DOI: 10.5748/19CONTECSI/REX/SOC/6946

UMA SOLUÇÃO ONLINE PARA INTEGRAÇÃO DE JOGOS ADAPTÁVEIS MULTI-PLATAFORMA

Teoian Quaresma ; https://orcid.org/0000-0001-9341-2810
Universidade Federal do Pará

Marcelle Mota ; https://orcid.org/0000-0001-9226-9020
Universidade Federal do Pará

UMA SOLUÇÃO ONLINE PARA INTEGRAÇÃO DE JOGOS ADAPTÁVEIS MULTI-PLATAFORMA

AN ONLINE SOLUTION FOR MULTI-PLATFORM ADAPTABLE-GAME INTEGRATION

RESUMO: Jogos educativos estimulam uma aprendizagem mais interessante para a criança que resulta em uma retenção natural do conteúdo ensinado, já que essa abordagem exercita sua criatividade e estimula a interação com outras crianças reforçando a noção do conceito de regras. O acompanhamento de um profissional é necessário, pois será o responsável pela definição dos objetivos e de como este conhecimento estará sendo retido pela criança. Pensando nisso, estudantes do laboratório de pesquisa (anonimizado para revisão) da Universidade (anonimizado para revisão) em parceria com Abrigo (anonimizado para revisão), criaram o projeto, Jogos Adaptáveis, com o intuito de desenvolver jogos totalmente customizáveis, onde o profissional configura o jogo de acordo com o progresso da criança. No entanto, com o crescimento da quantidade de jogos desenvolvidos, surgiu a necessidade de um sistema que os jogos estejam conectados à internet para ter seus dados distribuídos em servidores e, assim, partidas, configurações, progressos e temas estarem acessíveis em qualquer plataforma, independente do dispositivo ou local. Diante dos pontos observados, o objetivo deste trabalho é de implementar um novo *web service* robusto, escalável, fornecedor de recursos que possam ser acessados por diversos dispositivos, sejam ele *mobile*, *tablet* ou *desktop,* possibilitando centralização dos dados em uma única fonte.

ABSTRACT: Educational games encourage more interesting learning for the child, which results in a natural retention of the content taught, as this approach exercises their creativity and encourages interaction with other children, reinforcing the notion of the concept of rules. The follow-up of a professional is necessary because he will be responsible for defining the objectives and how this knowledge will be retained by the child. With that in mind, students from the research lab (anonymized for review) at the (anonymized for review) University, in partnership with Abrigo (anonymized for review), created the Adaptable Games project with the aim of developing fully customizable games, where the professional configures the game according to the child's progress. However, with the growth in the number of games developed, the need arose for a system where the games are connected to the internet to have their data distributed on servers so that the games, settings, progress and themes are accessible on any platform, regardless of the device. or location. In view of the points noted above, the objective of this work is to implement a new robust, scalable web service that provides resources that can be accessed by various devices, be it mobile, tablet or desktop, enabling centralization of data in a single source.

PALAVRAS-CHAVE: Jogo educativo. Desenvolvimento de jogos. Desenvolvimento de web service. Banco de dados.

KEYWORDS: Educational game. Game development. Web service development. Database.

1 INTRODUÇÃO

Jogos educativos estimulam uma aprendizagem mais interessante para a criança resultante em uma retenção natural do conteúdo ensinado, já que essa abordagem exercita sua criatividade e incentiva a interação com outras crianças reforçando a noção do conceito de regras. O acompanhamento de um profissional é necessário, pois será o responsável pela definição dos objetivos e de como esse conhecimento será retido pela criança (FIALHO, 2008).

Em um projeto de conclusão de curso realizado em 2018, por estudantes da Universidade (anonimizado para revisão) em parceria com o Abrigo (anonimizado para revisão), foi desenvolvido um sistema com o objetivo de se adaptar a necessidade do paciente de acordo com a configuração feita pelo profissional responsável. O estudo coletou resultados que comprovam uma melhor atenção por parte do paciente ao conteúdo ensinado. Esse sistema trata-se de um jogo de memória chamado Cuca Fresca, cujo propósito é se adaptar de acordo com a necessidade física motora dos pacientes e conta com um jogo disponível para *tablets* e *smartphones.* O jogo possui uma arquitetura online que contém banco de dados, servidor de arquivos, e um *web service* (DE CARVALHO, 2019). O projeto deu início a implementação de diversos outros jogos, por exemplo, o Ritmo Mania e o Quebra Cabeça. Com isso, surgiram novos requisitos os quais destacam a necessidade de um *web service* mais robusto, escalável e que atenda a possibilidade de ser acessado por qualquer outro serviço, independente de plataforma (TADAIESKY et al, 2021).

Diante dos pontos levantados, o objetivo deste trabalho é de implementar um novo *web service* robusto, escalável, utilizando tecnologias modernas e populares no mercado, e familiar por grande parte dos desenvolvedores, garantido um sistema manutenível. O *web service* proporciona também com que os usuários acessem os jogos em qualquer dia, através de quaisquer dispositivos configurados pelos profissionais assim como terão seus dados, configurações, temas e progressos salvos e sincronizados. Além disso, cria a possibilidade da integração de novos sistemas de gerenciamento de dados, para a geração de relatórios, por exemplo.

2 METODOLOGIA

Para levantar os requisitos, foram utilizadas reuniões com membros da equipe de desenvolvimento, resultando em documentos como as estórias de usuário, a modelagem do banco de dados e os fluxogramas de integração entre a *web service* e as aplicações dos clientes. Nesse sentido, foi realizado revisão bibliográfica de livros e artigos, de onde também, foram retirados requisitos que embasaram as escolhas das tecnologias e o desenvolvimento deste artigo.

Além disso, com a finalidade de obter dados para apoiar o resultado obtido, foi elaborada uma pesquisa *survey* com perguntas de múltipla escolha baseadas na escala Likert, com cinco opções, indo de "discordo totalmente" até "concordo plenamente", com o foco de avaliar a experiência dos desenvolvedores que irão integrar seus jogos no *web service* proposto. O questionário foi realizado através do Google Forms, formulário onde os entrevistados respondem as perguntas online e gera relatórios com gráficos e planilha contendo as respostas dos entrevistados.

3 DESENVOLVIMENTO

Após reuniões com membros do laboratório de pesquisa (anonimizado para revisão), laboratório responsável pelo desenvolvimento dos jogos adaptáveis, foram identificados novos casos de usos e uma nova modelagem do banco de dados para atender a adição de novos jogos centralizados em uma única plataforma de gerenciamento dos dados obtidos nas partidas. Para atender essa demanda, foi necessário a construção de um *web service* utilizando os conceitos de REST, pois é necessária uma comunicação rápida e segura entre a plataforma e os jogos, sendo esse conceito bastante popular, sem deixar de citar, a amplitude da utilização no mercado (NOLETO, 2022). O repositório do projeto encontra-se disponível neste link (anonimizado para revisão).

3.1 TECNOLOGIAS UTILIZADAS

A linguagem empregada para o desenvolvimento do *web service* foi o Javascript, que segundo (CASS, 2022), é uma das linguagens mais populares e pode ser utilizada tanto na API quanto na plataforma web. Nesse sentido, usando o Typescript, como explica (CHERNY, 2019), torna a aplicação mais escalável devido ao grande conjunto de funcionalidades que essa linguagem oferece como encapsulamento, herança, abstração e o polimorfismo.

Com o intuito de construir um *software* manutenível, escalável, auto testável e que atende a boas práticas de programação, foi escolhido usar o *framework* NestJs que tem uma arquitetura altamente opinada, utiliza Typescript. Ademais, possui uma sólida semelhança com a arquitetura do Angular, *framework,* bastante popular no mercado, o que traz familiaridade para desenvolvedores *front-end* que venham a programar o presente sistema (PHAM, 2020).

Para a realização de testes *end-to-end,* método de teste usado para testar um fluxo da aplicação desde o começo até o fim, foi usada a plataforma de API Postman. Ela é útil não só para auxiliar no desenvolvimento da API, mas também é útil para os desenvolvedores que irão consumir o *web service,* uma vez que essa ferramenta permite acessar os recursos da API sem a necessidade de ter uma aplicação cliente, pois ela assume esse papel.

Foi operado o mapeamento objeto-relacional Prisma ORM para se comunicar com o base de dados. Isso simplificou drasticamente a modelagem de dados, migrações e acesso a dados para bancos de dados SQL em Javascript ou TypeScript (KAUPPI, 2022). Além disso, o uso dessa tecnologia diminui a dependência de *queries* programadas na linguagem SQL, já que todas as consultas são realizadas com notações de funções de Javascript ou Typescript.

O sistema online foi hospedado na plataforma de nuvem como serviço Heroku. A plataforma oferece serviços de hospedagem gratuitos e nela é possível adicionar complementos de base de dados como o Heroku Postgres, serviço esse escolhido para o banco de dados em produção. Desse modo, facilita a análise de *logs*, o que ajuda a encontrar erros na aplicação e, ainda conta, com *deploy*

automático, bastando apenas subir o código para o repositório remoto e automaticamente a aplicação implanta as alterações em produção.

O *upload* de arquivos foi implementado com os serviços de armazenamento de mídias do Firebase, o Firestorage, produto da empresa Google. Essa ferramenta permite o envio de imagem, vídeo, áudio, entre outros tipos de mídias e arquivos. O serviço é gratuito até determinada quantidade de memória utilizada. Usando essa ferramenta, os desenvolvedores dos jogos poderão fazer upload de mídias dos jogos ou de tema, e salvar esse local posteriormente no banco de dados.

3.2 MODELAGEM DO BANCO DE DADOS

No banco de dados, temos as tabelas *institutions* e *professionals* que representam a entidade do usuário da aplicação, pois é na instituição que estarão os profissionais e os pacientes que na modelagem é representada pela tabela *children*. A tabela *groups,* que têm relação com outras tabelas pivô, trata-se de armazenar os grupos de crianças e os grupos de profissionais. A necessidade se dá para atender a funcionalidade de temas compartilhados de um profissional para um grupo de crianças ou entre um grupo com os próprios profissionais da mesma instituição.

As tabelas que representam a entidade de um jogo são compostas pelas tabelas *games*, *games_categories* e *match_history*. Foi pensado em ter uma tabela para cada jogo, por exemplo, uma tabela chamada *cuca_fresca* ou *ritmo_mania,* mas após reuniões com os desenvolvedores desses jogos, observou-se que cada jogo tinha atributos específicos e não seria escalável essa abordagem. Então, decidiu-se criar uma tabela genérica somente para representar o jogo e, outra tabela, somente para armazenar as configurações daquele jogo no momento em que a criança jogou, gerando assim, um histórico que o profissional poderá analisar posteriormente. Essa também salva informações sobre o tema que é representado pela tabela *themes* que tem relação com a tabela de *medias,* cuja responsável pelas informações dos arquivos de mídias do tema.

3.3 WEB SERVICE

A arquitetura do software foi totalmente baseada na arquitetura em camada do *framework* NestJs, pois, a primeiro momento, não se entendeu a necessidade de construir algo mais complexo, como levanta (DA CONCEIÇÃO, 2021). Isso foi o principal motivo da escolha deste *framework*, já que ele traz uma arquitetura bastante opinada e familiar (PHAM, 2020). Portanto, esse *framework* é ideal para a utilização do padrão REST (Representational State Transfer) que propõe um conjunto de restrições a serem usadas por quem gera os recursos e para quem consome.

Para que os desenvolvedores pudessem utilizar a aplicação e entender quais pontos de extremidade acessar, foi configurado o acesso aos recursos da API no Postman, com exemplos de

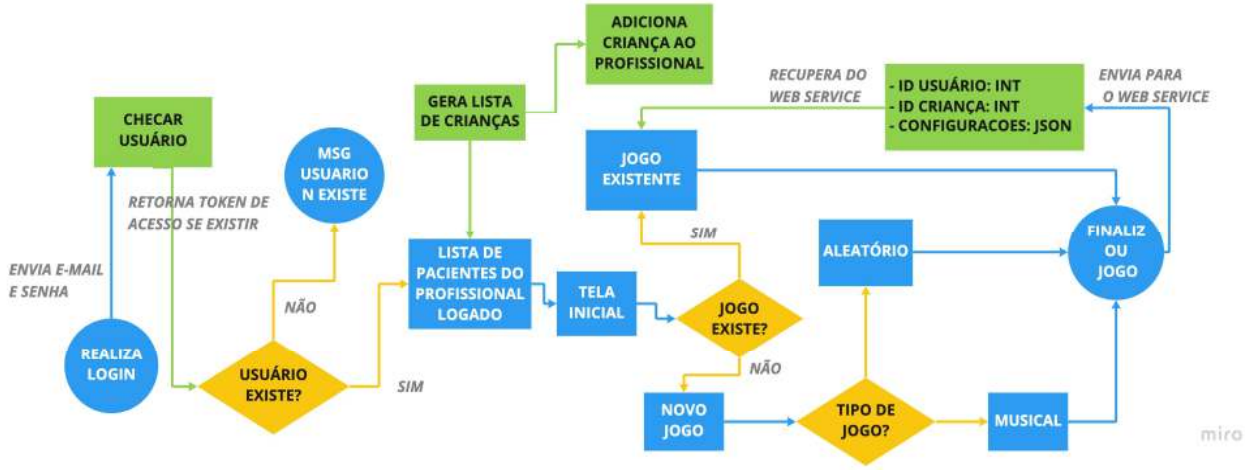
requisições e respostas. Além disso, foi elaborada uma documentação no próprio Postman contendo informações sobre como obter os recursos, com exemplos de requisições e respostas em diversos tipos de linguagens de programação e contendo informações de acessos, como *token*, *id* de jogos, etc. Dessa forma, o objetivo é de garantir uma melhor experiência para quem estiver consumindo o *web service*.

Para realizar uma requisição, o cliente precisa estar autenticado e enviar um *token* temporário no *Header* da requisição. O *token* permite que o sistema identifique o usuário e verifique quais recursos ele pode ter acesso. O *token* tem validade de 24 horas e, após esse período, ele é expirado, sendo necessário gerá-lo novamente para acessar a API. Isso garante segurança na aplicação, pois, além de proteger as rotas de usuários que não pertencem a aplicação, ainda protege os recursos de usuários sem permissão.

Os recursos deste serviço serão acessados pelos jogos adaptáveis e por aplicações futuras que poderão ter a intenção de obter acesso aos dados fornecidos pelo serviço. O fluxograma da Figura 1, representa como o jogo realizará seu fluxo trabalhando em conjunto com o *web service.* Para exemplificar, foi selecionado o fluxograma do jogo Ritmo Mania. No entanto, esse fluxo também é semelhante para outros jogos, como o Quebra Cabeça citado em (TADAIESKY et al, 2021).

O jogo Ritmo Mania é representado pelos ícones de cor azul, e o fluxograma, de cor verde, é o *web service.* O primeiro passo do jogo, é verificar se o usuário está autenticado. O jogo envia uma requisição solicitando o *token* de acesso, caso o usuário exista e suas credenciais estejam corretas, o servidor retorna este *token* concedendo autorização ao usuário. O próximo passo, é a solicitação da lista de crianças ao profissional, então, o usuário seleciona a criança para jogar a partida atual. Antes de iniciar a partida, ele verifica localmente se já existe alguma configuração do jogo no *web service,* caso já exista o jogo, carrega essas configurações do servidor, caso não, ele inicia uma nova partida. No final de cada partida, o jogo envia uma última requisição com as configurações da partida atual para que sejam carregadas em uma próxima partida.

Figura 1 — Fluxograma do jogo Ritmo Mania integrado ao *web service.*



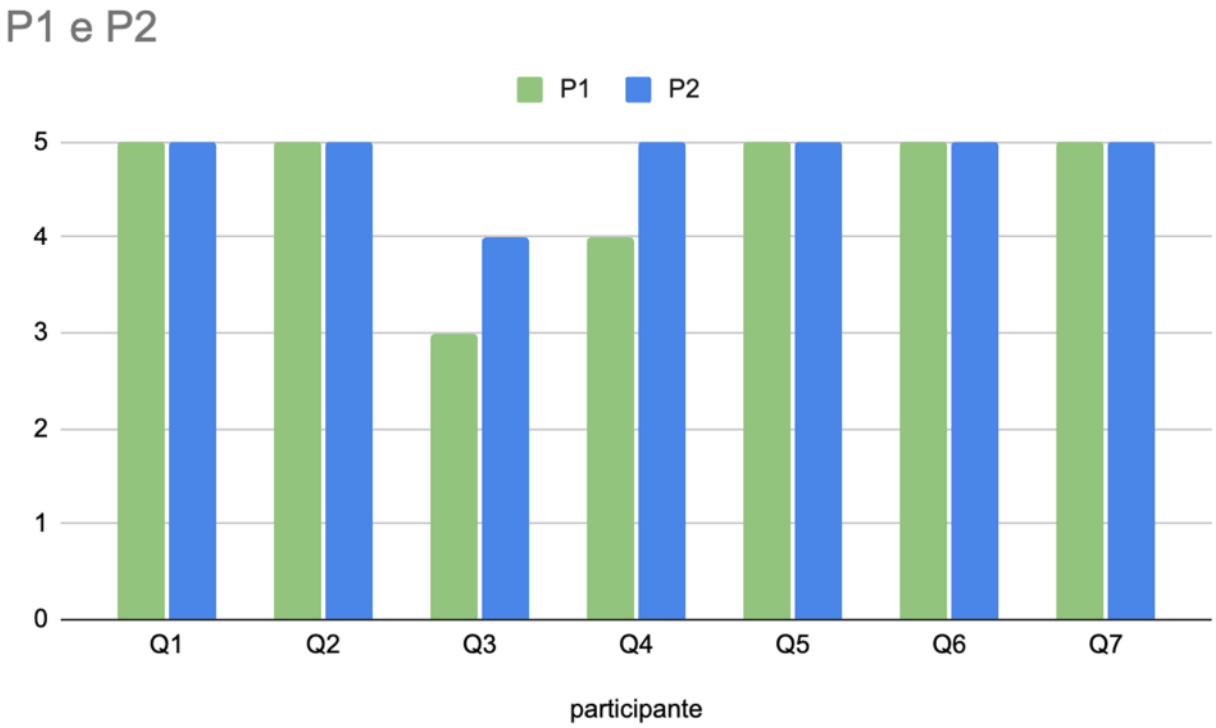
Fonte: Próprio autor.

4 RESULTADOS OBTIDOS

Para a realização dos testes, foram selecionados dois desenvolvedores disponíveis (P1 e P2) que testaram a integração do *web service* e, em seguida, responderam ao questionário de 7 questões. As questões foram enumeradas de 1 a 5, na escala Likert, que variam de “discordo plenamente” a “concordo plenamente”. Os candidatos responderam as seguintes questões: "a documentação foi útil para o entendimento do *web service*?” Q1, "os *endpoints* disponíveis atenderam suas expectativas?” (Q2), "com relação ao bom desempenho do servidor, o quão você concorda?” (Q3), "concorda que o *web service* é seguro?” (Q4), "você acredita que a API pode ser integrada com qualquer cliente, independente do dispositivo?” (Q5), "você acredita que o *web service* facilitará na centralização e na coleta dos dados das partidas dos jogos?” (Q6) e "você concorda que essa solução online agrega valor aos jogos adaptáveis no geral?” (Q7).

Ambos os participantes concordaram plenamente com as questões Q1, Q2, Q5, Q6 e Q7. As Q3 e Q4 não tiveram o mesmo desempenho, porque o servidor que fornece a API tem recursos limitados, em virtude da sua gratuidade, logo, impacta no desempenho do serviço. O participante P1 afirma que o serviço é seguro, porém sem a certeza, pelo fato de ter sido testado somente em ambiente de desenvolvimento.

Gráfico 1 — Resultado das repostas dos desenvolvedores



Fonte: Próprio autor

5 CONSIDERAÇÕES FINAIS

O trabalho, apresentado neste artigo, buscou implementar um *web service* como solução para a integração entre os jogos adaptáveis, com uma plataforma onde o usuário responsável por configurar os jogos e gerenciar os jogadores, que possa ter acesso a esses dados online e que os jogos possam ter seus dados salvos e sincronizados em qualquer dispositivos. Para as validações dos itens citados, foi elaborado um questionário onde os desenvolvedores dos jogos testaram a integração com o *web service* para, desse modo, avaliar de acordo com as questões abordadas na seção anterior.

Com os resultados obtidos com a pesquisa, foi possível certificar de que a aplicação desenvolvida aqui está atendendo ao que se propõe e entrega valor no ponto de vista técnico dos jogos adaptáveis. Como trabalhos futuros, há a necessidade de implementação da plataforma para o profissional acessar dados de seu perfil, da instituição e de seus pacientes e, terá também, a possibilidade de gerenciar seus pacientes, analisar seus desempenhos, para assim, gerar relatórios precisos a respeito da evolução da criança nas partidas jogadas.

REFERÊNCIAS

WERNER, Hilda Maria Leite. **O processo da construção do número, o lúdico e tics como recursos metodológicos para criança com deficiência intelectua**l. Secretaria do Estado de Educação Superintendência da Educação Diretoria de Políticas e Programas Educacionais Programa de Desenvolvimento Educacional–PDE. Paranaguá-PR, 2008.

DE CARVALHO, Caio Pinheiro et al. **A study on customizing interaction in adaptable games**. In: Proceedings of the 18th Brazilian Symposium on Human Factors in Computing Systems. 2019. p. 1-11.

TADAIESKY, Von Harisson et al. **Um Estudo de Caso Sobre Jogos Adaptáveis: Ritmo Mania e Quebra-Cabeça**. In: Anais Estendidos do XX Simpósio Brasileiro de Jogos e Entretenimento Digital. SBC, 2021. p. 981-984.

FIALHO, Neusa Nogueira. **Os jogos pedagógicos como ferramentas de ensino**. VIII Congresso Nacional de Educação, p. 1-9, 2008.

CHERNY, Boris. **Programming TypeScript: making your JavaScript applications scale**. O'Reilly Media, 2019.

CASS, Stephen. **Top Programming Languages 2022**. IEEE Spectrums, 23 ago. 2022. Disponível em: https://spectrum.ieee.org/top-programming-languages-2022/ieee-spectrums-top-programming-languages-2022, acesso em 23 set. 2022.

PHAM, Anh Duc. **Developing back-end of a web application with NestJS framework:** Case: Integrify Oy's student management system. LAB Univercity of Applied Sciences, 2020.

KAUPPI, Jonne. **Tietokantakäsittely asiakasprojektille Prisman ja GraphQL**: n avulla. 2022.

DA CONCEIÇÃO, Melissa Tidori; PINTO, Giuliano Scombatti. **ARQUITETURA DE MICROSSERVIÇOS**. Revista Interface Tecnológica, v. 18, n. 2, p. 53-64, 2021.