform has been proposed, and as a result of this proposal, a minimum viable product has been developed so that the system viability is evaluated based on client-server applications that constantly exchange messages between clients, such as Mconf and Google Meet. The result of this minimum viable product showed that it is possible to have a platform that can mix the main elements of virtual environments geared towards games, thus bringing an alternative to teaching for this moment of social distance.

Keywords: Collaborative virtual environments. Logic board games. Virtual learning platforms.

LISTA DE FIGURAS

Figura 3.1 – Módulos propostos pelo projeto LoBoGames	18
Figura 4.1 – Protótipo do ambiente virtual	26
Figura 5.1 – Tela de cadastro no MVP	31
Figura 5.2 – Painel do professor no MVP	31
Figura 5.3 – Imagem de uma sala de jogos	32
Figura 5.4 – Imagem de uma sala de jogos com chamada de vídeo	33
Figura 5.5 – Arquitetura geral do sistema	35
Figura 5.6 – Arquitetura servidor	38
Figura 5.7 – Canal de comunicação fornecido pela biblioteca socket.io	41
Figura 5.8 – Funcionamento das <i>Rooms</i>	42
Figura 5.9 – Comunicação entre cliente, servidor e <i>Peers</i> para transmissão de vídeo	46

LISTA DE TABELAS

Tabela 2.1 - Categorização de ambientes virtuais colaborativos	13
Tabela 5.1 - Lista de eventos que o servidor espera receber	42
Tabela 5.2 - Lista de eventos que o cliente espera receber	45

LISTA DE ABREVIATURAS E SIGLAS

API	Interface de programação de aplicações (sigla do inglês <i>application programming interface</i>)
AVA	Ambiente virtual de aprendizagem
CVE	Ambiente virtual colaborativo (sigla do inglês <i>cooperative virtual environments</i>)
EaD	Ensino à distância
HTTP	Protocolo de transferência de hipertexto (sigla do inglês <i>hypertext transfer protocol</i>)
IP	Protocolo de internet (sigla do inglês <i>Internet protocol</i>)
LGPD	Lei Geral de Proteção de Dados
MVC	<i>Model view controller</i>
MVP	Mínimo produto viável (sigla do inglês <i>minimum viable product</i>)
RPG	Jogo de interpretação (sigla do inglês <i>role-playing game</i>)
SPA	Aplicação de página única (sigla do inglês <i>single page application</i>)
SSR	Renderização do lado do servidor (sigla do inglês <i>server side rendering</i>)
TCP	Protocolo de controle de transmissão (sigla do inglês <i>transmission control protocol</i>)
TDD	Desenvolvimento guiado a testes (sigla do inglês <i>test driven development</i>)
LoBoGames	Jogos lógicos de tabuleiro (sigla do inglês <i>logic board games</i>)
HTML	Linguagem de Marcação de Hipertexto (sigla do inglês <i>HyperText Markup Language</i>)
WebRTC	Comunicação em tempo real na internet (sigla do inglês <i>web real time communications</i>)

SUMÁRIO

1 INTRODUÇÃO	9
2 AMBIENTES VIRTUAIS COLABORATIVOS	11
2.1 O que são ambientes virtuais colaborativos?	11
2.2 Elementos de um CVE	12
2.3 Categorização dos ambientes virtuais colaborativos	13
2.4 Ambientes virtuais colaborativos aplicados a jogos	14
2.5 Ambientes virtuais colaborativos aplicados à educação (AVA)	15
3 JOGOS LÓGICOS DE TABULEIRO	17
3.1 Jogos lógicos de tabuleiro aplicados à educação	17
3.1.1 Programa de extensão LoBoGames	18
3.2 Jogos de tabuleiro e ambientes virtuais aplicados como instrumento pedagógico	19
4 PROPOSTA	21
4.1 Funcionalidades propostas para o sistema	22
4.1.1 Modo Professor	23
4.1.2 Modo Aluno	27
4.2 Inovações e desafios que o ambiente virtual proporciona	28
5 MVP - IMPLEMENTANDO ALGUMAS FEATURES DO SISTEMA	30
5.1 Arquitetura geral do MVP	33
5.2 Servidor	36
5.2.1 Cadastro de professores	39
5.2.2 Login de usuários	39
5.2.3 Cadastro de salas	39
5.2.4 Sala de jogos	40
5.3 Cliente	43
5.4 Principais desafios no processo de desenvolvimento do MVP	46
6 CONCLUSÃO	48
REFERÊNCIAS	49

1 INTRODUÇÃO

No mundo em que vivemos é cada vez mais comum passarmos grande parte do nosso tempo em ambientes virtuais, seja visualizando *posts* da redes sociais ou assistindo a uma aula *online*. Cada vez mais esses tipos de ambientes fazem parte das nossas vidas, fazendo atividades tradicionais, como assistir a uma aula presencial, ganharem uma nova perspectiva. Se fizermos um paralelo do avanço tecnológico que tivemos nos últimos 30 anos com o momento de pandemia em que vivemos, podemos considerar que se não fossem essas tecnologias a nossa situação estaria muito pior. Isso porque, nos dias de hoje, basta termos acesso à Internet que conseguimos trabalhar, estudar, interagir com os amigos e se divertir. Se pensarmos na educação, que é um elemento básico para o crescimento de qualquer civilização, esses ambientes permitem que os danos de um momento de pandemia sejam amenizados.

Os jogos de tabuleiro são elementos que estão presentes na humanidade desde as civilizações mais antigas, como forma de diversão mas também como forma de interação e evolução da sociedade como um todo. Considerando os jogos lógicos de tabuleiro aplicados à educação, é possível perceber que a sua utilização como parte do ensino pode ser um grande aliado no desenvolvimento do raciocínio lógico e na resolução de problemas. Os jogos de damas e xadrez, por exemplo, podem ser usados de diversas formas, auxiliando o aluno a desenvolver diversas habilidades. Visando trazer para o ensino tradicional uma nova forma de desenvolvimento dos alunos, o Programa de Extensão “Jogos Lógicos de Tabuleiro” (LoBoGames) traz uma metodologia própria para utilização desses jogos como instrumento pedagógico (LOBOGAMES, 2019).

Quando fazemos um paralelo entre ambientes virtuais aplicados à educação e jogos lógicos de tabuleiro, percebemos que existem poucas plataformas que possibilitam ao professor aplicar as metodologias propostas pelo projeto LoBoGames. Essa necessidade de um ambiente virtual que não esteja vinculado a um jogo específico, mas que disponha de diversos tabuleiros onde vários jogos possam ser aplicados, tornou-se ainda maior a partir do ano de 2020, quando a pandemia do coronavírus fez as escolas ficarem fechadas.

O objetivo deste trabalho é propor um ambiente virtual voltado para a educação que disponha de diversas métricas para análise de desempenho dos alunos e que permita que um professor possa aplicar a metodologia LoBoGames também no ensino remoto.

No próximo capítulo é apresentada uma discussão sobre ambientes virtuais colaborativos, visto que é muito importante entender quais são os elementos básicos para a

construção de um ambiente que seja fácil de utilizar, que dá a sensação de pertencimento para os usuários e que tenha na interação uma forma de manter o engajamento. Já no Capítulo 3 é feito um estudo histórico da importância dos jogos de tabuleiro para a sociedade e, a partir disso, o projeto de extensão LoBoGames é explicado. O Capítulo 4 é onde a plataforma é idealizada e todas as suas características são detalhadas e explicadas e, por fim, no Capítulo 5 temos a explicação da implementação do mínimo produto viável, onde os aspectos técnicos e a viabilidade do sistema são discutidos.

2 AMBIENTES VIRTUAIS COLABORATIVOS

No mundo globalizado em que vivemos os ambientes virtuais são de suma importância, tanto para compartilhamento de informações, trabalho em equipe e interação entre pessoas distantes (BRITO, 2004). Nos dias de hoje, temos acesso a várias plataformas que possibilitam essa interação, mas nem sempre foi assim.

Apesar de ser um fator extremamente relevante para a evolução da sociedade como um todo, antigamente essa interação só era possível se dois ou mais indivíduos estivessem presencialmente no mesmo local. Porém, com a constante evolução da tecnologia, saímos de um mundo onde as trocas, tanto de informações como de conhecimento, eram limitadas, para um mundo onde conseguimos trabalhar, estudar, fazer amizades e se relacionar de forma virtual. Segundo Machado (apud PELLANDA, 2000), estamos em uma nova cultura, chamada de cibercultura, que é de suma importância na construção da sociedade e dos sujeitos. Essa nova cultura é baseada na presença de tecnologia existente na rede de computadores, o que traz novas possibilidades de ver, perceber, sentir e vivenciar o mundo.

2.1 O que são ambientes virtuais colaborativos?

Ambientes virtuais colaborativos, ou CVEs (*collaborative virtual environment*), são uma simulação do mundo real onde os participantes podem conversar, colaborar, se divertir e compartilhar informação, tudo através desse espaço de interação, sem necessariamente estarem juntos.

Em uma visão mais abrangente, os CVEs são caracterizados como um espaço compartilhado, 3D ou 2D, baseado em texto ou vídeo, onde pessoas interagem de forma distribuída através de agentes e objetos virtuais (RAPOSO, 2011). Com essa definição, conseguimos dizer que tanto as redes sociais quanto os jogos de RPG são uma forma de CVE.

Nos ambientes virtuais colaborativos, um conceito conhecido na literatura sobre o assunto é o ciberespaço. Segundo Gouveia (apud TOMAS, 1991), ciberespaço é um ambiente de trabalho pós industrial baseado em uma nova interface proporcionada pelas redes de comunicações que possibilita o acesso direto a mundos paralelos potencialmente utilizados como espaços de trabalho. Com essa definição, podemos caracterizar um CVE como um ponto de encontro no ciberespaço onde as pessoas se juntam para realizar alguma atividade em comum.

Desde o início dos anos 1990, esse tipo de ferramenta vem crescendo e atendendo a

todo tipo de usuário, desde pessoas mais jovens, como os jogadores de RPG, até pessoas mais velhas com estudantes de ensino à distância (EaD).

Segundo Fucks (2002), um paradigma aplicável à área de CVE, principalmente aqueles aplicados à educação, é o da comunicação, cooperação e coordenação. Esses elementos juntos conseguem trazer a percepção de que estamos realizando alguma tarefa de forma colaborativa. A cooperação pode ser vista como o ato de dois ou mais indivíduos estarem atuando simultaneamente em determinado ambiente. É importante notar que não necessariamente deve haver dependência entre as ações dos usuários.

A comunicação é vista como a troca de informações entre os usuários de um determinado ambiente. Isso pode se dar através de *chat*, áudio, vídeo ou correio eletrônico. Já a coordenação está ligada à forma como o sistema vai gerenciar as ações executadas pelos integrantes.

Todos esses conceitos somados nos dão a dimensão do potencial que esses tipos de ambientes têm para o nosso mundo globalizado. Neste momento de isolamento social, esse tipo de ambiente encurta a distância entre as pessoas fazendo com que nós, seres humanos, tenhamos o mínimo de interação.

É de suma importância que, ao criar um CVE, os desenvolvedores tenham plena noção de quais são os elementos fundamentais que esse tipo de ambiente deva ter para que possa cumprir o seu papel.

2.2 Elementos de um CVE

Levando em consideração o paradigma aplicável aos CVEs, discutido na Seção 2.1, podemos dizer que existem alguns elementos fundamentais que estão ligados a esse tipo de ambiente.

O primeiro deles é o mundo virtual que é definido como um espaço imaginário, manifestado através de imagens geradas por computador (RAPOSO, 2011). Além disso, outro ponto importante sobre o mundo virtual são as regras que o governam, ou seja, quais são as regras dos objetos que compõem o mundo virtual. Por exemplo, em muitos jogos de computador os personagens são representados por avatares e estes realizam suas ações de acordo com as regras determinadas pelo jogo.

O segundo elemento fundamental de um CVE é a interatividade, pois é ela que possibilita a troca de experiências entre os usuários. Se não existisse a interatividade, estar em um ambiente virtual seria como assistir a um filme animado (RAPOSO, 2011).

O terceiro elemento são os avatares, que são os objetos virtuais responsáveis por representar um participante. Avatares são de suma importância para dar a sensação de imersão dentro do ambiente. Se existimos no mundo real, no mundo virtual nós precisamos de algo que nos distingue de outros participantes, e isso acontece com os avatares. Juntando esses três elementos, os CVEs conseguem nos dar a sensação de completa imersão, tanto física quanto mental.

Outro elemento importante de um CVE é o canal de comunicação. Seria muito frustrante estarmos em um ambiente que nos fornece toda essa imersão se não fosse possível interagir através de mensagem escrita ou por voz com outros participantes.

2.3 Categorização dos ambientes virtuais colaborativos

Conforme descrito acima, existem várias características que compõem um CVE. Com isso, temos várias plataformas que se enquadram nesse conceito, mas nem todas servem ao mesmo propósito. Sendo assim, podemos categorizar os ambientes virtuais segundo a sua aplicação, ou seja, ambientes aplicados ao ensino, aos jogos, às redes sociais e ao trabalho colaborativo.

Para traçar um paralelo entre as características que cada categoria implementa, apresentamos na Tabela 2.1 alguns exemplos de plataformas, por categoria:

Tabela 2.1 - Categorização de ambientes virtuais colaborativos.

Categorias	Exemplos de plataformas
CVEs aplicados aos jogos	League of Legends (https://na.leagueoflegends.com/)
CVEs aplicados ao ensino	Mconf (https://mconf.com/)
CVEs aplicados às redes sociais	Instagram (https://www.instagram.com/)
CVEs aplicados ao trabalho colaborativo	Miro (https://miro.com/)

Vale ressaltar que muitas plataformas não se encaixam somente em uma categoria, podendo ser usada para diferentes propósitos. Com a categorização apresentada na Tabela 2.1 podemos obter algumas características dos diferentes tipos de ambientes. Nos CVEs aplicados aos jogos a representação do ambiente é de suma importância, por isso, os avatares e objetos em tela estão sendo cada vez mais reais. Já os CVEs aplicados ao ensino têm na interatividade a sua principal característica, conversas por *chat*, vídeo ou fóruns de discussão e o compartilhamento de documentos e de videoaulas são muito importantes nesse tipo de ambiente. Em ambientes aplicados às redes sociais a imersão mental através do engajamento é o mais importante, pois assim conseguimos ter conexões de pensamentos e ideias. Por fim,

nos ambientes virtuais aplicados ao trabalho colaborativo, é de suma importância ter um espaço *real-time* (i.e., em tempo real) onde os participantes possam manipular os objetos do ambiente.

Mas será que é possível unir o melhor desses mundos, ou seja, ter as principais características de dois ou mais tipos de CVEs para oferecer um ambiente mais flexível e atrativo?

2.4 Ambientes virtuais colaborativos aplicados a jogos

Segundo Raposo (2011), o primeiro CVE a se ter notícia foi criado pela NASA em 1974. Esse era um jogo em primeira pessoa que tinha vários dos elementos citados na Seção 2.2, como comunicação por mensagens instantâneas e navegação em primeira pessoa.

Apesar de hoje em dia os CVEs estarem sendo aplicados em diversas áreas, é notável que a sua popularização se deu através de ambientes de entretenimento, e as aplicações mais populares desses tipos de ambientes são os jogos.

Em 1993 o jogo *Doom* foi lançado e pela primeira vez o grande público teve contato com a implementação de um CVE. Um número grande de cópias foram distribuídas, fazendo com que os administradores de rede fossem pegos de surpresa com o grande número de pacotes que eram transmitidos pelo jogo. No início da década de 2000, vários jogos foram lançados, o mais popular deles foi o *Counter Strike*. O jogo em rede mais bem sucedido é o *World of Warcraft* - WOW (RAPOSO, 2011). O WOW está na categoria de jogos MMORPG que significa jogo de interpretação *online* e massivo para múltiplos jogadores. Essa categoria de jogos cresceu muito nos últimos 20 anos. Nesses tipos de jogos, os jogadores criam seus personagens e interagem com o mundo virtual simultaneamente.

Quando falamos de ambientes aplicados aos jogos, uma das principais características é a representação do ambiente virtual, porém, hoje em dia, esse tipo de característica tem se tornado cada vez mais irrelevante visto que a qualidade gráfica está em um nível jamais visto. Uma das características que faz esse tipo de CVE não ter uma queda no seu número de usuários é o fato de que cada vez mais os jogos estão possibilitando a imersão completa através de um dos pilares dos ambientes virtuais colaborativos que é a colaboração. No *World of Warcraft*, por exemplo, os jogadores são divididos em “reinos” (servidores na rede) para poder jogar em conjunto. Os jogadores recebem missões (por exemplo, ir a determinado lugar e matar determinado monstro), mas também podem conversar com amigos, exercer sua

profissão virtual, realizar treinamentos, trocas, entre outros. Esse tipo de vivência que os jogos trazem faz com que nós consigamos lidar com situações obscuras do nosso mundo real.

Outro tipo de jogo que pode ser enquadrado como um CVE é o xadrez. Hoje em dia existem vários ambientes que possibilitam jogar e até fazer torneios, como o *Chess* (<https://www.chess.com/>) e o *Litchess* (<https://lichess.org/>). Nessas plataformas, a interação com os objetos do jogo é fácil e intuitiva.

2.5 Ambientes virtuais de aprendizagem (AVA)

Um tipo de aplicação dos CVEs que vem ganhando muito destaque nos últimos anos são as aplicações voltadas para a educação. De fato, em um mundo tão globalizado, é fundamental a existência de tecnologias que possibilitem o aprendizado. Outro argumento que torna essas ferramentas essenciais é que o fechamento das escolas, por conta do distanciamento social, traz um prejuízo enorme para a correta formação das crianças.

Ambiente virtual de aprendizagem (AVA) nada mais é do que um ambiente que visa apoiar atividades de educação à distância (EaD), onde a principal característica é ter um conjunto de ferramentas que auxiliam a comunicação e aprendizagem dos participantes. A plataforma Moodle (<https://moodle.org/>) é um exemplo de AVA bastante utilizado.

Analisando esses ambientes, podemos dizer que o uso de CVEs na educação oferece diversas vantagens, tais como a possibilidade de interação entre computador e aluno, atenção individualizada e possibilidade do aluno ir avançando no conteúdo no seu tempo. Cada indivíduo tem um ritmo de aprendizagem e isso é respeitado nos AVAs.

Para tornar tudo isso possível, os ambientes virtuais de aprendizado reúnem diversas tecnologias para prover comunicação, administração e disponibilização de materiais. As tecnologias de comunicação são importantes para os alunos se comunicarem com os professores e com outros alunos. Isso pode ser feito com fóruns de discussão, bate-papo e videoconferência.

Para tornar esses ambientes mais atrativos para os alunos, existe um conceito na literatura que é a “gamificação” do ensino. Por mais que o uso da tecnologia no aprendizado traga novas possibilidades, os AVAs ainda pecam em disponibilizar a sensação de pertencimento ao mundo virtual e os jogos colaborativos, pelo fato de criarem uma cena que representa o mundo virtual, conseguem dar essa sensação de imersão. Então por que não juntar o divertimento dos jogos com o aprendizado? Foi assim que surgiu a ideia da

gamificação do ensino. Hoje em dia, diversas plataformas possibilitam o aprendizado de algo através da diversão dos jogos, através do lúdico.

Neste momento de distanciamento social, os ambientes virtuais para o ensino tornaram-se ferramenta obrigatória para a escolarização. Nunca o conceito de ensino à distância esteve tão presente na nossa sociedade. Porém, quando falamos do exercício lógico, pensamento rápido e fatores ligados à capacidade do aluno raciocinar, torna-se difícil ter um ambiente virtual que traga essa experiência de forma completa. Existem projetos que visam o ensino de jogos lógicos de tabuleiro para que os alunos desenvolvam um rápido raciocínio. Mas como continuar com esses projetos no mundo virtual? Por isso, é de suma importância termos um ambiente virtual que possibilite aos alunos desenvolverem o seu raciocínio lógico tendo uma experiência parecida com a presencial.

Tendo em vista esse momento de pandemia, desde a Segunda Guerra Mundial não tínhamos esse cenário com milhares de escolas fechadas e alunos ficando somente em casa. Diante disso, fica evidente o efeito devastador que o distanciamento social traz para o ensino. Porém, temos a oportunidade de repensar a forma como as disciplinas são aplicadas, pois o ensino à distância será cada vez mais uma realidade, não só no ensino universitário mas também no ensino fundamental e médio.

3 JOGOS LÓGICOS DE TABULEIRO

Segundo Huizinga (2010), o jogo faz parte da cultura humana e da evolução da sociedade, sendo assim os jogos são de suma importância para nós como forma de aprendizado e interação.

Desde o Egito antigo os jogos de tabuleiro fazem parte da sociedade, muitas vezes com fins recreativos, mas também para fins religiosos, esses tipos de jogos apresentam princípios parecidos. Os jogos mais antigos que se tem registro são o Jogo-de-Ur, que surgiu na região da Mesopotâmia, e o Senet, que surgiu no Egito antigo. Mesmo esses jogos tendo surgido em localidades e tempos diferentes o princípio é o mesmo, em ambos o vencedor é aquele que chega no final do tabuleiro. Isso reforça o conceito de que o jogo faz parte da cultura humana.

Recentemente, os jogos de tabuleiro contextualizados vêm ganhando muita notoriedade pelo mundo. Esses tipos de jogos normalmente apresentam um cenário, personagens e um roteiro de movimentos bem específicos. Um exemplo clássico desse tipo de jogo é o Banco imobiliário. Quando tentamos definir jogos de tabuleiro, temos que separá-los em diferentes categorias. Por exemplo, o Detetive pertence à categoria de jogos de tabuleiro contextualizados, já o Jogo-da-Velha (ou *Tic-Tac-Toe*) pode ser definido como um jogo abstrato de estratégia. Outra distinção que existe entre os diferentes tipos de jogos de tabuleiro é quando estes são aplicados à educação. Normalmente os jogos abstratos de estratégia são utilizados para ensino de Matemática, já os jogos contextualizados são utilizados para ensino de História e Geografia (RIBAS, 2018).

3.1 Jogos lógicos de tabuleiro aplicados à educação

Ao longo dos anos, as estratégias de ensino utilizadas pelos professores vêm sendo criticadas, pois baseiam-se fundamentalmente na repetição e cópia, deixando assim uma enorme lacuna na potencialização da autonomia e curiosidade dos alunos (PACHECO, 2014). É fundamental encontrar outras opções que possam fazer o desenvolvimento do aluno ser ainda mais potencializado. A gamificação do ensino tem sido uma alternativa bastante interessante para deixar o ambiente escolar mais atraente e desafiador, essa estratégia de ensino é baseada na utilização de elementos dos jogos, tais como desafios, prêmios e competição. Esse tipo de abordagem oferece inúmeras vantagens no processo de ensino, principalmente para trazer um maior engajamento, que é fundamental quando falamos do

ensino remoto.

3.1.1 Programa de extensão LoBoGames

O programa de extensão Jogos Lógicos de Tabuleiro desenvolvido na UFRGS tem como objetivo utilizar os vários jogos abstratos de estratégia como instrumento pedagógico (RIBAS et al., 2018)(LOBOGAMES, 2019). Nesse projeto, cerca de cem jogos são utilizados e agrupados segundo o seu funcionamento, conforme mostra a Figura 3.1.

Figura 3.1 – Módulos propostos pelo projeto LoBoGames.

Módulos	Princípio básico	Exemplos
Jogos de bloqueio	Bloquear as peças adversárias	Pong hau k'i, Madelinette, Mu Torere, Amazonas, Rastro
Jogos de alinhamento	Movimentar as peças para alinhá-las	Picaria, Tapatan, Shisima, Three Men's Morris, Moinho, Tic Tackle, Tsoro Yematatu, Dara, Tonkim
Jogos de deslocamento	Deslocar as peças para um lugar específico	Halma, Xadrez Chinês, Tábula, Gamão, Mancalas
Jogos de posicionamento	Posicionar estrategicamente as peças que não serão mais movimentadas	Jogo-da-velha, Reversi, Go, Gomoku, Quina, 4-em-linha
Jogos de captura	Capturar as peças adversárias	Alquerque, Felli, Luta-das-serpentes, Pretwa, Damas, Fanorona, Surakarka, Yoté
Jogos de caça	Um jogador persegue as peças do outro	Lebre e cachorros, Leopardo e caçadores, Tablut, Assalto, Lobo e cabras, Jogo-da-onça

Fonte: Ribas (2019, p. 22).

Todos esses grupos de jogos podem ser trabalhados para enfatizar algum tipo de conceito com os participantes. Por exemplo, nos jogos de posicionamento a decisão tomada na hora de colocar uma peça no tabuleiro é definitiva, por mais que o participante tenha feito uma péssima jogada, não é possível voltar atrás. Isso traz um ensinamento muito importante para a vida do participante.

Em Ribas (2019), é descrito um trabalho de pesquisa feito em uma escola onde o objetivo era analisar a aplicação de jogos de tabuleiro nas séries iniciais do ensino fundamental. Para isso, a metodologia e os materiais do projeto de extensão (LoBoGames) foram utilizados. A pergunta inicial proposta pela autora foi “Quais os benefícios e a viabilidade de inserção dos jogos de tabuleiros no currículo escolar?” Dentre os benefícios descritos no artigo, os principais foram:

Interesse: Diz respeito à capacidade que os jogos lógicos têm de despertar o interesse dos alunos pelo novo. Isso traz engajamento e faz com que os professores possam trabalhar diversos assuntos sem que o aluno enxergue isso como algo monótono.

Socialização: A interação social é de extrema importância para o desenvolvimento do ser humano. Essas dinâmicas envolvendo jogos fazem os alunos socializarem com os seus colegas, fazendo com que até mesmo os mais tímidos e com dificuldades de convivência sejam acolhidos.

Cooperação: Um dos ensinamentos mais importantes que o ambiente escolar pode trazer para o aluno é o conceito de cooperação. Todos os avanços que a nossa sociedade teve foram através da cooperação entre diferentes tipos de pessoas. Quando trazemos esse conceito para dentro das escolas através de uma dinâmica de jogo em grupo, isso faz os alunos entenderem o valor da cooperação.

Cognição: Quando jogos são usados como estratégia de ensino na sala de aula, isso cria diversas situações que permitem que os alunos desenvolvam métodos para resolução de problemas, além de estimular a criatividade.

Todos os benefícios trazidos pela aplicação do projeto LoBoGames são extremamente difíceis de serem alcançados quando falamos do ensino à distância pelo fato de os professores perderem o controle sobre as ações que acontecem ao redor do aluno. No momento em que vivemos, é de extrema importância criar algum mecanismo que seja possível fazer com que os alunos, mesmo estando em ensino remoto, consigam ter interesse pelo assunto que está sendo apresentado, consigam socializar com outros colegas e estar em um ambiente cooperativo que ajude no desenvolvimento da sua cognição. Com isso, torna-se clara a importância de criar um ambiente que traga para o mundo virtual todos os benefícios que a metodologia LoBoGames proporciona no contato presencial.

3.2 Jogos de tabuleiro e ambientes virtuais aplicados como instrumento pedagógico

Hoje em dia existem diversas plataformas que possibilitam que pessoas, em lugares diferentes, consigam se divertir através dos jogos de tabuleiro. Todas essas plataformas implementam os aspectos discutidos no Capítulo 2. Porém, como o foco dessas plataformas está no jogo, fica difícil realizar alguma oficina de aprendizado, pois muitas vezes as regras do jogo podem ser complexas e acabar afastando participantes que não as dominam. Outro problema é o propósito da plataforma. Esses ambientes de jogos estão na categoria de CVEs aplicados a jogos, sendo assim, seu uso para educação se torna prejudicado.

Com o constante avanço tecnológico, é imprescindível que tecnologias sejam utilizadas para tornar o ensino mais atrativo para os alunos, porém, o emprego de jogos digitais nas escolas ainda é modesto. Dentre vários motivos estão a falta de equipamentos, falta de treinamento para os professores e a necessidade de cumprir a grade curricular tradicional. Se pensarmos no ensino remoto, como conseguir fazer uma atividade com jogos, onde o professor consiga trazer os ensinamentos propostos e ao mesmo tempo manter o engajamento dos alunos? Segundo Prensky (2012), um dos grandes desafios da aprendizagem baseada em jogos digitais é manter os participantes focados no jogo e na aprendizagem ao mesmo tempo, considerando uma atividade de ensino remoto esse desafio torna-se ainda maior.

4 PROPOSTA

Conforme discutido no Capítulo 2, existem algumas características importantes para um ambiente virtual ser considerado um CVE. A representação do ambiente virtual e as regras que o governam são de suma importância para que o ambiente seja agradável para quem está usando. Outra característica é a interatividade, pois é ela que permite a troca de experiências entre os usuários. E, por fim, os avatares são os responsáveis por trazerem a sensação de imersão dentro da plataforma. Já no Capítulo 3 foi discutido a gamificação do ensino e a inserção de jogos lógicos de tabuleiro nas escolas. Dentre os principais benefícios dessa abordagem estão o interesse do aluno, a socialização, a cooperação e a cognição. Tendo em vista os aspectos discutidos nos capítulos anteriores, o objetivo deste capítulo é propor um ambiente virtual colaborativo em que seja possível trazer para o mundo virtual todos os benefícios que são obtidos com a aplicação do projeto LoBoGames.

O trabalho proposto aqui tem como missão ser uma plataforma que possibilite a aplicação de jogos abstratos de estratégia como instrumento pedagógico no ensino remoto, embora nada impede que seja usado no ensino presencial, mas a principal característica do ambiente é trazer uma possibilidade de desenvolvimento da cognição através do lúdico, utilizando tecnologia para tornar essa experiência possível. Tendo como base as dinâmicas trazidas em Ribas (2019), sabemos que existe um potencial gigantesco da plataforma ser usada como uma forma de facilitação para o ensino remoto. Analisando o ensino à distância, é possível perceber que existem diversas plataformas que possibilitam ao professor ter um controle de tarefas, controle do progresso dos alunos, possibilidade de aulas ao vivo e muito mais. Porém, quando olhamos para a utilização de jogos de tabuleiro como forma de ensino não temos tantas opções.

De maneira geral, a proposta é fornecer um ambiente virtual que não esteja preso a nenhum jogo ou regra específica, que possibilite a escolha de qualquer tipo de tabuleiro e permita o monitoramento do desempenho dos alunos através de algumas métricas. Essas métricas ajudam o professor a perceber quais tipos de atividades são mais importantes para o desenvolvimento de cada aluno, possibilitando assim uma experiência muito parecida com a presencial. Além disso, outra característica da plataforma que diminui a distância do virtual para o presencial é a possibilidade de interação via *chat*, voz e vídeo, possibilitando que os participantes do ambiente interajam entre si. Essa possibilidade de interação torna a experiência dos usuários mais completa e abre diversas possibilidades de utilização da plataforma.

4.1 Funcionalidades do sistema

Antes de explicar as principais funcionalidades que foram pensadas para a plataforma, algumas nomenclaturas precisam ser definidas. Uma sala de jogos nada mais é do que a representação virtual de uma sala de aula do mundo real. Quando o projeto LoBoGames é aplicado em alguma escola, existe o tabuleiro, os alunos e os professores, e todos esses estão dentro de um mesmo espaço, por isso o termo “sala de jogos”. Em relação aos alunos e professores dentro do sistema há algumas diferenças. Os usuários que são do tipo professor têm a permissão para criar, editar e excluir salas, já os usuários do tipo aluno só conseguem entrar em uma determinada sala. As dinâmicas de jogos são divididas entre partidas e rodadas. As partidas referem-se a aplicação de algum jogo entre dois ou mais usuários, já as rodadas referem-se a um conjunto de uma ou mais partidas.

A plataforma apresenta dois modos distintos, um modo professor e outro aluno. Como a ideia principal do ambiente é ser utilizado como ferramenta de aprendizagem para crianças e adolescentes do ensino fundamental, o cadastro dos alunos fica por conta do professor. Sendo assim, um aluno só pode entrar na plataforma se for convidado por algum educador. Essa limitação de acesso foi pensada para evitar que os alunos sofram algum tipo de constrangimento dentro da plataforma, ficando sob responsabilidade do professor a supervisão dos alunos cadastrados. Outro aspecto importante é que, apesar do ambiente ter diversas características semelhantes aos ambientes de jogos *online*, esse ambiente foi criado para ser usado como ferramenta de educação. Portanto, existem diversas configurações e métricas que estão disponíveis para os professores e que não fazem sentido no modo aluno. Todos os ambientes virtuais de aprendizagem (AVA) disponibilizam algum modo dos professores acompanharem o desempenho dos alunos. Dessa forma, todos os dados gerados em uma sala de jogos serão salvos para uma análise posterior, ficando disponível somente no modo professor.

Quando um aluno entra no sistema, no painel do aluno, aparecerá somente as salas em que ele foi convidado e as atividades que o professor cadastrou para esse aluno. Isso faz com que esses usuários não percam tempo com questões relacionadas a cadastro e configuração.

Conforme comentado anteriormente, o sistema não faz nenhum tipo de verificação das regras dos jogos, ficando sob responsabilidade do professor definir, juntamente com seus alunos, quais são as regras para cada partida. Em uma sala de jogos existem as partidas e as rodadas, cada partida é iniciada e finalizada pelo professor, e cada rodada é controlada pelo sistema, conforme informado na configuração da sala. Por exemplo, ao criar uma sala, o

professor pode dizer que quer aplicar cinco rodadas com dez partidas cada uma. Como o sistema não tem controle sobre os jogos que estão sendo aplicados, o administrador da sala deve informar quando uma partida começou e quando uma partida terminou. Essa dinâmica facilita o controle do aumento de dificuldade pelo sistema. Outra característica é que a plataforma não bloqueia os alunos que não estão jogando, ou seja, até mesmo a organização de quem deve mexer no tabuleiro é de responsabilidade do professor, pois isso também faz parte do aprendizado. O professor só deve informar quais alunos estão jogando (ou qual grupo de alunos) para que a plataforma faça o controle de desempenho. Todos os dados gerados em uma sala de jogos, como quantidade de partidas, ganhadores, anotações dos jogos e mensagens do tabuleiro são gravadas e disponibilizadas para o professor verificar o desempenho de cada aluno.

Após esse panorama geral do sistema, a explicação será dividida em duas partes. A primeira diz respeito ao modo professor e vai desde a criação de uma sala até o controle de uma oficina de jogos. A segunda fala sobre o modo aluno que vai desde a criação do seu avatar até a participação em uma partida.

4.1.1 Modo Professor

Ao se cadastrar na plataforma, o educador tem a possibilidade de configurar o seu avatar. Conforme discutido no Capítulo 2, os avatares são extremamente importantes em um ambiente virtual, pois são eles que nos distinguem dos outros participantes. Se existimos no mundo real, no mundo virtual precisamos de algo que nos defina. Outro aspecto importante e que justifica a criação dos avatares personalizados, e não simplesmente a foto dos usuários, é que isso diminui a distância entre alunos e professores, e entre os próprios alunos. É muito comum no ambiente escolar haver algum tipo de discriminação entre os alunos por questões estéticas assim como um distanciamento dos professores em relação aos alunos. Essa funcionalidade dos avatares visa diminuir esses problemas.

Após finalizar o cadastro com a criação do seu avatar, o educador é levado a um painel com as seguintes opções: “criar sala de jogos”, “desempenho dos alunos”, “gerenciador de salas” e “criar atividades individuais”. Cada uma dessas opções tem como objetivo disponibilizar ao professor uma experiência muito parecida com a presencial.

A ideia do ambiente não é estar preso a um jogo e suas regras, mas sim oferecer a possibilidade de o educador escolher qualquer tipo de tabuleiro e aplicar as regras que melhor se adequem ao nível de conhecimento da turma. Com isso, qualquer jogo pode ser aplicado,

deixando o ambiente virtual muito parecido com as oficinas aplicadas presencialmente. Além disso, durante uma rodada, o administrador da sala pode mudar o tabuleiro e aplicar outro tipo de atividade com os alunos. Isso mostra a flexibilidade do ambiente.

Para criar uma sala de jogos o professor deve configurá-la. Essa configuração é simples, porém, é muito importante para que o sistema gere os dados que o educador deseja analisar. Primeiramente, é solicitado que um nome de sala seja escolhido. Este nome deve ser único para o administrador. Logo após, o tipo de tabuleiro deve ser escolhido.

Uma das metodologias de ensino adotadas em Ribas (2019) é aumentar a dificuldade dos jogos com o passar das partidas. Essa abordagem visa deixar o aluno sempre instigado a aprender algo novo e ajuda no aprendizado de alunos com maior dificuldade, pois, dessa forma, a complexidade do jogo aumenta aos poucos e possibilita que todos os alunos consigam evoluir e aprender melhor com o passar das partidas. Visando tornar essa prática possível também no mundo virtual, o sistema permite a configuração de um aumento de dificuldade constante com o passar das rodadas. Para isso, três possibilidades são mostradas na criação da sala.

A primeira delas é a temporização. Na configuração da plataforma o professor pode dizer ao sistema que deseja diminuir o tempo de execução das jogadas a cada rodada. Por exemplo, supondo que o jogo de damas esteja sendo aplicado, o professor pode informar ao sistema que deseja que a primeira rodada tenha um tempo de jogada de trinta segundos e que, a partir da segunda rodada, esse tempo diminua em cinco segundos. Além de deixar a aula divertida, essa abordagem faz com que os alunos desenvolvam o seu raciocínio de forma mais rápida.

O segundo tipo de aumento de dificuldade é o bloqueio de caminhos. O sistema pode ser configurado para bloquear alguns caminhos do tabuleiro, de forma aleatória, a cada partida, fazendo com que os jogadores tenham que pensar em outro tipo de jogada.

O último tipo de dificuldade que pode ser configurada em uma sala de jogos são os desafios. Essa técnica visa aumentar o raciocínio e mesclar a aprendizagem de jogos de tabuleiro com outros assuntos. O que essa abordagem faz é colocar algum tipo de desafio na tela para que o aluno resolva antes de realizar a sua jogada, tendo um tempo máximo para responder esse desafio. Esses desafios são configurados pelo professor, onde as perguntas e respostas devem ser informadas e o sistema faz todo o gerenciamento.

Existe uma quarta forma de aumento de dificuldade que é a mescla de todas as possibilidades discutidas anteriormente. Por exemplo, na mesma partida podemos ter

temporização combinada com desafios. Isso traz uma dinâmica muito interessante, e possibilita que os alunos não sintam que a aula está sendo monótona.

Uma das principais características do projeto LoBoGames é a interação dos participantes. Isso acelera o aprendizado, possibilita que o professor tenha um melhor acompanhamento dos alunos e permite que questões emocionais e de características da personalidade de cada aluno sejam verificadas. Mas como trazer essa interação para o ambiente virtual? Um dos grandes desafios de qualquer AVA é diminuir o distanciamento natural que os ambientes virtuais trazem. Para tentar diminuir esse distanciamento e trazer uma experiência mais colaborativa para a plataforma, uma possibilidade é habilitar a comunicação via texto, voz e vídeo. Essas formas de comunicação são de suma importância pois fazem com que os alunos consigam conversar e interagir. Com a possibilidade do vídeo ser habilitado, o professor pode até mesmo dar uma aula mais teórica com apresentação de *slides*, ou seja, a plataforma não oferece somente um ambiente para jogos, mas sim um ambiente completo para que uma aula possa ser ministrada.

Tendo feito as configurações iniciais da sala, o educador deve convidar os alunos para participarem da sala de jogos. Se este aluno não estiver cadastrado no sistema, o professor deverá realizar o seu cadastro colocando as informações básicas do aluno. Além disso, é possível escolher o tipo de personalidade que mais se encaixa com as características de cada aluno. Por exemplo, se um aluno é mais tímido essa opção pode ser marcada no cadastro, possibilitando ao professor fazer um paralelo entre os resultados de cada aluno com a sua personalidade. Ao finalizar o cadastro, o sistema cria a sala e envia um email para todos os convidados, informando o *login*, senha de acesso (para usuários novos) e o *link* de acesso ao ambiente.

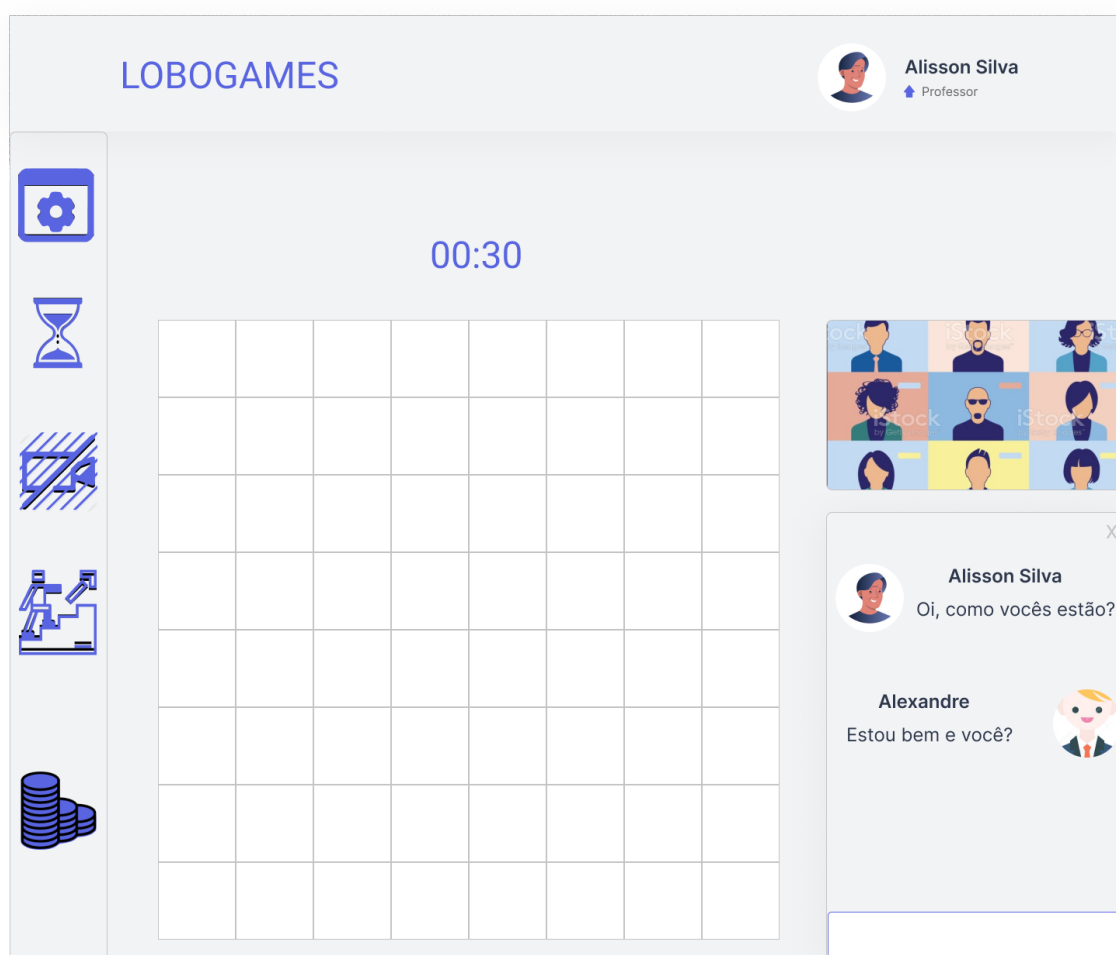
Conforme uma oficina de jogos é aplicada no sistema, acontece um monitoramento dos dados que são gerados e, dependendo da configuração da sala, esses dados são disponibilizados como métricas. Por exemplo, ao final de cada partida o professor deve informar qual foi o aluno vencedor e, com base nessa informação, o sistema grava quantas partidas aquele determinado usuário venceu. Além dos vencedores, a sequência de jogadas de cada partida pode ser salva para uma análise posterior. Todas essas métricas estão disponíveis na opção “desempenho dos alunos”, já a opção “gerenciador de salas” disponibiliza uma interface para a edição e exclusão de uma sala.

Além de todas essas possibilidades, ainda é possível configurar um ambiente de treinamento para alunos que queiram elevar os seus conhecimentos em jogos através da opção “criar atividades individuais”. Nessas atividades individuais o aluno joga contra o próprio

sistema em um ambiente com os mesmos desafios aplicados em uma sala de jogos em grupo, ou seja, temporização, desafios antes de cada jogada e bloqueio de caminhos também estão disponíveis na sala individual.

A sala de jogos no modo professor, conforme mostra a Figura 4.1, disponibiliza algumas opções para condução de uma atividade, como configuração das rodadas e partidas. Além disso, é possível anotar os confrontos que serão realizados em uma determinada rodada (lembrando que essa anotação é somente para facilitar o uso, pois o sistema permite que todos os usuários utilizem o tabuleiro ao mesmo tempo).

Figura 4.1 – Protótipo do ambiente virtual.



Fonte: Próprio autor.

4.1.2 Modo Aluno

No modo aluno, toda a parte de configuração da plataforma, métricas e desafios não estão visíveis. Essa distinção entre a interface do professor e do aluno é importante, pois dessa forma cada tipo de usuário concentra-se somente naquilo que é de sua responsabilidade.

Ao receber os dados de acesso, o aluno fica habilitado a entrar na plataforma. No seu primeiro acesso será necessário completar o cadastro criando um avatar próprio para representar a sua jornada no sistema. Na tela inicial será vista a lista de salas em que esse aluno está cadastrado e a lista de atividades individuais que foram configuradas para esse participante.

Para dar aos alunos a sensação de evolução dentro da plataforma, é fornecido um sistema de pontuação, onde a cada atividade realizada pelo aluno faz com que ele receba uma pontuação proporcional ao seu desempenho e participação naquela atividade. Tanto nas oficinas aplicadas pelo professor como nas atividades individuais o aluno é avaliado e a sua pontuação é aumentada. Vale ressaltar que essa pontuação não leva em consideração somente se o aluno ganhou ou perdeu uma determinada partida, mas sim o desempenho do aluno como um todo. Não existe a possibilidade do aluno perder ponto. Dependendo do seu desempenho, o aluno pode ganhar mais ou menos pontos, essa dinâmica faz o aluno ter a sensação de evolução tornando a experiência mais desafiadora.

Ao entrar em uma sala de jogos, o aluno terá acesso ao tabuleiro, *chat* e a possibilidade de entrar em uma vídeo chamada com os participantes da sala. Os alunos podem mexer no tabuleiro mesmo quando não estão participando de um jogo, conforme discutido na seção anterior. Fica a cargo do professor explicar e controlar as regras. Quando ocorre o aumento da dificuldade do jogo através da temporização, desafios ou bloqueio de caminhos, todos os usuários conseguem visualizar o que está acontecendo. Com isso, conseguimos manter todos os alunos concentrados na plataforma.

Uma das opções disponíveis no modo aluno é a sala de aprendizado individual. Essa sala tem as mesmas características da sala de jogos em grupo. Esse módulo do ambiente serve como uma plataforma de aprendizado conduzido onde, na configuração de uma sala, o professor escolhe as características que deseja abordar para que o aluno consiga treinar aspectos que ele ainda tem dificuldade. Essa sala de aprendizado conduzido, diferentemente da sala em grupo, é controlada pelo sistema, ou seja, o usuário só pode mexer no tabuleiro quando for a sua vez, e o sistema controla o início e fim de um jogo.

4.2 Inovações e desafios que o ambiente virtual proporciona

A idealização desta plataforma tem como base as experiências adquiridas na aplicação do projeto LoBoGames em escolas. Porém, podemos pensar um pouco mais além e discutir novas possibilidades que o ambiente virtual proporciona. Um exemplo disso são as anotações das jogadas. Essa funcionalidade diz respeito ao sistema de gravar a sequência de jogadas que foram realizadas em uma determinada partida. No ambiente presencial é difícil conseguir aplicar uma aula onde as jogadas sejam analisadas. Já no ambiente virtual, com as anotações, isso torna-se fácil de ser feito. Com os dados de uma partida em mãos, o professor pode ministrar uma aula onde a sequência de jogadas realizadas são analisadas e discutidas, fazendo com que o processo de aprendizagem seja melhorado.

Um dos principais desafios da aplicação de jogos lógicos de tabuleiro em escolas, conforme descrito em Ribas (2019), é manter o engajamento dos alunos. É muito comum no início da aplicação do projeto os alunos estarem motivados e com muita vontade de aprender algo novo. Porém, com o passar do tempo, a euforia passa e os alunos ficam cada vez mais dispersos. Apesar da estratégia de aumentar o nível de dificuldade ser importante para manter o aluno instigado em sempre aprender algo novo, a falta de engajamento é um problema, não só na aplicação do projeto LoBoGames mas também na aplicação de conteúdos tradicionais. Quando pensamos em trazer a tecnologia para dentro do ensino, um dos principais fatores é se aproximar cada vez mais da realidade dos alunos, que desde cedo já estão conectados à internet. Sendo assim, a plataforma proposta aqui dispõe de alguns mecanismos que visam manter o aluno interessado no conteúdo. Uma das formas que o sistema tem de manter o engajamento com a plataforma é trazer algumas características dos ambientes de jogos online, tais como sistema de pontuação, desafios aplicados no meio do jogo e temporização nas jogadas. Essas funcionalidades permitem que os alunos sempre tenham um desafio diferente ao entrar na plataforma, o que facilita o aumento do aprendizado.

Uma das questões mais importantes no mundo de hoje é a acessibilidade. É de suma importância que tanto as escolas quanto os professores estejam preparados para receber alunos com algum tipo de dificuldade. Porém, é muito comum vermos escolas que não estão preparadas para isso. Quando falamos de uma plataforma virtual para o ensino, é impossível não pensarmos em acessibilidade. A idealização da plataforma prevê um modo de “acessibilidade” onde todas as ações que acontecem no tabuleiro são convertidas em voz, as

peças do tabuleiro podem ser movimentadas através do teclado e as mensagens do *chat* também são transformadas em áudio.

Apesar de toda a capacidade tecnológica que temos hoje em dia, alguns desafios devem ser enfrentados quando pensamos em uma plataforma que tenha o lúdico como um instrumento de aprendizado. Diversas funcionalidades foram pensadas para minimizar os impactos, porém, a distração e a perda de foco são os principais problemas desse tipo de ambiente. Se no presencial é difícil manter os alunos interessados pelo conteúdo que está sendo passado, como fazer isso no ambiente virtual? A questão aqui não é somente a plataforma, mas sim todo o entorno que cerca o aluno. Questões como conexão de internet instável, ambiente tumultuado, falta de webcam e a falta de dispositivo de som fazem a experiência na plataforma ser completamente afetada. Outra questão importante é que, no presencial, muitas vezes o professor consegue perceber se um aluno está com alguma dúvida, mesmo que este não faça nenhum questionamento. Já no ambiente virtual essa percepção é completamente prejudicada.

Apesar de todos os problemas que esse tipo de ambiente possa ter, a sua importância é notável, não só pelo fato de ajudar os alunos a desenvolverem o raciocínio lógico, mas também por ser um espaço de interação e de trabalho colaborativo.

5 MÍNIMO PRODUTO VIÁVEL - IMPLEMENTANDO AS FUNCIONALIDADES DO SISTEMA

Com o objetivo de discutir a viabilidade técnica da plataforma, foi criado um MVP (mínimo produto viável) que implementa algumas funcionalidades que foram apresentadas na proposta. Se voltarmos ao Capítulo 4, onde a idealização do ambiente foi discutida, conseguimos perceber que a plataforma tem dois pilares. O primeiro deles é a interação. Portanto, uma funcionalidade essencial no MVP é disponibilizar um *chat* por texto e a comunicação através de voz e vídeo. Como o objetivo é ter um ambiente de jogos que não fique preso a nenhuma regra específica, podemos dizer que o trabalho colaborativo também é um pilar importante, visto que todos os usuários da plataforma podem mexer no tabuleiro ao mesmo tempo, abrindo um grande leque de possibilidades.

Com esses pilares, o MVP construído possui as funcionalidades de cadastro do professor, *login* de alunos e professores, criação de uma sala de jogos com um tabuleiro 3x3, tabuleiro interativo onde o participante movimenta as peças e as suas ações são replicadas para os outros usuários, *chat* para comunicação via texto e a possibilidade de realizar uma chamada de vídeo com os participantes da sala.

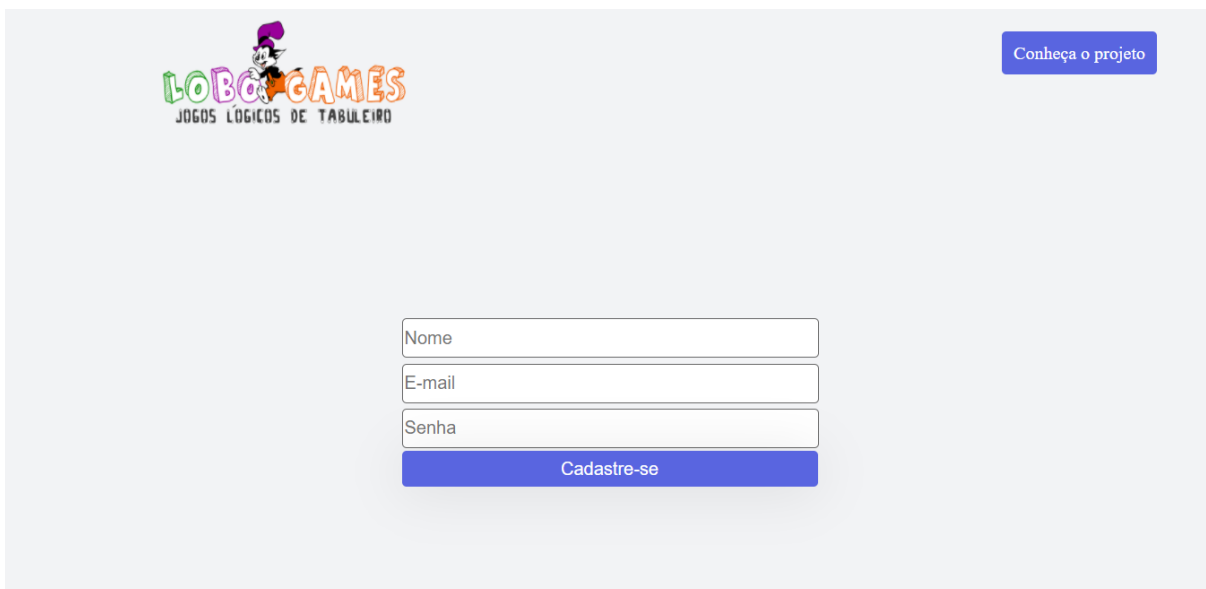
Dando um panorama geral de como o MVP funciona, primeiramente a plataforma é dividida em dois tipos de usuários, alunos e professores. Um professor que deseja ministrar alguma atividade utilizando a plataforma deve realizar o seu cadastro. Esse cadastro é bem simples, sendo necessário informar somente o nome, email e senha para acesso, conforme mostra a Figura 5.1.

Após realizar o cadastro, o sistema direciona o professor para o painel de usuários. É nessa tela que ficam os menus “Criar sala de jogos” e “Gerenciador de salas”. Na Figura 5.2 é possível perceber que o menu “Desempenho dos alunos” não está disponível (isso porque essa funcionalidade não foi implementada).

Ao clicar em “Criar sala de jogos”, o professor deverá preencher um formulário colocando o nome da sala, o tipo de tabuleiro desejado e os dados dos usuários convidados. Esse MVP disponibiliza somente um tabuleiro 3x3. Essa medida foi tomada por simplicidade pois a ideia é mostrar a viabilidade do sistema. Na criação da sala, o professor pode adicionar convidados. Se esses convidados forem novos usuários, o professor deverá informar os dados e cadastrar esses usuários na plataforma. Se eles já existirem somente o email deve ser informado. Essa dinâmica tem como objetivo deixar na mão do professor a responsabilidade sobre os alunos cadastrados. Além de alunos, o professor pode convidar outros professores

para participarem da sala, pois é comum algumas pessoas auxiliarem os professores na aplicação das dinâmicas.

Figura 5.1 – Tela de cadastro no MVP.



A tela de cadastro do MVP apresenta o logo "LOBOGAMES" com o subtítulo "JOGOS LÓGICOS DE TABULEIRO" no canto superior esquerdo. No canto superior direito, há um botão azul com o texto "Conheça o projeto". O formulário centralizado contém três campos de entrada: "Nome", "E-mail" e "Senha". Abaixo dos campos, há um botão azul "Cadastre-se".

Fonte: Próprio autor.

Figura 5.2 – Painel do professor no MVP.



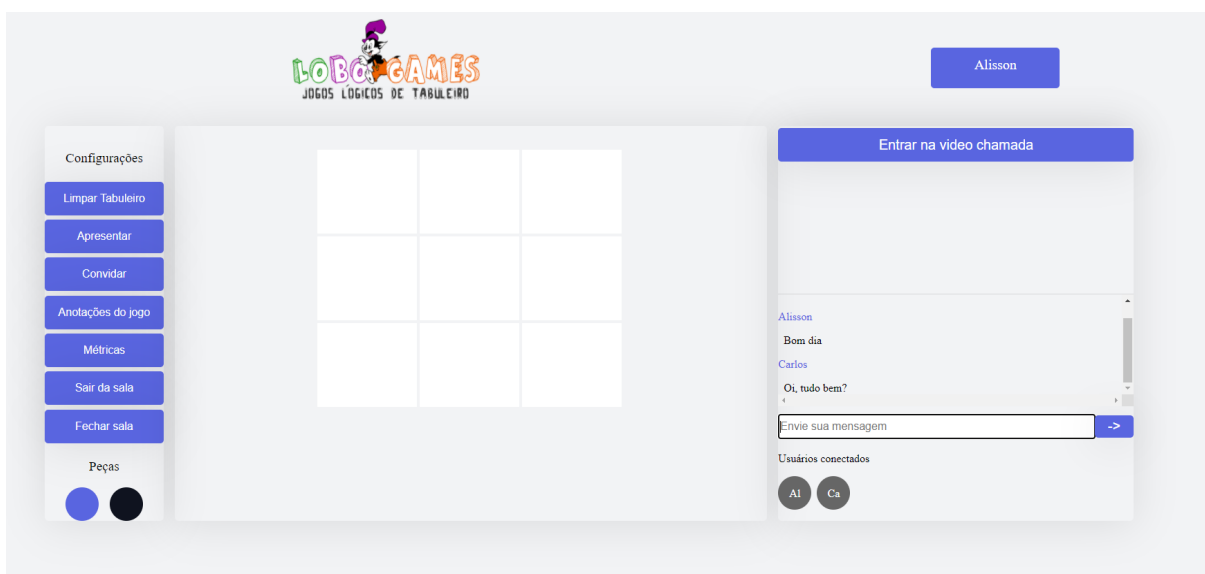
O painel do professor no MVP exibe o mesmo logo "LOBOGAMES" no canto superior esquerdo e o nome de usuário "Alisson" no canto superior direito. Abaixo, há uma mensagem de boas-vindas: "Bem vindo(a) a plataforma que vai revolucionar o jeito que você ensina." Três cartões de funcionalidade são apresentados: "Criar sala de jogos" (acompanhado de uma imagem de crianças jogando), "Desempenho dos alunos" (acompanhado de uma imagem de uma sala de aula) e "Gerenciador de salas" (acompanhado de uma imagem de uma sala de aula).

Fonte: Próprio autor.

Ao realizar a criação da sala, o sistema manda um email para todos os usuários convidados. A Figura 5.3 mostra uma sala de jogos com dois usuários conectados trocando

algumas mensagens pelo *chat*. Além disso, no menu à esquerda temos algumas opções que são disponibilizadas para os usuários. Os menus “Apresentar”, “Convidar”, “Anotações do jogo” e “Métricas” não foram implementados, mas fazem parte dos próximos passos para a evolução da plataforma. Para os professores são disponibilizados os botões de “Limpar o tabuleiro”, “Sair da sala” e “Fechar a sala”. Essa distinção entre sair da sala e fechar a sala serve para o sistema saber quando deve gravar os dados no banco de dados. Quando um professor fecha uma sala, todos os usuários são desconectados da sala e são automaticamente redirecionados para o painel principal. No meio da tela temos o tabuleiro. Esse tabuleiro é preenchido arrastando as peças que estão à esquerda para alguma posição disponível. A ideia é que no futuro a plataforma tenha diversos tipos de tabuleiro, para dar mais flexibilidade ao educador.

Figura 5.3 – Imagem de uma sala de jogos.



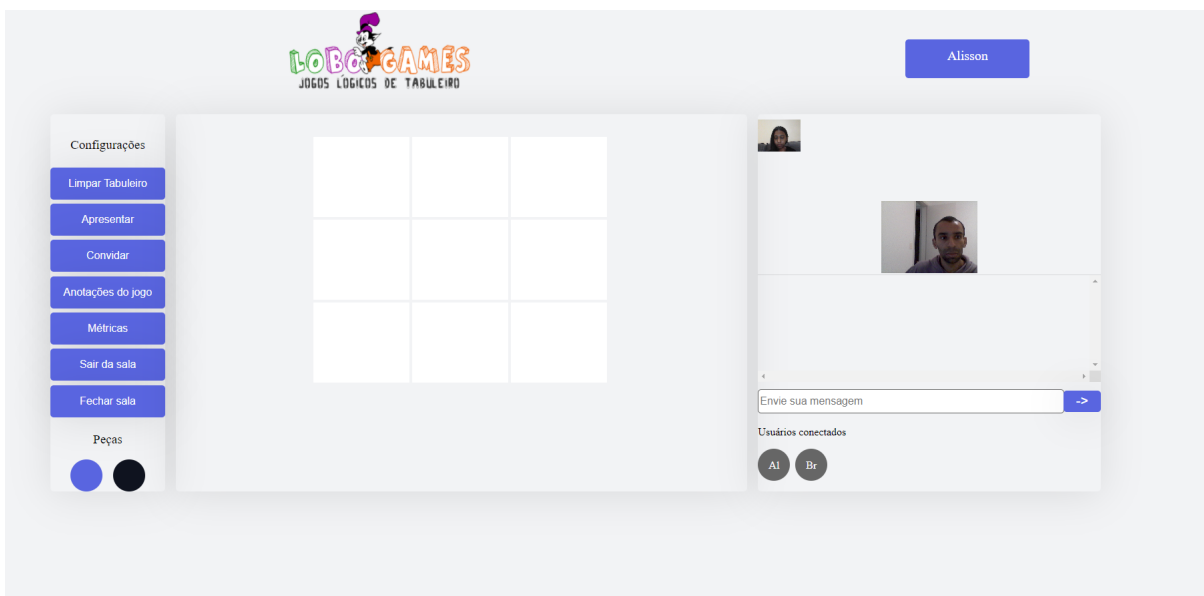
Fonte: Próprio autor.

Mais à direita, nós temos um módulo que disponibiliza duas formas de interação com outros participantes, uma via *chat* e outra através de videochamada. Todos os participantes conectados na sala podem mandar mensagens pelo *chat* e entrar na chamada de vídeo. A Figura 5.4 mostra o *layout* do sistema quando uma chamada por vídeo está acontecendo.

Quando um aluno entra na plataforma, é disponibilizado um painel com todas as salas em que ele faz parte. Diferente de um professor, os usuários do tipo aluno não têm acesso a nenhum tipo de configuração do sistema, até mesmo na sala de jogos só fica disponível o menu “Sair da sala”.

Até agora foram discutidas todas as funcionalidades do MVP e qual o caminho que cada tipo de usuário percorre dentro da plataforma, porém, como fazer tudo isso acontecer? Como fornecer um ambiente com tanta troca de mensagens? A partir de agora os aspectos técnicos para a construção da plataforma serão discutidos.

Figura 5.4 – Imagem de uma sala de jogos com chamada de vídeo.



Fonte: Próprio autor.

5.1 Arquitetura geral do MVP

Quando pensamos em um software que tenha a necessidade de ser facilmente acessado e que exige poucos recursos da máquina, é quase natural pensarmos em uma solução voltada para a *web*. No caso do MVP proposto aqui, a grande maioria dos usuários serão crianças e adolescentes do ensino fundamental. Portanto, exigir que esses usuários façam a instalação de um sistema em seu computador é praticamente inviável. Dito isto, fica claro que o produto desenvolvido aqui é um sistema *web*, pela facilidade de acesso.

Agora que sabemos que o ambiente desenvolvido é um sistema *web*, vamos pensar o que é necessário para desenvolver um projeto desse tipo. Primeiramente, nós temos usuários que se conectam no sistema através de *login* e senha, portanto, precisamos ter um banco de dados para armazenar as informações desses usuários. Esse banco de dados deve ter duas entidades, uma entidade que representa os usuários e outra que represente as salas, visto que é possível fazer o cadastro de salas no sistema. Mas como essas informações são gravadas,

acessadas e atualizadas no banco de dados? Através de um servidor, portanto, temos que ter um sistema rodando em um computador que seja responsável por receber requisições, acessar o banco de dados e responder a essas requisições. E quem faz essas requisições ao servidor são os clientes, ou seja, os clientes são os responsáveis por fazerem as solicitações ao servidor pedindo alguns dados.

Durante muito tempo, quando uma aplicação *web* era desenvolvida, o conceito de renderização do lado do servidor (SSR - *server-side-rendering*) era aplicado. Basicamente, o servidor da aplicação ficava responsável por tratar os dados, gerar a interface visual da aplicação e retornar tudo pronto para o cliente. Essa forma de desenvolvimento faz com que os servidores tenham que ser mais robustos, aumentando o custo de forma considerável. Com o avanço das tecnologias *frontend*, o lado do cliente foi ganhando cada vez mais responsabilidades, deixando para o servidor somente a missão de tratar os dados da aplicação. Através desse movimento surgiram as aplicações de página única ou SPA (*single page application*). Apesar do nome, as SPAs não têm somente uma página. O que acontece é que ao invés de gerar todo o *html* em tela toda vez que uma rota da aplicação é acessada (como acontecia com a SSR), as SPAs mantêm o esqueleto da aplicação e mudam somente o conteúdo que foi alterado.

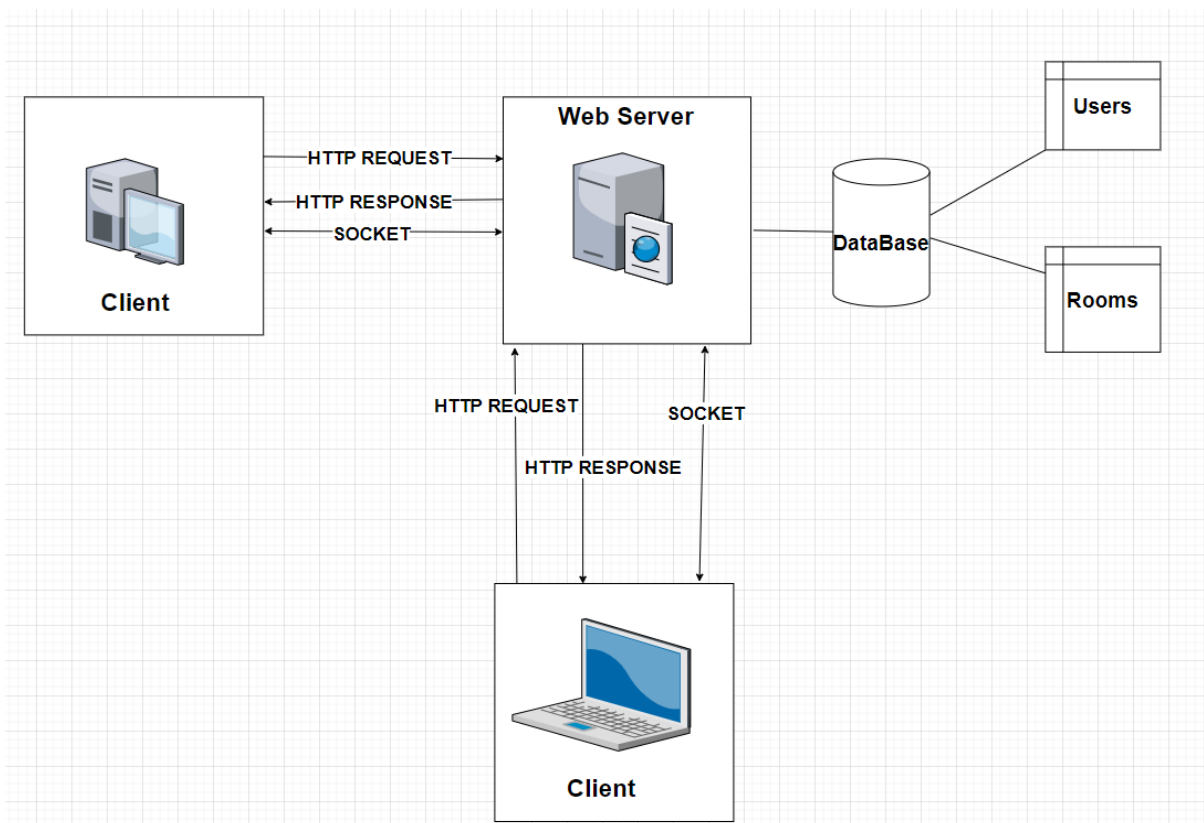
Sendo assim, a arquitetura cliente-servidor, onde o cliente é um SPA e o servidor só fica responsável por fornecer os dados para o cliente via *json*, foi utilizada para desenvolver a plataforma. Com isso, o servidor expõe uma interface de comunicação ou API (*application interface*) e essa interface é através do protocolo HTTP. O HTTP (protocolo de transferência de hipertexto) é um protocolo de comunicação da camada de aplicação que funciona como um protocolo de requisição e resposta, o que possibilita que mensagens sejam trocadas entre cliente e servidor.

O HTTP funciona muito bem quando queremos buscar, editar, gravar e deletar dados do servidor. Porém, por ser um protocolo que não mantém nenhuma informação sobre os clientes, é difícil implementar uma solução que forneça comunicação em tempo real entre todos os usuários conectados. Em uma sala de jogos as atualizações do tabuleiro, do *chat* e do vídeo são informações que precisam ser atualizadas em tempo real. Portanto, não basta o servidor expor somente a interface HTTP para comunicação. Sendo assim, o servidor desenvolvido expõe, além de uma interface de comunicação via HTTP, um canal de comunicação via *sockets*, que fornece atualizações em tempo real para os clientes. Em redes de computadores, *socket* é uma interface que permite a comunicação entre processos na mesma máquina ou em máquinas diferentes, na camada de transporte. Basicamente os *sockets*

abstraem a camada de rede fornecendo uma API onde quem inicia uma conexão só precisa saber o endereço IP e a porta em que o servidor está escutando as requisições. Utilizando o protocolo TCP temos um canal de comunicação confiável, via *socket* TCP, que garante a entrega das mensagens (ou segmentos), garante que essas mensagens chegarão na ordem correta ao seu destino e, através dos números de *socket*, consegue manter o estado da conexão com os clientes.

Portanto, o servidor desenvolvido para essa plataforma possui duas interfaces de comunicação, uma via HTTP para as funcionalidades de cadastro e *login*, e outra via *sockets* para as funcionalidades de tabuleiro interativo, mensagens de texto e transmissão de vídeo. A Figura 5.5 ilustra a arquitetura geral do sistema.

Figura 5.5 – Arquitetura geral do sistema.



Fonte: Próprio autor.

5.2 Servidor

Uma das decisões mais importantes na criação de um software, e que impacta em todo o processo de desenvolvimento, é a escolha da arquitetura. Arquitetura de software refere-se a

estrutura fundamental do sistema, onde cada estrutura compreende elementos de um software, a relação entre eles e as suas propriedades. Dando uma pesquisada sobre arquitetura, é possível encontrar diversos padrões como MVC, arquitetura hexagonal (*hexagonal architecture*) e a arquitetura limpa (*clean architecture*). Muitas dessas arquiteturas têm um objetivo em comum que é o de separar os elementos de um software em diferentes camadas de acordo com as suas responsabilidades.

Para o servidor desse MVP foi utilizada a arquitetura limpa. Com isso, cinco princípios de desenvolvimento foram seguidos neste projeto, sendo eles, independência de *frameworks*, independência de interface de usuário, independência de banco de dados, independência de qualquer elemento externo e testabilidade. Todos esses princípios têm o objetivo de fazer as regras de negócio da aplicação não dependerem de nenhum elemento externo e separar o software em diferentes camadas, cada uma com a sua responsabilidade. Esse tipo de arquitetura faz com que possíveis mudanças sejam fáceis de serem realizadas. Por exemplo, imagine que o protocolo HTTP seja substituído, provavelmente a biblioteca responsável por implementar o HTTP deva ser modificada. Porém, essa modificação não afeta as outras camadas desde que a mesma interface de comunicação seja usada, portanto, temos uma flexibilidade muito grande ao usar esse tipo de arquitetura.

Com isso temos um software servidor que utiliza a arquitetura limpa como forma de estruturar a aplicação em diferentes camadas. Além disso, esse sistema foi desenvolvido utilizando o *Nodejs*, que basicamente é uma plataforma que permite que a linguagem *Javascript* seja executada no servidor. Para isso, uma máquina virtual chamada V8 é responsável por compilar, otimizar e interpretar os códigos gerados por essa linguagem. A escolha dessa plataforma deve-se ao fato do *Nodejs* ser uma tecnologia assíncrona (devido ao uso do *Javascript*) e que trabalha em uma única *thread* de execução. Portanto, cada requisição ao *Nodejs* não bloqueia o processo do mesmo, fazendo com que seja possível atender um número de requisições muito alto.

Tendo em vista que existem usuários no sistema e esses usuários podem configurar salas, é preciso que o servidor se conecte a um banco de dados para que essas informações sejam armazenadas. Como banco de dados foi utilizado o *MongoDB*, que é um banco de dados *NOSQL* orientado a documentos. O termo *NOSQL* refere-se a banco de dados que não são relacionais, ou seja, oferecem outro mecanismo para modelagem dos dados. No caso do *MongoDB* não existe a necessidade de termos tabelas criadas previamente, isso permite que um documento represente toda a informação necessária. Esses documentos são agrupados em *collections*, ou seja, um conjunto de *collections* forma um banco de dados. Uma das principais

características desse tipo de banco, que fez com que este fosse usado no projeto, é o fato deles fornecerem alta disponibilidade, tolerância à partição e velocidade.

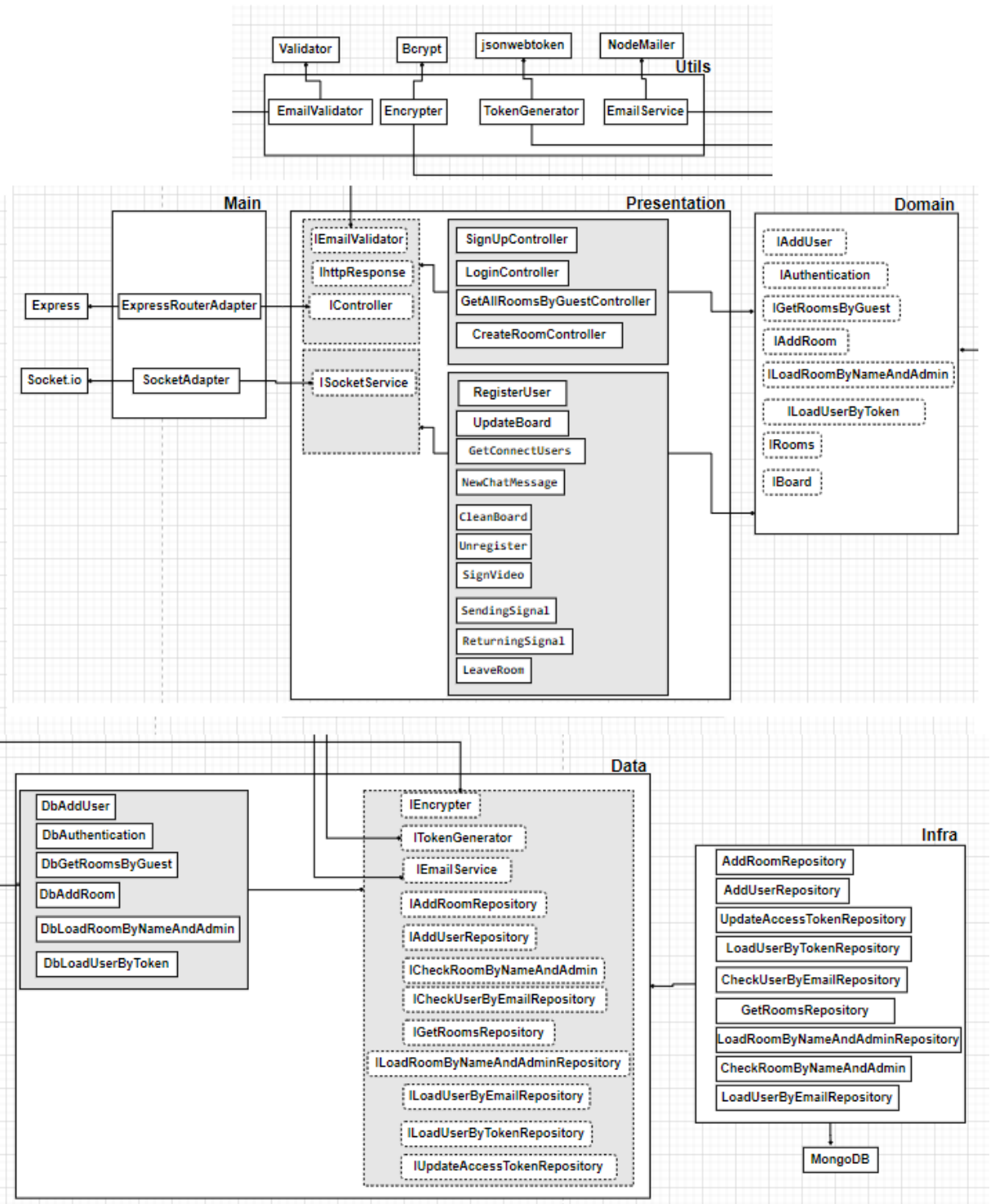
A Figura 5.6 mostra a arquitetura do servidor pelo fato de ter sido construído aplicando os conceitos da arquitetura limpa, o software se divide em seis camadas. A camada mais interna é a camada de domínio (*domain*). É nessa camada que ficam as regras de negócio da aplicação. É importante notar que essa camada não tem implementação de classes. O que ela disponibiliza são interfaces que dizem qual a regra de negócio que a aplicação tem, deixando para outra camada fazer essa implementação. Essa abordagem deixa as regras de negócio da aplicação isoladas e independentes de *frameworks* externos, o que torna o sistema mais flexível e adepto a mudanças de tecnologia.

A camada de domínio contém oito interfaces que representam a definição das regras de negócio e as entidades do sistema. Um usuário no sistema é definido pelo email, senha e o seu tipo, aluno ou professor. Portanto, para ser possível adicionar um usuário no sistema é preciso implementar a interface “*IAddUser*”, que é responsável por definir um método para adicionar um usuário no sistema. Além disso, os usuários podem entrar na plataforma através da autenticação do seu email e senha. Para isso, a interface “*IAuthentication*” é responsável por definir um método que recebe o email e a senha do usuário e retorna um *token* de acesso. Uma das principais entidades do sistema é o tabuleiro, para que seja possível atualizar o tabuleiro de todos os clientes, o servidor precisa implementar uma abstração que representa esse tabuleiro. Para isso, a interface “*IBoard*” define um método responsável por receber um número de linhas, um número de colunas e criar uma matriz em memória para representar o tabuleiro. Nem sempre um tabuleiro será uma matriz, porém, para as funcionalidades propostas nesse MVP, essa abordagem atende todas as necessidades.

Para representar uma sala de jogos, a interface “*IRooms*” foi criada. Essa interface é responsável por definir os atributos de uma sala de jogos no sistema. Além de ter uma lista com todos os usuários conectados, essa interface define também quem é o usuário administrador dessa sala e qual o nome da sala, basicamente esses são atributos que serão gravados na coleção de salas no banco de dados.

A segunda camada é a de dados (*data*). É nessa camada que são feitas as implementações das regras de negócio definidas na camada de domínio. Perceba que, apesar de implementar as interfaces do domínio, essa camada é independente de *frameworks*, pois toda vez que ela depende de algum desses, uma interface é criada fazendo ela depender da interface que implementa a comunicação com esse *framework*. Com isso, podemos trocar as bibliotecas que usamos no projeto sem problemas.

Figura 5.6 – Arquitetura servidor.



Fonte: Próprio autor.

Essa técnica utilizada para fazer as camadas mais externas dependerem das camadas mais internas é chamada de inversão de dependência (*dependency inversion*).

A camada de infraestrutura é responsável por implementar as interfaces criadas pela camada de dados. Nesse projeto, a camada de infraestrutura vai se preocupar com implementações referentes aos métodos para criar, buscar, editar e deletar campos no banco de dados. Essas classes vão depender do *MongoDB*.

Outra camada bastante importante é a de utilitários (*utils*). Essa camada é responsável por fazer a implementação das interfaces de alguns *frameworks* que foram utilizados ao longo do projeto. Por exemplo, quando um usuário faz *login* ou se cadastra na plataforma, o email informado deve seguir o padrão *email@dominio.com*, para garantir o formato correto, antes de procurar o usuário no banco de dados. A biblioteca *validator* foi utilizada.

A camada de apresentação (*presentation*) é a camada que de fato se conecta com o mundo externo. É nela que as rotas da API e as mensagens que são recebidas via *socket* serão definidas. Para expor a interface HTTP, para os clientes enviarem requisições, foi utilizado o *framework express*, pois ele disponibiliza um sistema de rotas completo onde existe tratamento de exceção e gerência de diferentes requisições e seus verbos. Já para expor uma porta para conexão de clientes via *socket*, foi utilizado a biblioteca *socket.io*, pois ela oferece uma API baseada em eventos que permite a comunicação entre cliente e servidor em tempo real.

Para que uma arquitetura com esse nível de desacoplamento funcione precisamos ter uma camada que seja responsável por instanciar todas as classes e permitir que essa comunicação entre as entidades ocorra de forma satisfatória. Em outras palavras, é preciso ter uma camada que conheça todas as outras. Para isso, a camada principal (*main*) foi criada, onde adaptadores de classe foram feitos para que todas as requisições e mensagens recebidas via HTTP com o *express* e via *sockets* com *socket.io* sejam transformadas em dados que as camadas mais internas consigam entender.

Analisando a Figura 5.6, é possível perceber que todas as bibliotecas e *frameworks* que foram utilizados no projeto estão nas extremidades do diagrama, ou seja, podemos trocar qualquer uma delas sem precisar mudar todo o projeto.

Agora vamos discutir alguns aspectos relacionados à implementação das principais funcionalidades do sistema.

5.2.1 Cadastro de professores

Ao acessar a página de cadastro, o cliente faz uma requisição HTTP do tipo *post* para o servidor. Essa requisição contém todos os dados do formulário de cadastro no formato *json*.

Ao chegar no servidor essa requisição é interceptada pelo *express* e enviada para o adaptador responsável por pegar somente os dados que importam. Isso ocorre pois não queremos que a nossa aplicação seja dependente dos dados do *express*, por isso pegamos somente o conteúdo da requisição. Depois de transformada, essa requisição é passada para o controlador responsável por implementar a lógica de cadastro. Antes de adicionar o usuário no banco de dados é verificado se o email é válido. Se o email for válido, a senha informada é criptografada (com a biblioteca *bcrypt*) e o usuário é adicionado no banco de dados. Vale ressaltar que todo usuário que se cadastrar no sistema é adicionado como um professor. Após realizar o cadastro no banco de dados, a classe responsável por fazer a autenticação do usuário no sistema é chamada. Essa classe de autenticação recebe o *login* e a senha do usuário e retorna um *token* de acesso que será o identificador do cliente enquanto este estiver conectado.

5.2.2 Login de usuários

De forma semelhante ao que acontece com o cadastro dos professores, no *login* a requisição também é interceptada pelo *express* e enviada a um adaptador. O que difere é que na autenticação o email do usuário é buscado no banco de dados. Se existir, a senha enviada na requisição é comparada com a senha que foi cadastrada. Se todas as informações estiverem corretas, um *token* de acesso é retornado.

5.2.3 Cadastro de salas

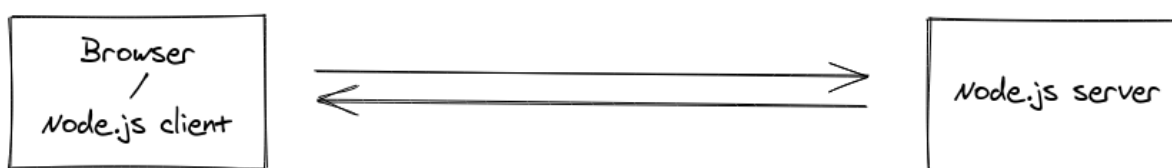
No cadastro de salas, a forma como as requisições são tratadas até chegar ao controlador responsável por implementar a lógica é igual ao que acontece no cadastro de usuários. E no *login* (acontece o mesmo para todas as requisições HTTP) a diferença é que, além de cadastrar a sala no banco de dados, o controlador responsável por implementar a lógica dessa rota também é responsável por cadastrar os usuários convidados no banco de dados e chamar um serviço que faz o envio de email automático para todos esses convidados. Quando a requisição chega ao controlador, é verificado se já existe uma sala cadastrada com aquele nome para aquele usuário. Caso ainda não exista, a lista de convidados é percorrida e todos os convidados, que ainda não são usuários, são cadastrados no sistema. Logo após esse cadastro, um email é enviado para todos os convidados da sala através da biblioteca

nodemailer, e o sistema cadastra a sala no banco de dados, colocando na lista de convidados somente o identificador de cada usuário.

5.2.4 Sala de jogos

Sem dúvidas a principal funcionalidade da plataforma é a sala de jogos, não só pela complexidade mas por ser o principal diferencial do sistema proposto aqui. Para que tudo aconteça em tempo real, é preciso que o protocolo de *sockets* da *web* (*websockets*) seja utilizado. Conforme discutido anteriormente, o servidor precisa expor duas interfaces de comunicação, uma via HTTP para as funcionalidades de cadastro, e outra via *sockets* para comunicação em tempo real. Para que as atualizações do tabuleiro, mensagens de texto e vídeo sejam transmitidas entre os clientes, o servidor precisa ficar responsável por receber todas essas informações e enviar para todos os clientes conectados naquela sala. A biblioteca *socket.io* permite comunicação em tempo real, bidirecional e baseada em eventos, entre o navegador (cliente) e o servidor. Quando um cliente entra em uma sala de jogos, este tenta estabelecer uma conexão *websocket* com o servidor. Se a conexão for aceita um canal de comunicação *full-duplex* e de baixa latência, é criado entre o navegador e o servidor. A Figura 5.7, ilustra como é esse canal de comunicação.

Figura 5.7 – Canal de comunicação fornecido pela biblioteca *socket.io*.

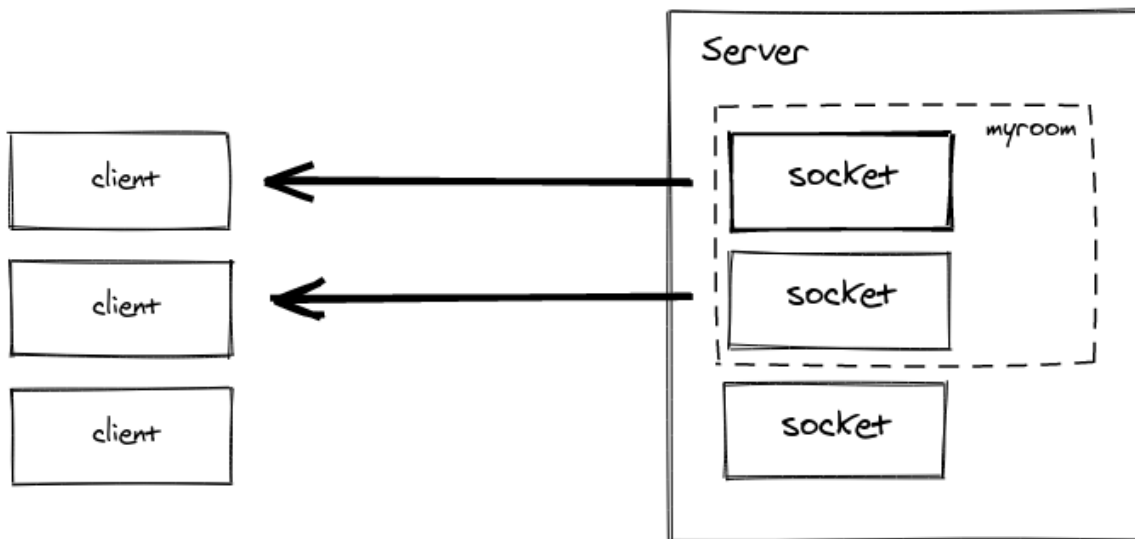


Fonte: ARRACHEQUESNE (Introduction: What Socket.IO is).

Para que o servidor seja capaz de atualizar as informações das salas a cada evento de cada cliente, este deve ter uma estrutura de dados em memória que seja responsável por guardar todas as informações referentes a uma sala quando esta estiver aberta. Enquanto uma sala de jogos estiver em pleno funcionamento, as informações mudam a todo instante. O estado do tabuleiro, a lista de mensagens e a transmissão de vídeo são atualizadas em uma frequência muito alta. Portanto, essas informações só devem ser salvas no banco de dados quando a sala for fechada.

Com o que foi explicado até aqui, você deve estar se perguntando como fazer a distinção das salas? Já que todos os clientes se conectam no mesmo servidor, como distinguir para qual cliente, de qual sala, replicar as mensagens? Para isso, a biblioteca *socket.io* disponibiliza um canal arbitrário no qual os *sockets* podem entrar e sair, denominado *Rooms*. As *Rooms* permitem que eventos sejam enviados para um subconjunto de clientes. A Figura 5.8 mostra a ideia básica das *Rooms*.

Figura 5.8 – Funcionamento das *Rooms*.



Fonte: ARRACHEQUESNE (Rooms).

Por ser orientado a eventos, a biblioteca *socket.io* permite definir um nome para cada evento que o servidor espera receber pelo canal de comunicação com o cliente. Por exemplo, quando o cliente realiza alguma ação no tabuleiro, um evento é emitido para o servidor e enviado para a camada responsável por tratar essas mensagens. A Tabela 5.1 mostra os eventos que o servidor espera receber e qual resposta é emitida para os clientes conectados no canal de comunicação.

Tabela 5.1 - Lista de eventos que o servidor espera receber.

Esperando eventos do tipo	Eventos de retorno
<i>Register</i>	<i>NewUser</i>
<i>UpdateBoard</i>	<i>NewMove</i>
<i>NewChatMessage</i>	<i>UpdateChat</i>
<i>CleanBoard</i>	<i>NewMove</i>

<i>Unregister</i>	<i>GuestDisconnect</i>
<i>SignVideo</i>	<i>AllUsersVideo</i>
<i>SendingSignal</i>	<i>UserJoined</i>
<i>ReturningSignal</i>	<i>ReceivingReturnedSignal</i>
<i>LeaveRoom</i>	<i>LeaveRoom</i>

Fonte: Próprio autor.

Quando uma sala de jogos é criada, aberta ou acessada, o cliente estabelece uma conexão via *socket* com o servidor. Esta conexão cria um canal de comunicação bidirecional que é reconhecido pelo identificador do *socket*. Após a conexão ser estabelecida, o cliente envia um evento do tipo *Register* passando como parâmetro o seu nome de usuário, o identificador da sala e o seu *token* de acesso. Ao receber esse evento o servidor salva as informações em memória (em um *array* de salas abertas) e cria uma *Room* para essa sala, colocando como nome da *Room* o identificador que foi enviado pelo cliente. Após atualizar as estruturas de dados internas, o servidor envia, somente para o cliente que se conectou, o estado atual de todos os dados da sala (usuários conectados, estado do tabuleiro e lista de mensagens do chat). Além disso, um evento *NewUser* é enviado para todos os clientes conectados na *Room*, contendo a lista de usuários atualizada.

Toda vez que um cliente mexe uma peça do tabuleiro, esta ação tem que ser replicada para todos os clientes conectados nesta sala. Para isso, um evento do tipo *UpdateBoard* (atualiza tabuleiro) é enviado para o servidor. Ao receber este evento, o servidor atualiza as estruturas de dados e emite essa informação para todos os clientes através do evento *NewMove*. Algo parecido acontece quando um cliente envia uma mensagem de texto através do chat ou clica em limpar o tabuleiro. Em ambos os casos um evento é enviado ao servidor, as estruturas de dados são atualizadas e um evento contendo os dados atualizados é emitido para todos os clientes.

Para se desconectar de uma sala, o usuário do tipo aluno envia um evento *Unregister* para o servidor. Este evento faz com que o usuário seja deletado da lista de usuários conectados e o *socket* seja retirado da *Room*. Além disso, um evento do tipo *GuestDisconnect* é emitido para todos os usuários conectados avisando que um cliente saiu da sala. Para usuários do tipo professor, além da possibilidade de sair da sala emitindo um evento do tipo *Unregister*, também é disponibilizada a possibilidade de fechar a sala. A ideia é que, futuramente, quando um professor optar por fechar uma sala, os dados gerados enquanto a sala estiver aberta sejam gravados no banco de dados. Para a funcionalidade de fechar a sala, o evento *LeaveRoom* é esperado e um evento do mesmo tipo é enviado para todos os usuários conectados informando que a sala foi fechada.

Um dos grandes desafios desse produto foi a criação da transmissão de vídeo em tempo real. Na Seção 5.3, o lado do cliente é detalhado e a transmissão de vídeo fica mais clara, porém, do ponto de vista do servidor é a mesma lógica dos outros eventos recebidos. Quando um usuário decide entrar na chamada de vídeo um evento *SignVideo* é recebido pelo servidor. Este evento possui os dados do usuário que quer entrar na chamada e o identificador da sala. A partir desses dados, a lista de usuários conectados com vídeo é atualizada e enviada para o cliente que fez a requisição. O servidor espera mais dois eventos relacionados à transmissão de vídeo, um deles é o *SendingSignal* e o outro é o *ReturningSignal*. Os dois eventos servem para ajudar os clientes a estabelecerem a conexão de vídeo entre si.

5.3 Cliente

Conforme discutido na Seção 5.1, o cliente responsável por fazer requisições ao servidor é uma aplicação *web* construída utilizando o *Reactjs*, que é uma biblioteca para criar interfaces de usuário. Criada pelo *Facebook*, essa biblioteca tem como objetivo otimizar as operações que acontecem na árvore de elementos do HTML. Além disso, é declarativa e permite criar componentes encapsulados e que gerenciam seu próprio estado. Essas características tornam o *Reactjs* bastante fácil de ser usado e muito performático.

A arquitetura utilizada para o lado do cliente segue o mesmo padrão do servidor. A principal diferença está na camada de apresentação (*presentation*), pois ao invés de trabalhar com controladores e serviços, no lado do cliente essa camada é responsável por criar os componentes de tela que serão utilizados pelo usuário.

A comunicação com o servidor é feita utilizando duas bibliotecas. Para a comunicação via HTTP foi utilizado o *axios* que é um cliente HTTP para fazer requisições a API, já para a comunicação via sockets foi utilizado a biblioteca *socket.io-client*. Para as funcionalidades de cadastro de usuários, salas e *login*, todas as informações fornecidas pelo usuário nos formulários são encapsuladas em um objeto no formato *json* e enviadas para o servidor através de uma requisição feita pela biblioteca *axios*. Assim como no servidor, no cliente as camadas mais internas não dependem de *frameworks* ou bibliotecas, portanto, para as bibliotecas utilizadas foram criados adaptadores de interface.

Quando o cliente acessa uma sala, é de sua responsabilidade iniciar uma conexão via *socket* com o servidor. Para isso, uma conexão é feita chamando o método *connect* e o evento *Register* é enviado ao servidor. A Tabela 5.2 mostra a lista de eventos que o cliente espera receber.

Tabela 5.2 - Lista de eventos que o cliente espera receber.

Esperando eventos do tipo
<i>NewUser</i>
<i>NewMove</i>
<i>UpdateChat</i>
<i>GuestDisconnect</i>
<i>AllUsersVideo</i>
<i>UserJoined</i>
<i>ReceivingReturnedSignal</i>
<i>LeaveRoom</i>

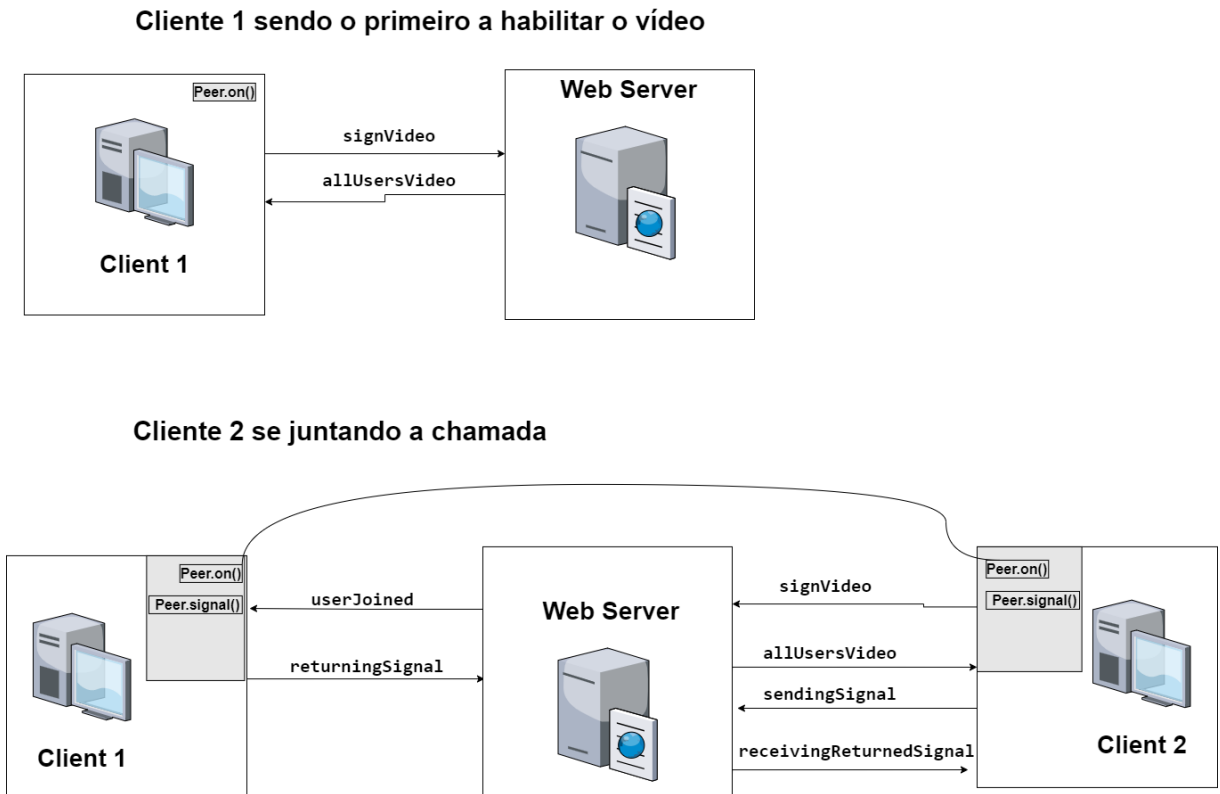
Fonte: Próprio autor.

Os eventos do tipo *NewUser*, *NewMove* e *UpdateChat* são eventos que atualizam os componentes que mostram a lista de usuários, o tabuleiro e as mensagens do *chat*, respectivamente. O evento *GuestDisconnect* também serve para atualizar a lista de usuários conectados, porém, esse ocorre quando um usuário sai da sala. O evento *LeaveRoom* ocorre quando um usuário professor fecha a sala. Ao receber esse evento o cliente é redirecionado para o seu painel de usuário.

Conforme discutido na Seção 5.2, a transmissão de vídeo para o servidor segue o mesmo fluxo dos outros eventos, porém, para o cliente a lógica é um pouco diferente das outras comunicações. Quando o usuário clica no botão “Entrar na chamada de vídeo”, o cliente pede permissão para acessar a *webcam* do usuário. Se esse pedido for aceito, o *stream* da câmera do cliente é colocado dentro de um componente na interface gráfica. Logo após, o cliente envia um evento *SignVideo* para informar ao servidor que ele se conectou na chamada. Toda vez que o servidor recebe essa mensagem ele retorna com um evento do tipo *AllUsersVideo* contendo a lista de usuários que estão conectados na chamada de vídeo. A partir desse evento, essa lista é percorrida e um *Peer* é criado para cada usuário. Para fazer essa transmissão, a biblioteca *simple-peer* foi utilizada. Essa biblioteca é responsável por criar uma conexão *WebRTC*. O *WebRTC* (*web real time communications*) é um projeto de código aberto que tem como objetivo fornecer comunicação em tempo real para navegadores e dispositivos móveis, onde a transmissão de voz, vídeo e texto são permitidos. Portanto, para cada usuário que está com o vídeo conectado, o cliente, que recém se conectou, cria um canal de comunicação de voz e vídeo diretamente com os usuários. Esse tipo de comunicação é chamada de *peer-to-peer* (par a par), onde o cliente se conecta direto com outro cliente. A biblioteca *simple-peer* disponibiliza um método responsável por ouvir eventos, denominado *on*, que sabe quando um par deseja enviar dados, e um método chamado *signal* que serve para

realizar a conexão. A Figura 5.9 mostra a troca de mensagens existentes entre cliente, servidor e os *Peers*.

Figura 5.9 – Comunicação entre cliente, servidor e *Peers* para transmissão de vídeo.



Fonte: Próprio autor.

Na parte superior da Figura 5.9 vemos o que acontece quando um cliente entra em uma chamada de vídeo. Conforme discutido acima, este cliente diz ao servidor que quer acessar a chamada e recebe a lista de todos os usuários que estão transmitindo vídeo (no caso da imagem o cliente 1 é o primeiro a entrar na chamada). Quando o cliente 2 entra na chamada, conforme mostra a parte inferior da Figura 5.9, este cliente diz ao servidor que se conectou e recebe a lista de usuários na chamada. Ao receber essa lista, que contém o identificador do *socket* do cliente 1, o cliente 2 cria um *Peer* com o cliente 1 e, a partir desse momento, um evento do tipo *SendingSignal* é enviado ao servidor contendo o identificador do usuário que vai receber o vídeo. O servidor emite ao cliente 1 um evento do tipo *UserJoined* indicando que um novo cliente entrou na chamada. Com isso, o cliente 1 adiciona na tela o *stream* do cliente 2 e retorna ao servidor um evento *ReturningSignal* informando que recebeu o sinal com sucesso. Ao receber esse evento, o servidor emite um evento *ReceivingReturnedSignal* para o cliente 2, que adiciona o *stream* do cliente 1 em tela.

5.4 Principais desafios do projeto

No desenvolvimento deste ambiente, diversos padrões e técnicas de desenvolvimento foram utilizados. Como a ideia é ter um produto a partir dos conhecimentos adquiridos com este trabalho, o desenvolvimento da plataforma levou em consideração que futuramente outras pessoas podem mexer no código. Portanto, ter um sistema desacoplado, separado em camadas, seguindo os conceitos da *clean architecture* e do *clean code* é extremamente importante. Um dos pilares da *clean architecture* é a testabilidade, ou seja, as camadas são separadas em diferentes responsabilidades e os testes são utilizados para que possamos ter uma maior segurança na hora das modificações e no desenvolvimento de novas funcionalidades. Para realizar o desenvolvimento foi utilizado o TDD (*test driven development*), que é uma técnica de desenvolvimento de software que visa desenvolver os testes antes das funcionalidades da aplicação, ou seja, primeiro o desenvolvedor escreve um teste automatizado e depois produz o código para que o teste seja validado. Esta técnica trouxe uma segurança na hora do desenvolvimento, pois, quando os testes passam, temos a certeza de que a funcionalidade está de acordo com o que foi proposto no teste. Porém, por não ter experiência na aplicação desse método, o processo de desenvolvimento tornou-se mais lento, fazendo que essa técnica fosse abandonada para que o prazo pudesse ser cumprido. Outra dificuldade encontrada no processo de desenvolvimento foi a troca de mensagens em tempo real, via *websockets*, que a plataforma faz. A cada mensagem que é trocada o sistema precisa validar o usuário, verificar se o cliente é um participante da sala de jogos, verificar de qual sala veio a mensagem, manter todas essas informações em memória e ainda replicar essas mensagens para todos os clientes que estão conectados (em determinada sala). Com o passar do tempo, a implementação tornou-se complexa e difícil de ser entendida, fazendo que diversas refatorações fossem feitas.

Em relação à aplicação do ambiente no ensino, existem alguns obstáculos que precisam ser superados para que o sistema discutido nesse trabalho seja amplamente utilizado. O primeiro deles é em relação a segurança dos dados dos participantes. Em razão da LGPD (lei geral de proteção de dados) todo software desenvolvido tem que se preocupar em como os dados serão armazenados, acessados, transferidos (entre cliente e servidor) e anonimizados. O desenvolvimento deste trabalho não levou em consideração as exigências da LGPD, portanto, antes do ambiente ser disponibilizado essas questões têm de ser superadas. Outro obstáculo

está relacionado a aceitação dos alunos. Muitas funcionalidades do sistema foram pensadas para proporcionar ao professor uma nova forma de aplicação do projeto LoBoGames. Porém, será que as funcionalidades do modo aluno estão de acordo com as necessidades desse tipo de usuário? Até que ponto a interface gráfica é fácil e intuitiva para crianças e adolescentes do ensino fundamental? Essas perguntas só serão respondidas a partir do momento que o ambiente virtual for realmente utilizado.

Quando pensamos em próximos passos, fica claro que a disponibilização das métricas é uma das principais funcionalidades a serem desenvolvidas, todos os dados que são gerados em uma sala de jogos precisam passar por um processo de análise e mineração, para que padrões possam ser encontrados, ajudando o professor a entender melhor seus alunos.

6 CONCLUSÃO

A educação sempre foi um dos pilares para o desenvolvimento de qualquer civilização. Frequentar a escola, interagir com os colegas e ser exposto a coisas novas faz com que o indivíduo evolua não só em relação ao conhecimento, mas também em relação ao convívio em sociedade. Quando um indivíduo não é exposto a esse ambiente, o seu desenvolvimento fica completamente prejudicado. Sendo assim, o fechamento das escolas por conta da pandemia traz um atraso na alfabetização das crianças, fazendo com que os reflexos sejam sentidos daqui alguns anos. Portanto, o desenvolvimento de ambientes virtuais que permitam que a alfabetização continue, mesmo em momentos de isolamento social, se faz necessário.

O presente trabalho idealizou e desenvolveu uma plataforma que ajuda os professores a trazerem o lúdico para o ensino remoto, tendo como base a metodologia de ensino do projeto LoBoGames. O ambiente virtual proposto aqui dispõe de diversas métricas e possibilidades que auxiliam os professores. Com o MVP desenvolvido foi possível perceber que existem tecnologias disponíveis que permitem que uma plataforma tenha mensagens sendo trocadas através do *chat*, transmissão de vídeo em tempo real e um tabuleiro interativo onde todos os usuários conectados podem mexer. O MVP também mostrou que existe a possibilidade da plataforma ser utilizada não só como ferramenta de ensino, mas também como ferramenta de diversão, interação e trabalho colaborativo.

Tendo esse MVP em mãos, o próximo passo é testá-lo com crianças do ensino fundamental e verificar possíveis pontos de melhoria. Além disso, o desenvolvimento das métricas também deve ser feito para trazer uma experiência mais completa para os professores.

REFERÊNCIAS

About HTML5 WebSocket. Disponível em <https://www.websocket.org/aboutwebsocket.html>. Último acesso em 20 Abril 2021.

ALMEIDA, Maria Elizabeth Bianconcini de. **Educação a distância na internet: abordagens e contribuições dos ambientes digitais de aprendizagem.** Educ. Pesqui., São Paulo , v. 29, n. 2, p. 327-340, Dec. 2003.

ARRACHEQUESNE, Damien. **Rooms.** Disponível em <https://socket.io/docs/v3/rooms/>. Último acesso em 01 Maio 2021.

ARRACHEQUESNE, Damien. **Introduction: What Socket.IO is.** Disponível em <https://socket.io/docs/v4/>. Último acesso em 10 Maio 2021.

BERNERS-LEE, T. **Hypertext Transfer Protocol -- HTTP/1.1.** Publicado em Junho 1999. Disponível em <https://datatracker.ietf.org/doc/html/rfc2616>. Último acesso em 15 Março 2021.

BRITO, Ronnie Fagundes de; PEREIRA, Alice Theresinha Cybis. **Um estudo para ambientes colaborativos e suas ferramentas.** Florianópolis. Congresso Nacional de Ambientes Hiperídia para Aprendizagem, Florianópolis, junho 2004.

COMUNICAÇÃO em tempo real para a web. Disponível em <https://webrtc.org/>. Último acesso em 30 Abril 2021.

FUCKS, H., RAPOSO, A.B & GEROSA, M.A. **Engenharia de groupware: Desenvolvimento de aplicações colaborativas.** Rio de Janeiro. XXI Jornada de atualização em informática 2002.

GONÇALVES, Mariana Sbaite, **Mapeamento de dados pessoais: o coração do projeto!**. Publicado em 01 Março 2021. Disponível em <https://www.lgpdbrasil.com.br/mapeamento-de-dados-pessoais-o-coracao-do-projeto-lgpd/>. Último acesso em 10 Maio 2021.

GOUVEIA, LMB. **Ambientes Virtuais Colaborativos: a procura de formas alternativas de interação.** Revista Politécnica Retrieved nº2. 2000.

HUIZINGA, J.; **Homo Ludens: o jogo como elemento da cultura.** São Paulo: Editora USP, 2010.

KUROSE, James F. **Computer networking : a top-down approach.** 6. ed. Londres: Pearson, 2013.

LOBOGAMES, **Jogos Lógicos de Tabuleiro.** Programa de Extensão, UFRGS. Disponível em <http://www.inf.ufrgs.br/lobogames/>. Último acesso em 21 abril 2021.

MACHADO, Glaucio José Couri. **Refletindo sobre a interação social em ambientes virtuais de aprendizagem**. Porto Alegre, v. 3, n. 1, p. 1-10, maio 2005. Disponível em: <https://seer.ufrgs.br/renote/article/view/13726>. Acesso em: 10 fev. 2021.

MARTIN, Robert C. **Código limpo: habilidades práticas do Agile Software**. 1. ed. Rio de Janeiro: Alta Books, 2009

MARTIN, Robert C. **Arquitetura limpa: o guia do artesão para estrutura e design de software**. 1. ed. Rio de Janeiro: Alta Books, 2019

MARTIN, Robert C., **The Clean Architecture**. Publicado em 13 agosto 2012. Disponível em <<https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html>>. Último acesso em 31 março 2021.

MIRASHE, Dr-Shivaji & Kalyankar, N. **Peer-to-Peer Network Protocols**. 2010

NEOTERIC, **Single-page application vs. multiple-page application**. Publicado em 02 Dezembro 2016. Disponível em <<https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58>>. Último acesso em 05 Março 2021.

PACHECO, José. **Aprender em comunidade**. São Paulo : Edições SM, 2014.

PRENSKY, Marc. **Aprendizagem baseada em jogos digitais**. São Paulo: Editora Senac São Paulo, 2012.

RAPOSO, A. B. Ambientes Virtuais Colaborativos. In: Pimentel, M; Fuks, H. **Sistemas Colaborativos**. Rio de Janeiro: Elsevier, 2011

RIBAS, Renato Perez. **Jogos lógicos de tabuleiro como instrumento pedagógico de socialização e emancipação**. In: Congresso Brasileiro de Extensão Universitária (CBEU), 8., 2018, Natal, Rn. Congresso.

RIBAS, Nilseia Lapresa. **Jogos de tabuleiros movimentando a escola**. 2019. 76 f. TCC (Graduação) - Curso de Pedagogia, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2019.

RUSSEL, Aaron, **What is HTTPS?**. Publicado em 13 Julho 2020. Disponível em <<https://www.ssl.com/faqs/what-is-https/>>. Último acesso em 10 Março 2021.

TRANSMISSION control protocol. Publicado em Setembro 1981. Disponível em <<https://datatracker.ietf.org/doc/html/rfc793/>>. Último acesso em 02 Abril 2021.