UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA

CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

LUIS GUSTAVO DE CASTRO SOUSA

UMA ARQUITETURA WEB PARA IMPLEMENTAÇÃO DE JOGOS MULTIPLAYER INCLUSIVOS

FORTALEZA

2023

LUIS GUSTAVO DE CASTRO SOUSA

UMA ARQUITETURA WEB PARA IMPLEMENTAÇÃO DE JOGOS MULTIPLAYER INCLUSIVOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia De Computação do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia De Computação.

Orientador: Prof. Dr. José Marques Soares.

FORTALEZA

2023

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

S697a Sousa, Luis Gustavo de Castro.
Uma Arquitetura Web Para Implementação de Jogos Multiplayer Inclusivos / Luis Gustavo de Castro Sousa. – 2023.
49 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Centro de Tecnologia, Curso de Engenharia de Computação, Fortaleza, 2023.
Orientação: Prof. Dr. José Marques Soares.

1. Jogos. 2. Acessibilidade. 3. Inclusão. I. Título.

CDD 621.39

LUIS GUSTAVO DE CASTRO SOUSA

UMA ARQUITETURA WEB PARA IMPLEMENTAÇÃO DE JOGOS MULTIPLAYER INCLUSIVOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia De Computação do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia De Computação.

Aprovada em: 11/07/2023.

BANCA EXAMINADORA

Prof. Dr. José Marques Soares (Orientador)
Universidade Federal do Ceará (UFC)

Eng. Dr. Fábio Cisne Ribeiro
Laboratório de Engenharia de Sistemas de Computação (LESC/DETI)

Prof. Dr. Edilson Rocha Porfírio Filho
Universidade Federal do Ceará (UFC)

À minha família, por todo apoio e sustento me dado desde a infância. Mãe, seu cuidado e preocupação sempre me deram conforto e tranquilidade para alcançar meus sonhos. Pai, sua presença e caráter sempre me inspiraram a ser um homem melhor.

AGRADECIMENTOS

A minha família, meu Pai e Mãe, por todo apoio e compreensão durante a graduação.

Ao meu amigo e co-autor no projeto que envolve esse trabalho Aluísio Alves, pelos esforços e parceria durante toda a graduação.

Ao Billy, meu cachorro e companheiro há 20 anos, que faleceu durante a escrita deste trabalho.

A todos meus amigos do grupo "Segue a Call", Iury, Tayllan, Helena, Michel, Vinícius, Matheus, Tomás, Pedro, Laura e Vanderley, pela amizade e apoio durante todo o curso de Engenharia de Computação.

Ao Prof. Dr. José Marques Soares, pela excelente orientação.

Aos professores participantes da banca examinadora Eng. Dr. Fábio Cisne Ribeiro e Prof. Dr. Edilson Rocha Porfírio Filho pelo tempo, pelas valiosas colaborações e sugestões.

A todos voluntários que participaram das sessões de teste desse trabalho.

"Amarás o teu próximo como a ti mesmo."

(Mateus 22:39)

RESUMO

Com o passar do tempo, os videogames têm ocupado um espaço cada vez mais importante na nossa sociedade. Porém, a falta de acessibilidade desse tipo de aplicação tem impedido o acesso de muitos indivíduos a benefícios cognitivos e sociais que podem derivar de seu uso. Apesar de grandes empresas estarem desenvolvendo jogos acessíveis, não há ferramentas ou guias que facilitem de maneira objetiva o processo de desenvolvimento. Assim, é proposto neste trabalho uma arquitetura para a construção de um jogo de perguntas e respostas *online*, *multiplayer* e acessível. O paradigma adotado visa facilitar o desenvolvimento de múltiplas interfaces, sejam elas acessíveis ou convencionais, para um mesmo jogo, buscando-se proporcionar um ambiente inclusivo aos usuários. Desse modo, foi concebido um componente referido como núcleo lógico funcional que foi adotado para a construção. O trabalho integra um projeto maior para construção de um jogo que considera uma interface para deficientes visuais, além de uma interface convencional. O desenvolvimento do núcleo lógico funcional foi feito de forma iterativa e incremental, com diversas etapas de validação por diferentes usuários. As melhorias foram integradas à plataforma após a análise dos *feedbacks* e das avaliações produzidas a partir dos testes realizados com voluntários. Ao final do processo de desenvolvimento descrito neste trabalho, foi possível alcançar médias de avaliação de 4,86/5 em um teste com uma pessoa cega, e 4,61/5 em um teste com diversos usuários que jogaram múltiplas partidas simultaneamente.

Palavras-chave: Jogos; Acessibilidade; Inclusão

ABSTRACT

Over time, video games have occupied an increasingly important space in our society. However, the lack of accessibility of this type of application has prevented many individuals from accessing the cognitive and social benefits that may derive from its use. Although large companies are developing accessible games, there are no tools or guides that objectively facilitate the development process. Thus, this work proposes an architecture for the construction of an accessible online, multiplayer game of questions and answers. The paradigm adopted aims to facilitate the development of multiple interfaces, whether accessible or conventional, for the same game, seeking to provide an inclusive environment for users. Thus, a component referred to as functional logic core was conceived and adopted for construction. The work is part of a larger project to build a game that considers an interface for the visually impaired, in addition to a graphical interface. The development of the functional logical core was done in an iterative and incremental way, with several stages of validation by different users. The improvements were integrated into the platform after analyzing the feedback and evaluations produced from the tests carried out with volunteers. At the end of the development process described in this work, it was possible to reach evaluation averages of 4.86/5 in a test with a blind person, and 4.61/5 in a test with several users who played multiple games simultaneously.

Keywords: Games; Accessibility; Inclusion

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

HTTP	*Hypertext Transfer Protocol* ou Protocolo de Transferência de Hipertexto
MVC	*Model, View, Controller*
PAAS	Platform as a Service ou Plataforma como Serviço
RF	Requisitos Funcionais
UFC	Universidade Federal do Ceará
UUID	*Universally Unique Identifier* ou Identificador Único Universal

SUMÁRIO

1 INTRODUÇÃO

Videogames têm se tornado cada vez mais presentes culturalmente. Eles têm cada dia mais se transformado em um fator cultural importante enquanto se revelam bons coadjuvantes para auxiliar tratamento de doenças, bem como no desenvolvimento emocional e cognitivo de seus jogadores (GROSSI, 2017; TOH; KIRSCHNER, 2022). Contudo, nem todos têm acesso aos videogames devido à natureza do produto. Em grande parte dos casos, os jogos são projetados presumindo que os jogadores possuem capacidades mínimas suficientes de visão e coordenação motora, entre outras. Com isso, muitos jogos se tornam inacessíveis para jogadores que possuem alguma deficiência que impacte as capacidades que foram presumidas pelos desenvolvedores.

O contexto atual da acessibilidade em videogames é reconhecido pelo público geral que tem demandando maior inclusão e acessibilidade (Interactive Games and Entertainment Association, 2022). A acessibilidade determina, portanto, a capacidade dos videogames poderem ser aproveitados por indivíduos que não conseguiriam antes desfrutar desses recursos, sendo imprescindível para trazer a inclusão digital ao cenário de jogos mundial.

Fomentando essa tendência, grandes desenvolvedoras do cenário mundial têm desenvolvido jogos com mais atenção à acessibilidade como é o caso de *The Last of Us Part II* desenvolvido pela *Naughty Dog* (HöFKE, 2022). Contudo, casos como esse são minoria entre a maior parte dos estúdios e jogos acessíveis não são comuns no mercado de videogames. Percebe-se, portanto, que a industria enfrenta grande dificuldade de desenvolver e viabilizar a acessibilidade em seus jogos.

A acessibilidade nos jogos populares atuais se dá, em grande parte, através de ferramentas adaptativas sobre o projeto inicial. Dentre elas podemos citar dicas sonoras, ferramentas de alto contraste, ferramentas de *feedback* tátil e leitores de tela, como visto em *The Last Of Us II* (GALLANT, 2020). Tal forma de pensar o desenvolvimento de acessibilidade não consegue abranger uma ampla variedade de deficiências e, em geral, encontra limitações e barreiras quanto ao custo e prazo do projeto.

Assim, na forma atual de produzir jogos, é notável que a acessibilidade, invariavelmente, adicionará uma maior complexidade e custo ao projeto (BIERRE *et al.*, 2005). Com tais barreiras, podemos imaginar que pequenas desenvolvedoras podem ficar sem recursos para desenvolver jogos acessíveis. Por outro lado, grandes estúdios e investidores são afugentados pelo acréscimo no orçamento de seus projetos. Com isso, abre-se espaços para proposições que facilitem e viabilizem o desenvolvimento de jogos que levam a acessibilidade e a inclusão como

princípio.

1.1 Objetivos

Este trabalho foi desenvolvido de forma concomitante e cooperativa com outro, constituindo um projeto maior. As duas partes se integram para oferecer ao usuário um jogo *mobile*, *multiplayer* de perguntas e respostas com múltiplas interfaces acessíveis.

A contribuição proposta neste trabalho busca uma mudança de paradigma no desenvolvimento de jogos. Nesta nova forma, um componente centralizador será inserido na arquitetura do jogo e agirá como núcleo lógico funcional ao abstrair a lógica de jogo, seus estados e isola-los das possíveis interfaces. Ele será desenvolvido de forma desacoplada das suas interfaces e, então, conectados a múltiplas interfaces, acessíveis ou não, desenvolvidas posterior ou concomitantemente com o núcleo do jogo.

Para isso, é desenvolvido como núcleo lógico funcional e parte central do jogo um componente, em que múltiplos jogadores, por meio de interfaces diferentes, consigam interagir em uma mesma partida, de forma integrada e inclusiva.

Durante esse trabalho, o componente central a ser desenvolvido será referido como "núcleo de jogo"ou "núcleo lógico funcional", sendo utilizado ambos os termos para se referir a ele.

Assim, surgem como objetivos principais deste trabalho: desenvolver um componente arquitetônico que desempenhe a função de núcleo lógico funcional de um jogo de perguntas e respostas, e que possa:

- Apresentar uma abordagem alternativa ao modelo vigente de desenvolvimento de jogos acessíveis, por meio do desacoplamento da interface da estrutura lógico-funcional;
- Conectar jogadores usuários de interfaces acessíveis e convencionais de forma competitiva e justa;
- Fornecer uma arcabouço de ferramentas que permita o desenvolvimento de múltiplas interfaces que sejam acessíveis ou convencionais;
- Isolar as estruturas lógico-funcionais de um jogo de perguntas e respostas acessível e inclusivo.

1.2 Organização do documento

O trabalho é dividido em 5 capítulos. Após esta introdução, é apresentada a fundamentação teórica, na qual se discute sobre o mercado atual de videogames, os benefícios dos jogos, a falta de acessibilidade e as dificuldades do desenvolvimento de jogos acessíveis. No capítulo 3 é apresentado o desenvolvimento da aplicação, que descreve a concepção de um jogo de perguntas e respostas, incluindo a infraestrutura e as tecnologias do núcleo de jogo utilizadas, os serviços expostos pelo núcleo, os modelos, os *endpoints*, a conexão bidirecional via *WebSockets*, os nomes de salas acessíveis, a construção conjunta e a melhoria incremental. No capítulo 4, os resultados são abordados, incluindo os testes de desempenho, os testes com usuários e a discussão de algumas considerações finais. Finalmente, no último capítulo são apresentadas as conclusões e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo será dissertado sobre o mercado de jogos e a importância do desenvolvimento de jogos acessíveis para pessoas com alguma deficiência. Falaremos sobre como os videogames ao longo das ultimas décadas têm se tornado um mercado gigantesco, impactando cada vez mais jogadores ao redor do mundo e se tornando uma parte relevante da nossa cultura e socialização. Abordaremos brevemente os benefícios dos jogos que vão além de simples entretenimento ou retorno financeiro para investidores. Ilustraremos sobre como eles trazem também uma camada social, educativa e reabilitativa que tem ganhado cada vez mais importância. Jogos podem servir como forma de socialização entre amigos e famílias, e também como poderosas ferramentas de educação para professores e escolas. Por fim, será discutido sobre a falta de acessibilidade nos jogos e as dificuldades de projetar jogos acessíveis para pessoas portadoras de alguma deficiência.

2.1 Mercado atual de videogames

Segundo a Entertainment Software Association (2022), apenas nos Estados Unidos, existem cerca de 215,5 milhões de jogadores ativos entre todas a idades e bilhões de jogadores mundiais, transcendendo idade, gênero, cultura e sociedades.

Com isso, o mercado de games mundial foi avaliado em 195.65 bilhões de dólares e tem um crescimento esperado de 12,9% ao ano até 2030. Além disso, os avanços tecnológicos, a expansão da internet e a fácil disponibilidade de jogos dentro da rede são fatores que contribuem ainda mais ao crescimento desse mercado (Grand View Research, 2020).

Vale ressaltar que, durante a pandemia de COVID-19, muito embora a economia global tenha sentido a recessão, a indústria de videogames demonstrou um bom crescimento. O *lockdown* e as restrições sanitárias impostas pela pandemia forçaram a população mundial a buscar atividades que possam ser realizas dentro de casa. Com isso, o numero de indivíduos que passaram mais tempo consumindo videogames aumentou.

Contudo, após a pandemia, o número de jogadores não reduziu de maneira considerável. O relatório da Entertainment Software Association (2022) afirma que 9 em 10 jogadores passam o mesmo tempo jogando atualmente do que passavam jogando no pico da pandemia. Tudo isso nos ajuda a compreender a imponência e robustez do mercado de videogames mundial.

2.2 Games muito além de entretenimento e lucro

Além de serem uma atividade economicamente viável a investidores e uma fonte de entretenimento aos consumidores, videogames têm desempenhando um papel social cada vez mais importante. Segundo estudos, eles têm demonstrado potencial para gerar impactos psicológicos e reabilitativos positivos aos seus jogadores, além de contribuírem com a promoção de interações sociais enriquecedoras.

O relatório da Entertainment Software Association (2022) demonstrou que 88% dos jogadores concordam que videogames podem unir diferentes tipos de pessoas, mostrando o potencial integrativo e social dos jogos. Além disso, o relatório registrou que 90% dos jogadores concordaram que os videogames podem criar experiências acessíveis para pessoas com diferentes níveis de habilidades.

Esses dados retirados da pesquisa com jogadores também encontram confirmação em estudos acadêmicos. Mercier e Lubart (2023) mostram uma correlação positiva entre a criatividade e a frequência de jogo, devido principalmente, ao desenvolvimento do otimismo do indivíduo. Segundo a conclusão de seus estudos, os videogames podem contribuir com a criatividade através dos pilares cognitivo, social, emocional e motivacional. Nesse sentido, os pesquisadores verificaram que jogar videogame pode aumentar o grau de otimismo de um indivíduo, e devido a isso, pode proporcionar maior criatividade no ambiente de trabalho.

Para Grossi (2017), os videogames se demonstraram como um tratamento coadjuvante viável na reabilitação de vítimas de acidente vascular cerebral. A pesquisadora demonstrou como o uso do jogo *Wii Fit* disponível na plataforma *Nintendo Wii*, em conjunto com um tratamento fisioterapêutico, foi eficaz no ganho de equilíbrio de pacientes com sequelas de acidentes vascular cerebral.

Somado a isso, Toh e Kirschner (2022) encontraram indícios de aprendizado de conceitos sócio-emocionais devido aos jogos, chegando a essa conclusão devido à análise de gravações, entrevistas e reflexões dos jogadores. A partir de suas observações, os autores sugeriram como professores podem usar o aprendizado baseado em jogos dentro de contexto pedagógico para gerar aprendizados sócio-emocionais.

Portanto, é promissora a capacidade dos videogames exercerem um papel social poderoso nos dias atuais ao influenciar positivamente diferentes áreas. Com eles é possível amplificar a criatividade de um indivíduo, contribuir para aprendizados socio-emocionais e também servir como ferramenta de reabilitação.

2.3 Os jogos e a falta de acessibilidade

Apesar de suas múltiplas potencialidades, nem todos possuem acesso aos jogos. Em grande parte, videogames não são desenvolvidos de maneira acessível a pessoas com deficiência e, dessa forma, frequentemente possuem uma única forma de serem experienciados.

Em geral, como descrito por Rollings e Morris (2009), podemos entender que a arquitetura de um jogo é projetada de forma acoplada a sua interface e gráficos. Jogos desenvolvidos em tal modelo pressupõem de seus jogadores um conjunto de habilidades e capacidades. Dessa forma, é previsível que potenciais dificuldades e barreiras, muitas vezes intransponíveis, sejam geradas e que impeçam que o conteúdo de um jogo desenvolvido em tal modelo seja aproveitado por usuários com alguma deficiência particular que impacte nas pressuposições definidas no projeto do jogo.

Nesse contexto, Bierre *et al.* (2005) mostrou que jogadores com deficiência frequentemente experienciam dificuldades em jogar devido à ausência de recursos de acessibilidade nos jogos. Isso sugere um cenário atual de videogames que, salvo por algumas exceções, não têm produzido jogos acessíveis e inclusivos a jogadores portadores de deficiências.

Somado a isso, os estudos de Porter e Kientz (2023) também dão indícios de que nem mesmo as ferramentas de acessibilidade comumente usadas por pessoas portadoras de deficiência visual, como as ferramentas de *text-to-speech*, têm sido suficientes para permitir um aproveitamento da experiência fornecida pelos jogos. De a cordo com a pesquisa, esses sistemas frequentemente não conseguem interpretar o que está sendo mostrado pela interface do jogo por incompatibilidades com o software.

Porter e Kientz (2023) discutiram, ainda, que certos gêneros e tipos de jogos tendem ser ainda menos acessíveis. Durante sua pesquisa, notaram que, para todos os participantes, jogos *online multiplayer*, apesar do crescimento constante de sua popularidade, eram aproveitados bem menos que jogos *singleplayer*. Isso pode indicar, segundo os pesquisadores, que há fatores inerentes ao *multiplayer* que afastam jogadores com alguma deficiência.

Devido a essa falta de inclusão, desenvolvedores e jogadores ao redor do mundo já começam a reivindicar mais acessibilidade nos videogames. Uma pesquisa conduzida pela Interactive Games and Entertainment Association (2022) mostrou que adultos australianos ‘concordam’ ou ‘concordam fortemente’ que há uma grande necessidade de maior diversidade representativa nos jogos, com 60% dos entrevistados afirmando que acessibilidade e inclusão devem ser consideradas no projeto de um jogo. O trabalho deixa claro que a demanda por

jogos mais acessíveis tem crescido ao redor do mundo e tem se tornado um tema relevante em convenções e eventos dentro do mundo dos jogos.

2.4 As dificuldades do desenvolvimento de jogos acessíveis

Desenvolver jogos é uma tarefa complexa. Blow (2004) demonstra diversos aspectos que tornam o desenvolvimento de jogos uma tarefa difícil e como o mercado permanece permanentemente carente de mão de obra qualificada para o desafio.

Nesse contexto, quando adicionamos a acessibilidade, tornamos o desafio de desenvolver um jogo ainda maior. Segundo Kulik *et al.* (2021) as barreiras mais comuns encontradas pelos desenvolvedores para o desenvolvimento de jogos acessíveis são:

- Falta de dinheiro e tempo para focar na acessibilidade;
- Falta de ferramentas de desenvolvimento;
- Falta de clareza sobre a acessibilidade;
- Falta de interesse do estúdio sobre o tema;
- Preocupação de que a acessibilidade ira prejudicar a intenção do projeto;
- Pouco entendimento sobre a acessibilidade.

Com isso, desenvolver um jogo acessível tem se mostrado um grande desafio para a indústria dos jogos. Esses desafios são de inúmeros tipos e vão desde encontrar estúdios e executivos que enxerguem o valor e a importância de tornar seus jogos acessíveis, a ter verba suficiente e mão de obra especializada. De fato, a natureza técnica do desafio torna o desenvolvimento complexo. Devido a esses fatores estima-se que poucos estúdios, inclusive dentre os de grande porte, têm tentado sobrepor tais desafios.

Corroborando a essa lógica, Bierre *et al.* (2005) conclui que, como o retorno sobre o investimento de um jogo é crucial para a industria, a acessibilidade deve estar alicerçada em um contexto financeiro realista. Caso contrário, a acessibilidade corre o risco de não ser implementada nos jogos em massa.

Um exemplo importante na industria, é o jogo *The Last of Us Part II*. Trata-se de um jogo de ação, aventura e drama desenvolvido pela *Naughty Dog*, publicado pela *Sony Interactive Entertainment*, lançado em 19 de junho de 2020 e que se tornou uma das maiores referências em acessibilidade em jogos. O estúdio conseguiu feitos impressionantes ao projetar um jogo em ambiente 3D, que oferece ao jogador um alto grau de liberdade, enquanto permite que pessoas com deficiência visual severa interajam através de amplas opções de acessibilidade e novas

formas de navegar no mundo do game. Ao todo, o jogo contou com mais de 60 opções de acessibilidade (GALLANT, 2020). Um exemplo de suas funções de acessibilidade pode ser observado na Figura 1

Figura 1 – *The Last of Us Part II* com as opções de alto contraste para jogadores com deficiência visual grave.

Fonte: https://blog.br.playstation.com/2020/06/09/the-last-of-us-part-ii-opcoes-de-acessibilidade-detalhadas/. Acessado em 16/06/2023

O resultados dos esforços do estúdio renderam prêmios de inovação em acessibilidade, melhor *design* de som, melhor jogo de ação e aventura, e melhor jogo do ano de 2020 na *The Game Awards*, a maior premiação anual da indústria de jogos. Além disso, importantes depoimentos de jogadores portadores de deficiência já foram feitos sobre o jogo. (HöFKE, 2022)

Contudo, o exemplo de *The Last of Us Part II* não é fácil de ser seguido. É importante citar que a inovação deste game na área contou com um grande investimento, utilizando código e padrões proprietários. Estima-se que o jogo tenha custado mais de 100 milhões de dólares (SAKELLARIOU, 2020). *The Last of Us Part II* é um exemplo de jogo altamente acessível, mas não deixa roteiros ou guias de como desenvolvê-lo, nem pistas de como reduzir os custos para construir jogos altamente acessíveis.

É nesse contexto que se insere este trabalho, propondo uma alternativa à maneira de pensar o desenvolvimento e a arquitetura de um jogo inclusivo. Busca-se superar as dificuldades de desenvolvimento de interfaces acessíveis ao desacoplar a interface da estrutura lógico-funcional do jogo, tornando possível que um mesmo jogo seja jogado por meio de interfaces diferentes e adaptadas às necessidades de perfis de diferentes usuários. Dessa maneira,

buscou-se construir um paradigma de desenvolvimento de jogos que permita uma nova abordagem e que tem como objetivo principal viabilizar jogos que integrem pessoas com as mais diferentes deficiências junto com o jogador que consegue utilizar as interfaces convencionais.

Para isso, buscamos desenvolver um jogo de perguntas e respostas introduzindo uma nova maneira de organizar o projeto. Rollings e Morris (2009) descreve uma possível arquitetura de um jogo desde o *hardware* e suas abstrações diretas até o nível de interação com o jogador. Um jogo pode, portanto, ser dividido em um conjunto de subsistemas e componentes interagindo entre si. Com isso em mente, esse trabalho sugere a criação de um novo subsistema/componente que possa se conectar com múltiplas interfaces diferentes. Neste sentido, foi desenvolvido um núcleo lógico funcional que contém toda a lógica principal do jogo e gerencia o seu estado ao longo de uma partida. O desenvolvimento desta arquitetura é apresentado no próximo capítulo.

3 DESENVOLVIMENTO

Neste capitulo, é descrito o desenvolvimento do núcleo lógico funcional proposto para a implementação de um jogo de perguntas e respostas *online*, *multiplayer* e acessível. Inicia-se pela discussão dos aspectos sobre a construção conjunta da arquitetura e da interface para pessoas com deficiência visual. Em seguida, é descrita a concepção do jogo de perguntas e respostas, bem como os princípios que foram levados em consideração para este feito. Após este preâmbulo, são descritos os requisitos funcionais principais e apresentada a arquitetura geral do núcleo de jogo e suas tecnologias. Assim, serão explicados mais detalhadamente os modelos principais do núcleo, seus *endpoints* e como funcionam em termos gerais as conexões bidirecionais entre cliente e servidor durante uma partida. Por fim, discute-se o impacto da solução acessível e as melhorias implantadas a partir do resultado dos testes e experimentações realizados.

3.1 Repositório

Para realizar o controle de versão e hospedagem dos códigos produzidos, a plataforma GitHub foi usada devido sua gratuidade e facilidade de uso. Assim, o código do núcleo lógico funcional de jogo poderá ser acessado e consultado em https://github.com/athekes/nucleo-logico-funcional.

3.2 Construção conjunta

O trabalho atual compõe um projeto maior de um jogo *mobile* de perguntas e respostas acessível para pessoas portadoras de deficiência visual, que engloba tanto o núcleo, que incorpora a lógica de jogo, como as interfaces acessíveis e convencionais do jogo. Como produto final do projeto, foi desenvolvido um aplicativo para celulares com o sistema operacional *Android*, onde jogadores com ou sem deficiência visual podem se integrar em uma mesma partida através de diferentes interfaces.

Nesse contexto, a construção das funcionalidades, definição de requisitos funcionais e especificidades do uso do núcleo de jogo foram construídos em conjunto com o desenvolvimento do módulo de interface acessível, objeto de outro Trabalho de Conclusão de Curso. O trabalho realizado concomitantemente e cooperativamente permitiu o desenvolvimento de um aplicativo que fornece uma interface *touch screen* e uma interface acessível completamente por voz para os

jogadores finais. Sendo assim, mediante a construção conjunta da aplicação final, foi possível notar quais são as necessidades específicas de cada funcionalidade do núcleo para que ele pudesse ser completamente viável em múltiplas interfaces.

3.3 Concepção de um Jogo de Perguntas e Respostas

Tendo em vista o objetivo do presente trabalho em fornecer uma nova forma de desenvolver jogos que viabilize a criação de interfaces acessíveis, foi definido que o jogo a ser construído seria um jogo trívia *online* de perguntas e respostas. Tal escolha se deu devido ao bom potencial em exemplificar e servir como ensaio para esse novo paradigma de desenvolvimento ao demostrar a capacidade de criar um jogo *online* e *multiplayer* completamente inclusivo em tempo hábil para esse projeto.

Os jogadores poderão criar uma partida e chamar amigos para participar. Uma vez dentro da partida, ao comando de iniciar dado pelo jogador que criou a partida, uma sequência de 5 perguntas com múltiplas alternativas serão apresentadas uma a uma e simultaneamente para todos os jogadores conectados na partida. Após todos os jogadores responderem a pergunta atual, automaticamente, o jogo mostrará uma nova pergunta e suas alternativas para que os jogadores possam responder. Após as 5 perguntas serem respondidas por todos os jogadores será então mostrado, de forma simultânea, o *ranking* de jogadores que mais acertaram perguntas.

É importante notar que o núcleo lógico funcional, ao menos na versão contida nesse trabalho, não possui cronômetros, limites de tempo, ou critérios de desempates. Esse *design* e modelo de jogo foi escolhido e desenvolvido levando em consideração a simplicidade, facilidade de compreensão, jogabilidade e universalidade. Assim, caso um jogador necessite de um tempo a mais para ler as perguntas ele terá tal aspecto do jogo a sua disposição e não terá que se preocupar em critérios de ranqueamento que são impactados por suas deficiências, assim ficando em desvantagem em relação outros jogadores.

O *design* final foi selecionado por oferecer uma boa oportunidade de implementação de interfaces acessíveis e de gerar situações e regras justas ao considerar um universo maior de jogadores. Ao levar em consideração diversas deficiências, torna-se evidente como jogadores distintos poderão se beneficiar de interfaces mais adaptadas a suas necessidades e, ao mesmo tempo, serem competitivos de maneira justa entre si.

3.4 Requisitos

Foram definidos os seguintes Requisitos Funcionais (RF) para o desenvolvimento de um Produto Mínimo Viável (RIES, 2017) da arquitetura do núcleo de jogo.

RF1 - Criar usuário com o nome. O cliente do núcleo, seja qual for, deve ter a possibilidade de criar um usuário ao informar apenas um nome ao servidor. Além disso, o núcleo irá manter esse usuário relacionado a atual conexão desse cliente.

RF2 - Criar partida. Um cliente do núcleo poderá criar uma partida quando quiser após ter criado um usuário e estar conectado.

RF3 - Entrar em uma partida. Um cliente do núcleo poderá entrar em uma partida ao informar o nome da partida ao servidor. Dessa forma, até 4 usuários diferentes podem se conectar a uma mesma partida.

RF4 - Iniciar a partida. Dentre todos os usuários conectados, apenas o dono (aquele que criou a partida) poderá iniciar a sequencia de perguntas enviadas pelo núcleo de jogo.

RF5 - Responder perguntas simultaneamente. Uma vez que uma partida esteja iniciada, todos os clientes receberão a mesma pergunta com múltiplas alternativas e poderão escolher uma alternativa. Após todos responderem, o núcleo de jogo enviará mais uma pergunta.

RF6 - Enviar atualizações sobre a partida. Na primeira vez que um usuário entrar em uma partida e toda vez que o estado da partida mudar, o núcleo de jogo deverá comunicar todos os clientes conectados.

RF7 - Enviar um relatório com o placar ao final da partida. Ao final das perguntas o núcleo de jogo deverá enviar a todos os clientes conectados na partida um breve relatório em que os jogadores são ranqueados pelo numero de perguntas acertadas.

3.5 Arquitetura geral

Com a finalidade de possibilitar que os jogadores façam o uso de diferentes interfaces para o mesmo jogo e que, ainda sim, esses jogadores possam ser integrados numa mesma partida de forma competitiva e acessível, foi sugerida a adição de um novo componente na arquitetura geral do jogo.

Diferente da possível arquitetura geral de jogo descrita por Rollings e Morris (2009), a estratégia escolhida tem como fundamento a introdução de um núcleo de jogo como um componente centralizador na arquitetura, a fim de abstrair os componentes lógicos-funcionais

que são indiferentes a interface acoplada ao jogo. Por meio desse núcleo, as interfaces poderão se conectar e repassar os comandos do jogador. A arquitetura do núcleo, então, estará associada a bancos de dados com os registros dos jogadores, suas respostas e todas as informações que necessitam ser persistidas. Além disso, o núcleo conterá toda a lógica e regras que regem o andamento do jogo.

Com isso, foi formado um padrão de comunicação aberto a qualquer cliente que deseja se comunicar com núcleo de jogo. Por meio dessa "linguagem comum", definida pela comunicação que as interfaces farão com o núcleo, uma mesma partida poderá ser jogada por jogadores que interagem com seus dispositivos por meio de diferentes interfaces mas que se comunicam com a mesmo núcleo lógico funcional de jogo.

3.6 Infraestrutura e tecnologias do núcleo de jogo

A aplicação principal, a qual fornece a conexão ao núcleo para as possíveis interfaces, é estruturada em torno de um componente principal que junta as tecnologias necessárias para conectar os múltiplos clientes de forma simultânea e persistir as informações necessárias. Assim, podemos definir a infraestrutura do projeto da seguinte maneira:

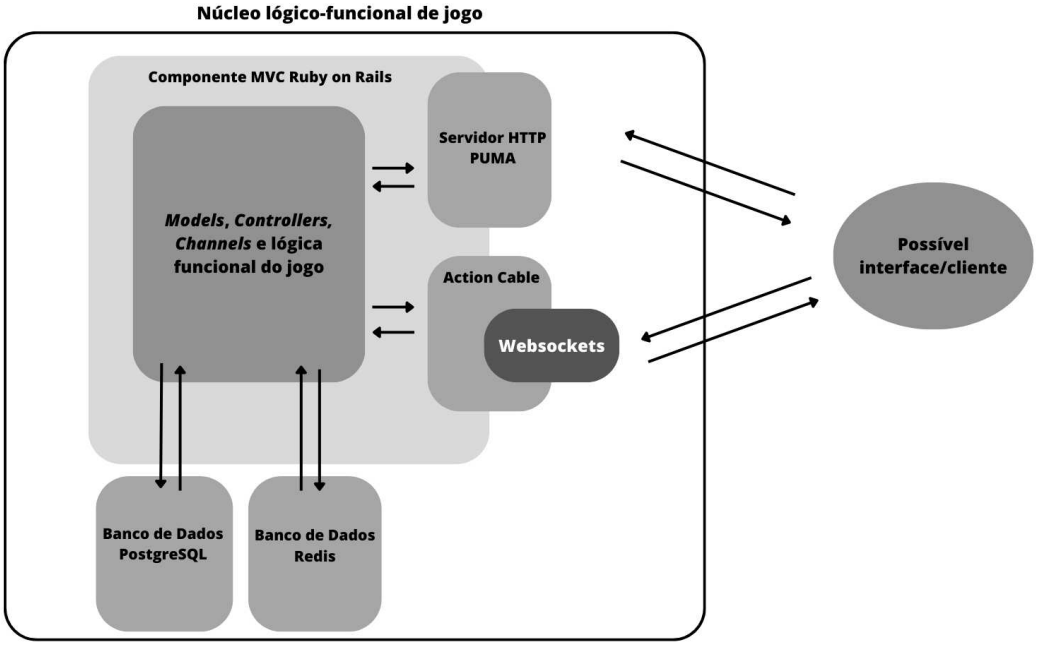
O componente principal do núcleo lógico funcional foi construído usando o *Framework* de desenvolvimento web *Ruby on Rails* (Ruby on Rails, 2004) que implementa o padrão de projeto *Model, View, Controller* (MVC). Com ele, foi possível estabelecer um servidor web de *Hypertext Transfer Protocol* ou Protocolo de Transferência de Hipertexto (HTTP) Puma (Puma, 2011) e integrá-lo a lógica de jogo através de um conjunto de *endpoits* HTTP dinâmicos. As conexões bidirecionais, estabelecidas durante as partidas, foram construídas através do *ActionCable* (Action Cable, 2016). Que abstrai a lógica das conexões *WebSockets* que serão definidas na sessão 3.7.3. Além disso, conectado ao componente principal, teremos um banco de dados relacional *PostgreSQL* (Postgress, 1996) para lidar com a persistência, e um banco de dados não relacional *Redis* (Redis, 2009) para suportar com as conexões bidirecionais do *ActionCable*.

O projeto foi inicialmente hospedado na plataforma Fly (Fly, 2008) que é um serviço Platform as a Service ou Plataforma como Serviço (PAAS) e posteriormente transferido para a plataforma *Heroku* (Heroku, 2009) com a finalidade de obter mais desempenho geral no núcleo de jogo. Com a ajuda dessas plataformas, foi possível simplificar todo o processo de instalação da aplicação em uma maquina remota, bem como a configuração devida de todas as tecnologias

para funcionarem em modo de produção. Além disso, espera-se simplificar a manutenção do projeto e futuras implementações.

Dessa forma, foi possível construir uma arquitetura básica em que seja possível responder requisições de criação de usuários nas salas virtuais onde as partidas irão ocorrer, bem como gerenciar múltiplas conexões bidirecionais e simultâneas via *WebSockets* como mostra a Figura 2.

Figura 2 – Arquitetura do Núcleo de Jogo, com seus serviços e tecnologias.



Fonte: Produzido pelo autor.

Com base na presente arquitetura, antecipa-se a possibilidade de criar partidas até que os recursos da máquina na qual o componente MVC Rails está hospedado, assim como os bancos de dados PostgreSQL e Redis, sejam totalmente utilizados. Esses são os pontos principais de limitação em termos de escalabilidade da arquitetura. Ou seja, à medida que o uso do núcleo se intensifica, é previsto o esgotamento dos recursos de memória e capacidade de processamento da máquina na qual a aplicação está sendo executada, o que poderá resultar em erros e inconsistências no núcleo. Com o intuito de expandir as capacidades do núcleo de jogo, é considerada como prospecção futura, a implementação de réplicas do componente principal Rails que possam distribuir a demanda de forma mais eficiente.

3.7 Desenvolvimento do núcleo de jogo

Após definida a arquitetura e as tecnologias que seriam usadas no núcleo de jogo, deu-se início ao desenvolvimento e codificação dos modelos, *endpoints* e lógica de comunicação bidirecional.

3.7.1 Modelos

Afim de alcançar todos os requisitos funcionais descritos na sessão 3.4, com o auxilio do *framework* de desenvolvimento *web Ruby on Rails*, um conjunto de modelos foi desenvolvido. Tais modelos representam os registros e tabelas salvas no banco de dados relacional juntamente com entidades e classes que dão suporte à programação.

Nos modelos relacionados as regras de negócio foram:

- ***Question***: É uma possível pergunta que pode ser realizada durante a partida. Ela possui um corpo com o comando da pergunta e está associada a 4 alternativas;
- ***Alternative*** : É uma das 4 alternativas de uma pergunta, que está sempre relacionada a uma pergunta. Ela também possui um corpo com a descrição da alternativa e um campo que define se ela será um alternativa correta ou não;
- ***Room*** : É a sala onde uma partida ocorre. Ela possui um código que os usuários poderão usar para se conectar a ela. Uma sala possui até 3 possíveis estados, sendo eles: "em espera", "realizando perguntas"e "finalizando". A sala sempre possuirá um proprietário (*User*) e também um campo informando qual a pergunta que está sendo realizada atualmente;
- ***User*** : É o usuário que, de fato, jogará o jogo. Ele possui um nome e pode ou não ser dono de uma sala;
- ***Answer*** : É a resposta apresentada por um usuário para uma pergunta. Ela se relaciona: à sala onde foi feita, à pergunta respondida, à alternativa escolhida e ao usuário que respondeu.

3.7.2 Endpoints

Para que um usuário pudesse interagir com tais modelos, afim de criar e participar de partidas, os seguintes *endpoits* foram criados.

- ***POST /users*:** Criar um usuário. Ao realizar esta operação, o servidor enviará um "header

Set-Cookie"com o UUID do usuário criado para que o cliente possa informar esse valor no *cookie* para todos os *requests* HTTP e *WebSockets* subsequentes. Dessa forma, o núcleo poderá distinguir qual usuário realizou qual ação.

- ***POST /rooms*:** Criar uma sala. Deverá ser passado no *cookie* o UUID informado no *header Set-Cookie* do *endpoins POST users*. Assim, o servidor associará à sala o jogador correto como criador. Após a criação da sala, o servidor irá retornar o UUID da sala criada ao cliente.
- ***GET /find_room:*** Achar uma sala. O cliente deve informar um parâmetro "*code*", que contém a palavra pela qual a sala é identificada. Caso o servidor encontre uma sala existente com o nome que foi pesquisado, ele irá retorna o *Universally Unique Identifier* ou Identificador Único Universal (UUID) da sala ao cliente. Nesse *endpoint*, também deverá ser passado o UUID do usuário no cookie para fins de autentificação.

3.7.3 Comunicação bidirecional via WebSockets

Websocket é uma tecnologia que permite uma comunicação bidirecional entre cliente e servidor. Ela funciona através do protocolo TCP, usando apenas uma conexão para comunicar as duas pontas.

Ele é um protocolo desenvolvido com o intuito de substituir técnicas bidirecionais de comunicação anteriores que se baseavam no HTTP como camada de transporte para conseguir se beneficiar da infraestrutura pré-desenvolvida como os *proxies*, filtros e autentificação. Contudo, pelo fato do HTTP não ter sido originalmente desenvolvido com o intuito de suportar uma comunicação bidirecional, as soluções anteriores não eram ideais (RFC 6455, 2011).

Nesse contexto, o protocolo *Websocket* surge com o objetivo de resolver os objetivos das técnicas de comunicação bidirecional anteriores que funcionavam com base no HTTP. Portanto, assim como o HTTP, o *Websocket* funciona através das portas 80 e 433 e consegue aproveitar da infraestrutura pré-existentes do HTTP como os *proxies* HTTP (RFC 6455, 2011).

O protocolo se divide em duas partes principais: o *Handshake* e a transferência de dados. O *Handshake* acontece por meio de uma requisição de HTTP de *upgrade* vinda do cliente como a seguir:

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
```

```
Connection: Upgrade
Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
```

Assim, é esperada uma resposta do servidor como a seguinte:

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo=
Sec-WebSocket-Protocol: chat
```

Após o cliente e o servidor terem realizado o *handshake*, os componentes podem começar a trocar mensagens de forma independente um do outro através do protocolo *Websocket* (RFC 6455, 2011).

3.7.3.1 RoomsChannel

Portanto, para lidar com as requisições bidirecionais necessárias ao funcionamento do jogo simultaneamente para todos os usuários, foi usado o Protocolo *Websocket* com adoção do *ActionCable*, um módulo do *framework Ruby on Rails* que dá suporte ao uso do *Websocket*.

O *ActionCable* permite a criação de uma abstração chamada *channel*, que funciona de forma similar a um *controller* dentro de uma arquitetura MVC, porém especializada no recebimento de mensagens via o protocolo *WebSockets*. Nesse sentido, toda vez que uma mensagem via *WebSocket* é enviado ao servidor, ele busca dentro da mensagem parâmetros que possibilitem seu roteamento a um *channel* especifico afim de realizar as trativas necessárias.

Assim, o *channel RoomsChannel* foi criado para abstrair a lógica de conexão e comunicação do cliente com as salas de jogo e se encarregar de realizar o processamento das requisições. Assim, sempre que um cliente quiser enviar uma mensagem a sua sala, ele deverá se comunicar usando parâmetros que podem ser roteados ao *channel*.

O *RoomsChannel* foi configurado para que o cliente possa assinar uma *stream* de informações identificada pelo UUID da sala à qual ele está se conectando. Logo, além de uma conexão *WebSocket* simples, ao se conectarem por meio do *RoomsChannel* os clientes se

tornam assinantes em um padrão de comunicação no qual o núcleo de jogo atua como publicador e os assinantes são todos os clientes que se inscreveram na *stream* de uma determinada sala. Nesse sentido, várias *streams* de informação poderão ser estabelecidas com diferentes grupos de clientes, permitindo a comunicação direcionada e simultânea entre todos os participantes de uma mesma sala, sem interferência na comunicação de salas paralelas.

A fim de fornecer serviços e estabelecer uma comunicação efetiva, semelhantemente a um *controller*, dentro do *channel RoomsChannel*, foram criado um conjunto de *actions*. Com elas, os clientes do núcleo podem, então, enviar uma mensagem *WebSocket* requisitando algum recurso do servidor e estabelecer uma comunicação.

3.7.3.2 Action Subscribed

Permite que um cliente estabeleça uma conexão *WebSocket* com o servidor e se inscreva na *stream* de uma sala específica. A partir desse ponto, após uma conexão bem sucedida, o servidor começa a enviar mensagens simultaneamente a todos os clientes que requisitaram essa *action* passando como argumento a mesma sala. É nesse ponto onde será estabelecida uma conexão *WebSocket* bidirecional, pela qual o cliente receberá atualizações da partida como a pergunta atual, suas alternativas e também poderá enviar sua respostas ao servidor no momento devido.

Contudo, previamente ao envio da mensagem *subscribed* por meio do protocolo *WebSocket*, o cliente deverá realizar a requisição HTTP de *Upgrade* estabelecendo uma conexão *WebSocket* simples com o núcleo. Com isso, por meio do UUID do usuário presente nos *cookies* da requisição, a conexão é associada a um usuário específico. Essa associação permite a identificação do usuário que está solicitando um recurso nas requisições subsequentes. Dessa forma, torna-se possível rastrear e determinar qual usuário está envolvido em cada requisição de maneira precisa.

Uma vez devidamente estabelecida a conexão *WebSocket*, o cliente poderá assinar a *stream* de uma determinada sala ao se comunicarem com a *action subscribed*. Os clientes devem passar como parâmetros o tipo de comando como "*subscribe*", o *channel* e o UUID da sala em que o cliente deseja se conectar no seguinte formato:

```
command: "subscribe",
identifier: {
  channel: "RoomsChannel",
```

```
    room_id: "efa85f58-4fdb-4d30-9836-f5a5efe5c4f9"
}
```

3.7.3.3 Action init_game

Permite que o dono de uma sala inicie a partida. Para usar essa *action*, os clientes devem, utilizando-se de uma conexão criada previamente, fornecer o tipo de comando como "*message*", identificar a sala, o canal e informar qual *action* ele esta requisitando no seguinte formato:

```
command: "message",
identifier: {
  channel: "RoomsChannel",
  room_id: "efa85f58-4fdb-4d30-9836-f5a5efe5c4f9"
},
data: {
  action: "init_game"
}
```

3.7.3.4 Action show_current_scoreboard

Requisita ao núcleo que envie o placar da partida para todos os jogadores conectados em uma sala. Utilizando-se de uma conexão criada previamente, os clientes também devem fornecer o tipo de comando como "*message*", identificar a sala e o canal, bem como informar qual *action* ele esta requisitando no seguinte formato:

```
command: "message",
identifier: {
  channel: "RoomsChannel",
  room_id: "efa85f58-4fdb-4d30-9836-f5a5efe5c4f9"
},
data: {
  action: "show_current_scoreboard"
```

```
}
```

3.7.3.5 Action register_answer

Registra uma resposta do atual jogador. Como requisito de uso, os clientes devem utilizar-se de uma conexão criada previamente e fornecer o tipo de comando como "*message*", identificar a sala e o canal, informar qual *action* ele esta requisitando e a alternativa selecionada no seguinte formato:

```
command: "message",
identifier: {
  channel: "RoomsChannel",
  room_id: "efa85f58-4fdb-4d30-9836-f5a5efe5c4f9"
},
data: {
  action: "register_answer",
  alternative_position: "a"
}
```

3.7.4 Mecanismo de jogo

O mecanismo principal de jogo tem como ponto de partida as requisições HTTP do projeto, onde o jogadores poderão criar seu usuário e criar ou achar a sala de jogo. Após uma sala criada ou encontrada, o jogador poderá estabelecer uma conexão *WebSocket* com a sala desejada através do *RoomsChannel*. A partir desse ponto, o cliente irá se comunicar com o servidor através das *actions WebSockets* descritas na secção 3.7.3.

Quando um usuário faz a chamada de alguma dessas *actions*, o servidor irá identificar, no corpo da mensagem, a qual sala ela pertence. Assim, a partida na qual ele está atualmente conectado é instanciada e então, os parâmetros da chamada são repassados para a lógica de jogo. Nesse ponto, o núcleo busca no banco de dados todas as informações contidas na partida, como os usuários conectados, as perguntas da partida e seu estado atual. Uma vez que essas informações estejam reunidas, a ação requisitada é executada, processada, os estados e variáveis

da partida são atualizados devidamente e uma resposta é fornecida a todos os clientes conetados a mesma sala.

Assim, ao invés de instanciar um processo para cada partida em andamento, a fim de otimizar recursos e tornar o projeto mais fácil de configurar e manter, buscou-se uma arquitetura na qual cada requisição, via *WebSocket*, deve conter a identificação de qual sala ela pertence. Uma vez que uma mensagem chega ao servidor, as informações da partida são carregadas e uma resposta é processada.

3.7.5 Nomes de salas acessíveis

Visando maior acessibilidade para um jogador entrar em alguma partida, principalmente jogadores com alguma deficiência visual, foi construída uma solução que sempre busca um nome simples para cada sala. Para entrar em uma sala, o cliente deve fazer uma requisição passando o nome da sala para o servidor. Se houver uma sala com o nome fornecido, ele será conectada a ela.

Nesse contexto, a fim de facilitar a entrada dos jogadores, foi construída uma lógica de geração de nomes simplificada. A lógica se define em nomear as salas com nomes de animais que facilitem a compreensão pelo jogador e reduza eventuais erros de interpretação (de áudio, por exemplo) no momento de enviar o nome da sala ao servidor. Assim, o nome de uma sala pode ser “tatu”, “zebra” ou então “aranha”.

3.7.6 Melhoria incremental

O desenvolvimento do núcleo lógico funcional se prolongou após o processo de desenvolvimento inicial, e estendendo-se ao período de testes do projeto. Dessa forma, ao confrontar o que já havia sido desenvolvido com as experiências e *feedbacks* de voluntários, foi possível perceber diversas melhorias e correções necessárias para garantir o bom desempenho do projeto.

Com isso, após a execução de sessões de teste com voluntários, correções específicas para resolver os problemas percebidos foram implementadas. O desenvolvimento, em boa parte, se deu de forma incremental, seguindo o ciclo de implementação, teste, coleta de *feedback* e correções. No capítulo de resultados, é detalhadamente registrado como cada teste influenciou no resultado final do núcleo de jogo desenvolvido.

4 RESULTADOS

Uma vez implementado, o núcleo de jogo passou a ser testado em cenários que buscavam compreender cada vez melhor seus aspectos, falhas e insuficiências quanto ao objetivo final. Nesse contexto, os testes executados podem ser divididos em testes de desempenho e testes com usuários. São descritos neste capítulo os procedimentos realizados para cada teste, seus cenários, resultados e como, por meio deles, foi possível melhorar de forma incremental a qualidade do projeto final.

4.1 Testes de desempenho

A fim de coletar dados sobre o desempenho da ferramenta e estabilidade da arquitetura do núcleo de jogo, uma bateria de testes foi realizada, tanto em ambiente de desenvolvimento local como em ambiente de produção.

Para realização desses testes, a ferramenta *Apache JMeter* (Apache JMeter, 1998) foi utilizada. O *Apache JMeter* é uma aplicação de código aberto projetada para testes de carga, testes funcionais e medição de performance. A aplicação permite a criação de um *script* de teste que será replicado simultaneamente através de diversas *threads* configuradas durante a programação dos testes. Dessa forma, foi possível simular o uso simultâneo de vários usuários ao sistema de forma concorrente e então observar o comportamento do núcleo nesse cenário.

Por padrão, o *Apache JMeter* não tem suporte para teste de conexões e requisições via *WebSocket*. Portando, foi necessária a instalação do *plugin JMeter WebSocket Samplers* (DOORNBOSCH, 2016). Com ele, foi possível estabelecer conexões *WebSockets*, enviar mensagens e recebê-las por dentro do *Apache JMeter*.

O principal objetivo dos testes de desempenho realizados foi analisar como os tempos médios de resposta se alteram frente a uma demanda concorrente considerável especulativa, descrita em diferentes cenários. Observa-se que o volume e as características dos testes precisarão ser precisamente dimensionados para o ambiente final, caso o produto seja implantado em um ambiente específico para um público aberto ou privativo. Assim, testes mais extensos fornecerão uma análise mais completa dos recursos consumidos pela aplicação, bem como seu comportamento diante de uso recorrente e prolongado.

4.1.1 Configuração da plataforma de teste

Com a finalidade de construir um cenário de teste que explore as reais funcionalidades do núcleo de jogo o seguinte *script* de teste foi definido:

1. Um usuário é criado;
2. Uma sala é criada para o usuário criado no passo 1 como seu proprietário;
3. Uma conexão via *WebSocket* é estabelecida com a sala criada;
4. A primeira mensagem enviada pela conexão;
5. A segunda mensagem enviada pela conexão;
6. A partida é iniciada;
7. 5 repetições:
    a) A pergunta é ouvida;
    b) A pergunta é respondida com a primeira alternativa;
8. É requisitado o placar da partida;
9. A conexão é finalizada após 10 segundos.

Os planos de teste no *Apache JMeter* possuem 2 parâmetros principais, os quais podem ser ajustados a fim de criar novos cenários para um mesmo *script* de teste:

- **Threads:** Especifica quantas execuções serão geradas para simular usuários se conectando e fazendo requisições à aplicação.
- **Ramp Up:** Especifica o tempo que o teste levará para alcançar o numero de *Threads* definido. Assim, é possível especificar, por exemplo, que desejamos 10 execuções (*Threads*) em 5 segundos (*Ramp up*), o que indica que, em 1 segundo, o teste irá iniciar uma execução 2 vezes de forma simultânea e poderá ter mais conexões simultâneas, caso a duração do plano de teste seja maior que 1 segundo, sendo 10 o número máximo de conexões.

4.1.2 Cenários de testes de desempenho

Os cenários foram executados na arquitetura do núcleo definida na Figura 2 e que esteve hospedada na plataforma Heroku. Como o serviço Heroku é um PAAS, as informações a respeito do poder computacional disponível e detalhamentos sobre a arquitetura das máquinas utilizadas não são acessíveis. Dessa forma, ressalta-se que os testes apresentados buscam apenas fornecer dados e parâmetros sobre o comportamento do núcleo desenvolvido frente a uma demanda concorrente, sendo necessário novos cenários dimensionados para um possível

ambiente final.

Os cenários estão resumidos na Tabela 1 e são apresentados nas próximas subseções, juntamente com os resultados obtidos com o uso do *Apache JMeter*. As tabelas de resultados para cada cenário contêm média, mediana, 90% *Line*, 95% *Line* e 99% *Line*, tempos máximos e mínimos de resposta para cada fase do plano de teste, ordenados pela maior média para cada cenário. Os valores são apresentados em milissegundos. As medidas 90% *Line*, 95% *Line* e 99% *Line* indicam o tempo de resposta médio necessário para que um determinado percentual de requisições seja concluído. Por exemplo, o valor apresentado para 90% *Line* (ou 90% Percentil), indica o tempo de resposta médio necessário para que 90% das requisições sejam concluídas. Além disso, as tabelas incluem uma coluna denominada *samples*, que indica o número de vezes que um mesmo *request* ou *action* foi requisitado durante a execução do cenário.

Assim, construiu-se 4 planos de testes variando as seguintes estratégias.

Tipo de cenário	Threads	Ramp up
Cenário base	1	1
Cenário 1	100	10
Cenário 2	200	5
Cenário 3	500	60

Tabela 1 – Configurações dos cenários de teste aplicados

4.1.2.1 Cenário base (1 Thread e 1 Ramp Up)

Esse teste serve como base para avaliação dos tempos de resposta do núcleo, quando não há concorrência. Assim, apenas uma execução do plano descrito foi realizada e os dados coletados. A Figura 3 reúne os dados que mostram que o núcleo de jogo manteve uma média de resposta de 100 milissegundos, considerando os *requests* HTTP e via *WebSockets*.

Nos resultados desse e dos cenários subsequentes, deve-se ignorar a taxa de erro em "*close websocket connection*". Ela indica apenas que não houve um retorno imediato ao *JMeter* após dar-se início ao processo de encerramento de uma conexão *WebSocket*. Isso acontece devido ao *delay* de 10 segundos no fechamento da conexão configurado no *script* de teste.

4.1.2.2 Cenário 1 (100 Threads e 10 Ramp Up)

Nesse teste, 100 *threads* diferentes são executadas em até 10 segundos. Assim, foi possível inspecionar o comportamento do núcleo de jogo quando exposto à uma carga produzida

Figura 3 – Resultados do teste de desempenho - Cenário base

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %
Connect on room	1	461	461	461	461	461	461	461	0.000%
Create user	1	449	449	449	449	449	449	449	0.000%
Show scoreboard	1	113	113	113	113	113	113	113	0.000%
Create room with user	1	112	112	112	112	112	112	112	0.000%
Init game	1	102	102	102	102	102	102	102	0.000%
Answer question	5	101	101	103	103	103	100	103	0.000%
Listem confirm subscription	1	99	99	99	99	99	99	99	0.000%
Listem question	5	10	13	13	13	13	1	13	0.000%
Listem end game	1	10	10	10	10	10	10	10	0.000%
Listem second message	1	1	1	1	1	1	1	1	0.000%
Close websocket connection	1	0	0	0	0	0	0	0	100.000%
TOTAL	19	100	100	113	449	461	0	461	5.263%

Fonte: Criado pelo autor.

por um conjunto de *request* simultâneos. Com esta configuração, cerca de 10 execuções do plano são requisitadas de forma simultânea, garantindo que, no mínimo, 10 salas são criadas e "jogadas"ao mesmo tempo. O teste permite investigar como o núcleo se comporta quando múltiplas requisições de diferentes recursos são feitas ao mesmo tempo. Os resultados desse cenário são apresentados na Figura 4.

Figura 4 – Resultados do teste de desempenho - Cenário 1

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %
Connect on room	100	483	475	550	575	598	398	598	0.000%
Create user	100	365	361	390	395	460	330	464	0.000%
Show scoreboard	100	219	224	270	288	314	112	402	0.000%
Init game	100	212	218	268	281	298	90	336	0.000%
Answer question	500	191	189	262	283	301	89	323	0.000%
Create room with user	100	133	131	145	152	215	99	458	0.000%
Listem confirm subscription	100	96	81	181	215	225	10	232	0.000%
Listem question	500	22	13	61	78	115	0	160	0.000%
Listem end game	100	21	15	48	65	107	0	110	0.000%
Listem second message	100	15	4	49	64	84	0	109	0.000%
Close websocket connection	100	0	0	0	0	0	0	0	100.000%
TOTAL	1900	137	116	338	405	533	0	598	5.263%

Fonte: Criado pelo autor.

4.1.2.3 Cenário 2 (200 Threads e 5 Rump Up)

Nesse teste, pode-se observar o comportamento do núcleo com uma carga bem mais elevada de requisições simultâneas. Nessa configuração, 40 execuções do plano de teste são iniciadas simultaneamente a cada segundo. Com isso, foi possível observar um considerável aumento nos tempos de resposta de cada requisição e o surgimento de alguns erros, que podem ser observados nas requisições que pousem um valor máximo acima de 6000 milissegundos, que foi o valor configurado para *timeout* ao longo de todas as requisições do plano. A Figura 5 ressalta em vermelho as taxas de erros relevantes obtidas.

Figura 5 – Resultados do teste de desempenho - Cenário 2

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %
Answer question	1000	2117	2217	2576	2603	2660	481	6004	1.000%
Connect on room	200	1806	1758	2867	3001	3217	437	3269	0.000%
Init game	200	1801	1922	2074	2369	2550	216	6004	1.000%
Show scoreboard	200	1463	1276	2386	2526	2613	100	2655	0.000%
Create user	200	980	944	1481	1679	1759	329	1797	0.000%
Create room with user	200	899	1048	1466	1493	1528	108	1649	0.000%
Listem confirm subscription	200	760	780	1148	1289	1460	98	1534	0.000%
Listem second message	200	758	776	1100	1282	1479	1	1505	0.000%
Listem question	1000	88	12	81	121	270	0	6004	1.000%
Listem end game	200	88	13	93	112	199	0	6003	1.000%
Close websocket connection	200	0	0	0	0	0	0	0	100.000%
TOTAL	3800	1031	877	2474	2567	2939	0	6004	5.895%

Fonte: Criado pelo autor.

4.1.2.4 Cenário 3 (500 Threads e 60 Ramp Up)

Nesse cenário, foi possível observar o comportamento do núcleo quando muitas requisições são realizadas em um período relativamente prolongado em relação ao demais cenários. A Figura 6 ilustra que, apesar da não obtenção de tempos de respostas elevados, foi possível observar um caso de erro na leitura de algumas respostas.

Figura 6 – Resultados do teste de desempenho - Cenário 3

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %
Connect on room	500	595	528	741	910	2132	397	2648	0.000%
Show scoreboard	500	374	308	673	886	955	105	2536	0.000%
Create user	500	372	377	403	414	439	325	585	0.000%
Init game	500	349	294	643	731	966	90	2448	0.000%
Answer question	2500	344	282	653	768	943	87	2535	0.000%
Listem confirm subscription	500	153	135	265	306	556	11	1430	0.000%
Create room with user	500	141	140	163	170	197	101	319	0.000%
Listem second message	500	116	47	264	316	570	0	6001	0.200%
Listem end game	500	30	17	77	103	170	0	234	0.000%
Listem question	2500	27	15	68	92	164	0	1359	0.000%
Close websocket connection	500	0	0	0	0	0	0	0	100.000%
TOTAL	9500	210	133	537	654	926	0	6001	5.274%

Fonte: Criado pelo autor.

4.2 Testes com usuários

Com o objetivo de coletar informações sobre a usabilidade da aplicação que se integra com o núcleo lógico desenvolvido em um contexto real, foram planejados e realizados uma série de testes com voluntários. Ao todo, 3 cenários de testes foram realizados e geraram importantes *feedbacks*, permitindo não só a melhoria de aspectos da interface, mas também das funções exportadas pelo núcleo.

Em cada sessão de teste, os voluntários foram convidados a participar de diversas partidas, enquanto suas experiências, dificuldades e *feedbacks* eram anotados. Durante cada sessão, todos os momentos do uso do aplicativo foram acompanhados a fim de compreender as

lacunas existentes no projeto.

4.2.1 Grupos de teste

A fim de melhorar a qualidade e a universalidade dos resultados, buscou-se formar diferentes grupos de testes. Nas subseções a seguir, cada grupo de testes será definido, bem como suas peculiaridades relevantes na análise dos resultados.

4.2.1.1 Grupo 1: Alunos e recém formados em Engenharia de Computação

O grupo 1 era composto por 2 alunos do curso de Engenharia de Computação na Universidade Federal do Ceará (UFC) e 2 alunos recém formados no mesmo curso. Todos os membros são pessoas que possuem por volta de 21 anos de idade, sem nenhuma deficiência visual e confortáveis com o uso de tecnologias e aplicativos de *smartphones*. Tal conjunto de voluntários era previamente conhecido pelos desenvolvedores e foram convidados com o objetivo de realizar os primeiros testes reais com o aplicativo e compartilhar suas impressões preliminares sobres o projeto.

4.2.1.2 Grupo 2: 14 alunos da Disciplina de Desenvolvimento Web

O grupo 2 foi composto por alunos da turma 2023.1 da Disciplina de Desenvolvimento de Aplicações *Web*, do curso de Engenharia de Computação na UFC. Os membros do grupo são pessoas que possuem por volta de 20 anos, também sem nenhuma deficiência visual e confortáveis com o uso de tecnologias moveis em *smartphones*. Esse grupo não era previamente conhecido pelos desenvolvedores e foram convidados no intuito principal de testarem o comportamento da aplicação quando exposta a uma carga simultânea com várias salas iniciadas e jogadas simultaneamente.

4.2.1.3 Grupo 3: Voluntário portador de deficiência visual

O grupo 3 é constituído por um voluntário com deficiência visual total (0% da visão). A condição do voluntário é desde o nascimento e foi causada por um glaucoma. Contudo, é importante notar que o voluntário escolhido possui ampla habilidade com soluções tecnológicas *mobile*, fazendo o uso diário com o auxílio de leitores de tela e ferramentas de acessibilidade. Os testes realizados com esse voluntário tiveram a finalidade principal de averiguar a real

acessibilidade do projeto e identificar oportunidades de melhoria, a fim de aumentar o potencial competitivo de jogadores portadores de deficiência visual.

4.2.2 Formulário de avaliação de usabilidade

Para coletar informações quantificáveis sobre a experiência dos voluntários, após a execução de cada cenário de teste, um formulário foi apresentado aos participantes. O formulário consiste de um conjunto de perguntas feitas em dois contextos diferentes, que pretendem auxiliar a entender como o ato de jogar usando a interface acessível impacta a experiência e a competitividade dos participante no jogo.

Assim, sobre a experiência dos voluntários jogando pela interface acessível foi questionado:

- De 0 a 5, quão fácil foi interagir com o jogo no geral?
- De 0 a 5, quão fácil foi criar uma partida? (não responda caso não tenha criado uma partida utilizando o modo acessível)
- De 0 a 5, quão fácil foi o processo de responder as perguntas?
- De 0 a 5, o quão fácil foi entender oque as perguntas e alternativas estavam dizendo?
- De 0 a 5, o quão competitivo o jogo lhe permitiu ser em relação aos outros jogadores?
- De 0 a 5, o quão fluida foi sua experiência de jogo durante as perguntas?
- De 0 a 5, o quão fluida foi sua experiência de jogo antes de entrar nas perguntas?

Da mesma forma, sobre a experiência dos voluntários jogando na interface convencional foram realizadas as mesma perguntas, para fins comparativos.

4.2.3 Cenários de Teste

4.2.3.1 Cenário 1 (Grupo 1)

Esse cenário foi realizado de forma remota com todos os voluntários participando através de seus *smartphones* pessoais, a fim de facilitar a participação.

Primeiramente, o instalador do aplicativo foi fornecido aos voluntários. Após a instalação e uma breve apresentação inicial das funções do jogo, os jogadores foram orientados a jogar uma partida de habituação com todos os participantes usando a interface acessível. Para isso, foram guiados para que executassem os seguintes passos:

1. Todos os voluntários criam seu usuário dentro do aplicativo, usando a interface de voz;

2. Um voluntário é escolhido para criar uma partida e compartilhar o código para que os outros participantes entrem na partida;
3. O voluntário escolhido para ser o responsável em dar inicio à partida e os participantes jogam até o final.

Após a partida de habituação ter sido realizada, os jogadores foram orientados a escolher um jogador para jogar na interface acessível e todos os demais jogarem com a interface convencional *touch screen*. Assim, mais partidas foram realizadas enquanto os participantes iam se revezando em usar o modo acessível do projeto ou o modo convencional.

4.2.3.2 Resultados do cenário 1

O primeiro cenário foi dividido em duas sessões diferentes devido a dificuldades encontradas. Diversos problemas no projeto como um todo inviabilizaram a realização completa dos procedimentos na primeira sessão. Alguns problemas encontrados vão além do escopo deste trabalho específico. Assim, serão descritas as dificuldades encontradas que possuem conexão direta ou indireta com o núcleo de jogo.

Portanto, dentre as dificuldades encontradas que estão relacionadas ao núcleo de jogo, ressaltam-se:

– O núcleo não foi capaz de enviar a atualização de quantos usuários estão conectados atualmente na sala quando um novo usuário consegue se conectar;
– Com mais de 3 usuários conectados em uma mesma sala, diversas instabilidades começaram a ocorrer e se tornou impossível iniciar uma partida.

Após a execução da primeira sessão de testes foi realizado o incremento da capacidade do servidor de gerenciar conexões simultâneas ao mudar do serviço PAAS Fly para o Heroku. Isso permitiu um considerável aumento de memoria e capacidade de processamento disponível, tanto na máquina principal onde está hospedada a aplicação Rails, como nas máquinas onde estão hospedados os banco de dados PostgreSQL e Redis.

Após a realização dos testes, o formulário de avaliação foi encaminhado aos participantes e os dados foram coletados, processados e estão dispostos nas Tabelas 2 e 3.

O primeiro teste obteve um mal desempenho frente aos participantes, e isso pode ser identificado ao analisarmos os dados coletados através dos formulários ou nas dificuldades observadas.

4.2.3.3 Cenário 2 (Grupo 2)

Na segunda sessão, as melhorias necessárias apontadas pelos resultados do primeiro cenário já haviam sido implementadas. Assim, a sessão de testes foi executada com o grupo 2 de forma presencial. Os voluntários foram dispostos em grupos de 3 a 4 pessoas. Cada um dos grupos tinha como objetivo realizar e concluir partidas juntos durante a sessão. Esse momento foi capturado por foto e é apresentado na Figura 7.

Figura 7 – Voluntários realizando os testes de usabilidade - Cenário 2

Fonte: Fotografado pelo autor.

Antes da realização dos testes, os voluntários foram orientados a realizar o *download* do aplicativo, instalar em seu respectivos *smartphones* e receberam instruções de que deveriam jogar todas a sessões de teste utilizando dados móveis 3G e 4G. A equipe de desenvolvedores forneceu uma conexão 4G aos voluntários, que não possuíam conexão com a internet no momento.

Com o intuito de inspecionar a clareza e acessibilidade das instruções do próprio aplicativo, os participantes não receberam instruções por parte dos desenvolvedores sobre como o jogo funcionaria ou como as interações com o aplicativo devem ser feitas.

Portanto, apesar de limitadas, apenas algumas orientações foram feitas aos voluntá-

rios afim de aumentar a qualidade dos dados obtidos, sendo elas:

- Realizar uma primeira partida de habituação com todos os participantes do grupo no modo acessível;
- Escolher um participante diferente a cada rodada para jogar via interface acessível;
- Alternarem o iniciador da partida para as rodadas consecutivas de forma que todos os integrantes do grupo possam experimentar jogar e usar as funções de criação e entrada de salas nas duas interfaces disponíveis.

Com isso, 4 grupos começaram e executar suas partidas de forma simultânea.

4.2.3.4 Resultados do cenário 2

Devido às melhorias implementadas após a execução dos primeiro cenário, observou-se uma maior estabilidade da aplicação. Contudo, durante a execução das partidas e formação das salas, desconexões (provenientes da natureza das conexões 3G / 4G) foram frequentes e isso ocasionou travamento de algumas partidas. São duas as hipóteses: (i) como o usuário que foi desconectado não havia respondido à pergunta, o servidor ainda agia como se o mesmo ainda estivesse presente na sala; (ii) quando o líder da sala era desconectado, os usuários ficavam presos em estados nos quais não podiam continuar a partida.

Após todos os grupos concluírem a primeira partida, no entanto, uma melhoria nos servidores foi implementada na tentativa de reduzir as desconexões e instabilidades. Tais mudanças, segundo os participantes, melhoraram a experiência ao reduzir, apesar de não completamente, os casos de desconexão e instabilidades.

Os testes revelaram boa conectividade em relação à simultaneidade de salas e competitividade entre os jogadores, uma vez que foi possível notar genuína competição entre os participantes que estavam habituados às funções do jogo.

Contudo, foi possível perceber que, ao ser jogado através de conexões mais instáveis 3G/4G, o núcleo não consegue lidar bem com tais instabilidades, ocasionando travamento das partidas. Isso se deve principalmente ao modelo de falha de conexões que se revelou insuficiente. Apesar de lidar melhor com as situações nas quais o aplicativo é fechado ou sofre uma falha critica, ele não foi capaz de realizar as tratativas corretas quando a conexão de internet é simplesmente desabilitada por alguma instabilidade na rede. Dessa forma, o núcleo, por alguns momentos, ficou esperando por uma resposta do usuário desconectado sem dar uma forma do usuário se reconectar ao jogo, ocasionando um travamento da partida.

Sendo assim, as melhorias desenvolvidas em decorrência dos resultados e *feedbacks* desse teste foram:

- Elaboração de uma lógica de checagem de conexões. O servidor, em intervalos de 5 segundos, checa se a conexão com o cliente ainda está corretamente estabelecida, enviando uma mensagem *ping* via *WebSockets* e esperando uma resposta *pong*.
- Caso o cliente não consiga responder a tempo, o cliente é tratado da mesma forma que um cliente que fechou o aplicativo. Dessa forma, caso um jogador se desconecte da partida por problemas de conexão, o jogo consegue continuar normalmente depois de alguns segundos.

Após a realização dos testes, o formulário de avaliação foi encaminhado aos participantes e os dados foram coletados, processados e estão dispostos nas Tabelas 2 e 3.

4.2.3.5 Cenário 3(Grupo 3)

O cenário 3 foi realizado de forma presencial, com um voluntário com deficiência visual. O teste foi realizado usando-se de um dos *smartphones* dos desenvolvedores, conectado em uma rede local, uma vez que o celular do voluntário contava com o sistema iOS, para o qual não foi possível realizar a portabilidade do aplicativo devido a procedimentos de avaliação necessários realizados pela Apple. Portanto, nesse cenário, não houve a instalação do aplicativo pelo voluntário.

Ao todo foi proposto ao voluntário jogar 2 partidas. Na primeira partida, o voluntário criou seu usuário, sala, iniciou a partida e jogou até a final, interagindo sozinho com o aplicativo através da interface acessível. Na segunda, o voluntário entrou em uma partida criada por um outro participante que estava jogando remotamente e, então, disputou a partida ate o final.

4.2.3.6 Resultados do cenário 3

Durante a execução da partida, nenhuma falha ou instabilidade foi notada no aplicativo ou no núcleo de jogo.

Assim, após a execução das partidas um formulário simplificado, apenas com as perguntas sobre a interface acessível, foi encaminhado e respondido pelo voluntário. Os resultados podem ser averiguados na Tabela 2.

O voluntário apontou que o aplicativo conseguiu ser acessível e cativante, segundo ele:

"Gostei, jogaria de novo e chamaria meus amigos pra jogar comigo, porque sou competitiva" (Voluntário portador de deficiência visual)

Contudo, foram apresentadas algumas sugestões, como informar a quantidade total de perguntas e a possibilidade de alterar a velocidade da leitura das frases.

4.3 Resultados Compilados

As tabelas a seguir descrevem os resultados obtidos em cada um dos testes realizados através das respostas dos usuários ao formulário de experiência que foi apresentado na sessão 4.2.2. A escala dos resultados para cada pergunta é de 0 a 5. Nas tabelas constam as médias das respostas para cada pergunta. Ao final, foi compilado todas as médias em uma média geral, sendo esta a nota final considerada para o cenário.

	Cenário 1(1ª sessão)	Cenário 1(2ª sessão)	Cenário 2	Cenário 3
Pergunta 1	2,75	4,75	4,22	5
Pergunta 2	5	4,75	4,71	5
Pergunta 3	4	4,5	4	5
Pergunta 4	3,66	5	4,44	5
Pergunta 5	3,33	5	4,33	5
Pergunta 6	3,66	4,75	3,66	4
Pergunta 7	2,0	5	3,66	5
Media	3,49	4,82	4,14	4,86

Tabela 2 – Médias das avaliações da interface acessível geradas pelo formulário de avaliação ao longo de cada teste.

	Cenário 1(1ª sessão)	Cenário 1(2ª sessão)	Cenário 2
Pergunta 1	4,5	5	4,77
Pergunta 2	5	5	4,75
Pergunta 3	5	5	5
Pergunta 4	4,75	5	5
Pergunta 5	4,25	5	4,33
Pergunta 6	4,5	5	4,22
Pergunta 7	4,25	5	4,22
Media	4,6	5	4,61

Tabela 3 – Médias das avaliações da interface convencional geradas pelo formulário de avaliação ao longo de cada teste.

4.4 Considerações finais

É possível afirmar que o núcleo de jogo desenvolvido alcançou bons resultados. Durante o desenvolvimento, percebeu-se crescente melhoria na avaliação fornecida pelos usuários a cada teste. Estima-se que isso se deu devido às melhorias e correções, que levaram em conta a experiência e recomendações recebidos nos testes anteriores. Com isso, foi possível evoluir a experiência acessível do projeto de uma média de nota 3,49 no primeiro teste para uma nota 4,86 com um usuário portador de deficiência visual, em uma escala de 0 a 5.

Além disso, o núcleo lógico funcional demonstrou bons indícios de escalabilidade e robustez nos testes de desempenho. Nesses testes, foi possível perceber o comportamento do núcleo quando exposto a diferentes cargas de requisições simultâneas e o quanto isso afeta os tempos de resposta e as taxas de erros. Por fim, o núcleo se mostrou eficiente em lidar com cargas simultâneas com uma arquitetura simples e fácil de manter.

5 CONCLUSÕES E TRABALHOS FUTUROS

A acessibilidade é um valor cada vez mais importante e requisitado no desenvolvimento de software e jogos. Pensar em formas de integrar mais pessoas com diferentes deficiências deve ser uma missão intrínseca de cada desenvolvedor.

Percebe-se, ainda, grande carência de ferramentas e modelos de processo que facilitem o desenvolvimento de aplicações acessíveis. Nesse contexto, neste projeto é inserida uma proposta de arquitetura e um exemplo de como um jogo pode possuir interfaces completamente diferentes e, ainda assim, oferecer uma experiência inclusiva, permitindo a integração de jogadores com diferentes capacidades.

Com este projeto, foi possível demonstrar como uma arquitetura baseada no uso de componentes que abstraem a lógica funcional do jogo pode facilitar o desenvolvimento de um jogo com múltiplas interfaces, visto que as regras e as funcionalidades do jogo são providas pelo mesmo servidor, independentemente da interface particular que o usuário esteja usando.

As experiências de construção e os *feedbacks* produzidos pelos voluntários em múltiplos cenários de testes, tornaram amadurecidos os possíveis caminhos para desenvolver um jogo *online* de perguntas e respostas. As interações foram imprescindíveis para construir soluções de *design* que proporcionem um ambiente acessível e integrativo. Muitos aspectos desta experiência podem ser generalizados para construção de outros jogos e aplicações inclusivas.

Como trabalhos futuros, prospectam-se as seguintes melhorias:

- Melhoria no tratamento das falhas de conexão que impactam nas experiências do usuário quando há instabilidades na rede;
- Evolução dos aspectos de concorrência para maior escalabilidade e balanceamento de carga para um número elevado de salas simultâneas, possivelmente com distribuição em diferentes servidores;
- Melhorias na jogabilidade e do suporte a funções acessíveis, a fim de garantir que mais portadores de deficiência possam ser seguramente integrados e incluídos;
- Adição de novas funcionalidades expostas pelo núcleo para que os jogadores possam escolher o tema das perguntas com as quais quiserem jogar;
- Fornecer a possibilidade de o jogador configurar as características do jogo, como limite de jogadores por sala, número de questões por partida, diferentes níveis de dificuldade, entre outros aspectos.

REFERÊNCIAS

Action Cable. **Action Cable Overview — Ruby on Rails Guides**. 2016. https://guides.rubyonrails.org/action_cable_overview.html. (Acessado em 06/14/2023).

Apache JMeter. **Apache Jmeter**. 1998. https://jmeter.apache.org/. (Acessado em 06/14/2023).

BIERRE, K. *et al.* Game not over: Accessibility issues in video games. Academia.edu, 2005.

BLOW, J. Game development: Harder than you think: Ten or twenty years ago it was all fun and games. now it's blood, sweat, and code. ACM Digital Library, 2004.

DOORNBOSCH, P. **pjtr / jmeter-websocket-samplers / Branches — Bitbucket**. 2016. https://bitbucket.org/pjtr/jmeter-websocket-samplers/branches/. (Acessado em 06/14/2023).

Entertainment Software Association. **2022 Essential Facts About the Computer and Video Game Industry**. 2022. https://www.theesa.com/resource/2022-essential-facts-about-the-video-game-industry/. (Acessado em 06/15/2023).

Fly. **Deploy app servers close to your users · Fly**. 2008. https://fly.io/. (Acessado em 06/14/2023).

GALLANT, M. **The Last of Us Part II: Opções de Acessibilidade Detalhadas – PlayStation.Blog BR**. 2020. https://blog.br.playstation.com/2020/06/09/the-last-of-us-part-ii-opcoes-de-acessibilidade-detalhadas/. (Acessado em 06/14/2023).

Grand View Research. **Video Game Market Size, Share Trends Analysis Report By Device (Console, Mobile, Computer), By Type (Online, Offline), By Region (North America, Europe, Asia Pacific, Latin America, MEA), And Segment Forecasts, 2022 - 2030**. 2020. https://www.grandviewresearch.com/industry-analysis/video-game-market. (Acessado em 06/14/2023).

GROSSI, S. F. M. Benefícios do video game em pacientes com sequelas de acidente vascular cerebral. Fisioterapia Brasil, 2017.

Heroku. **Cloud Application Platform | Heroku**. 2009. https://www.heroku.com/home. (Acessado em 06/14/2023).

HöFKE, L. **Acessibilidade: o exemplo de The Last of Us Part II**. 2022. https://clubedovideogame.com.br/acessibilidade-o-exemplo-de-the-last-of-us-part-ii/. (Acessado em 06/14/2023).

Interactive Games and Entertainment Association. **Digital Australia 2022 (DA22) - Connected by Games - IGEA**. 2022. https://igea.net/2021/10/digital-australia-2022-da22-connected-by-games/. (Acessado em 06/14/2023).

KULIK, J.; BEESTON, J.; CAIRNS. Grounded theory of accessible game development. ACM Digital Library, 2021.

MERCIER, M.; LUBART, T. Video games and creativity: The mediating role of psychological capital. **Journal of Creativity**, ScienceDirect, 2023.

PORTER, J. R.; KIENTZ, J. A. An empirical study of issues and barriers to mainstream video game accessibility. ACM Digital Library, 2023.

Postgress. **PostgreSQL: The world's most advanced open source database**. 1996. https://www.postgresql.org/. (Acessado em 06/14/2023).

Puma. **A Fast, Concurrent Web Server for Ruby Rack - Puma**. 2011. https://puma.io/. (Acessado em 06/14/2023).

Redis. **Redis**. 2009. https://redis.io/. (Acessado em 06/14/2023).

RFC 6455. **RFC 6455 - The WebSocket Protocol**. 2011. https://datatracker.ietf.org/doc/html/rfc6455#section-1.1. (Acessado em 06/14/2023).

RIES, E. **The Lean Startup: How Today's entrepreneurs use continuous innovation to create radically successful businesses**. [*S. l.*]: Currency, 2017.

ROLLINGS, A.; MORRIS, D. **Game Architecture and Design: A new edition**. [*S. l.*]: New Riders, 2009.

Ruby on Rails. **Ruby on Rails — A web-app framework that includes everything needed to create database-backed web applications according to the Model-View-Controller (MVC) pattern.** 2004. https://rubyonrails.org/. (Acessado em 06/14/2023).

SAKELLARIOU, A. **How Much The Last of Us Part 2 Cost To Make**. 2020. https://screenrant.com/last-us-2-budget-cost-development-production/. (Acessado em 06/14/2023).

TOH, W.; KIRSCHNER, D. Developing social-emotional concepts for learning with video games. **Computers Education**, ScienceDirect, 2022.