



Campus Professor Barros

RELATÓRIO A3 USABILIDADE, DESENVOLVIMENTO WEB E MOBILE

Salvador - BA
2023

Diego Pimenta dos Anjos;
Fernando de Caires Gonçalves;
Lucca Cintra Poggio;
Reinan Carvalho Amaral;
Silvestre Carlos Elioterio Neto.

DOCUMENTAÇÃO A3 SCS

Trabalho da disciplina de Usabilidade, desenvolvimento Web, mobile e jogos. Neste documento iremos apresentar o nosso relatório referente à A3, na qual desenvolvemos uma aplicação WEB utilizando as tecnologias HTML, CSS, JAVASCRIPT (REACT.JS) no front-end e NODE.JS com Express para o back-end.

Orientador: Thiago Dotto Fiuza Neves

Gamers Library

Gamers Library é uma plataforma online dedicada a fornecer uma experiência completa e organizada para os entusiastas de videogames que desejam gerenciar, descobrir e compartilhar sua coleção de jogos. Esta "biblioteca de jogos" virtual é um espaço onde jogadores podem catalogar, classificar e explorar seus jogos de maneira conveniente.

Gamers Library é o lugar perfeito para os amantes de jogos que desejam manter um registro organizado de suas experiências, descobrir novos títulos e se conectar com outros jogadores. É uma ferramenta versátil que atende às necessidades de colecionadores de jogos, jogadores casuais e entusiastas dedicados, tornando a gestão de sua biblioteca de jogos uma experiência mais rica e agradável. Seja você um jogador de PC, console ou mobile, Gamers Library está aqui para unir a comunidade de jogadores e aprimorar a paixão pelos videogames.

O que são as heurísticas de Nielsen ?

As heurísticas de Nielsen referem-se a um conjunto de diretrizes ou princípios de usabilidade criados por Jakob Nielsen, um renomado especialista em usabilidade e design de interação. Essas heurísticas são usadas para avaliar a usabilidade de sistemas, aplicativos, sites e interfaces de usuário. Elas são uma ferramenta valiosa para identificar problemas de usabilidade e melhorar a experiência do usuário. As heurísticas de Nielsen são compostas por dez princípios fundamentais, que são os seguintes:

Visibilidade do Estado do Sistema: O sistema deve manter os usuários informados sobre o que está acontecendo por meio de feedback apropriado no tempo adequado. Os usuários devem saber o estado do sistema e o que está acontecendo em cada momento.

Correspondência entre o Sistema e o Mundo Real: O sistema deve falar a linguagem dos usuários, com palavras, frases e conceitos familiares, em vez de termos técnicos ou jargões. Deve seguir o modelo mental do usuário.

Controle e Liberdade do Usuário: Os usuários devem ter a liberdade de voltar, desfazer ações ou sair do sistema a qualquer momento. As ações irreversíveis devem ser evitadas, e as opções de saída devem ser claramente identificadas.

Consistência e Padrões: Os sistemas devem seguir convenções de design e interação comuns para que os usuários possam prever o que acontecerá. Consistência na apresentação, terminologia e ações é fundamental.

Prevenção de Erros: O sistema deve ser projetado para evitar erros, se possível, ou ajudar os usuários a recuperar-se facilmente de erros quando eles ocorrerem. Isso inclui projetar interfaces que não levem a ações indesejadas por acidente.

Reconhecimento em Vez de Lembrança: Os usuários não devem precisar lembrar informações de uma parte do sistema para outra. Em vez disso, as informações relevantes devem ser visíveis ou facilmente recuperáveis quando necessário.

Flexibilidade e Eficiência de Uso: Os sistemas devem atender tanto a usuários novatos quanto a especialistas, permitindo que os usuários realizem tarefas da maneira mais eficiente possível.

Estética e Design Minimalista: A informação presente na interface deve ser relevante e não excessiva. O design deve ser limpo e focado nas tarefas dos usuários.

Ajuda aos Usuários a Reconhecer, Diagnosticar e Recuperar-se de Erros: O sistema deve comunicar mensagens de erro de forma clara e indicar como os usuários podem corrigir esses erros.

Ajuda e Documentação: Embora a interface do sistema deva ser autoexplicativa na medida do possível, é importante fornecer ajuda e documentação quando necessário. Essa documentação deve ser fácil de encontrar e usar.

Essas heurísticas fornecem diretrizes gerais que os designers e especialistas em usabilidade podem seguir para avaliar a usabilidade de um sistema ou interface de usuário. Ao identificar problemas que violam essas heurísticas, é possível melhorar a experiência do usuário e tornar os sistemas mais eficazes e fáceis de usar.

Wireframes

No desenvolvimento no GamersLibrary levamos em consideração algumas das heurísticas citadas acima como pontos focais do nosso desenvolvimento, sendo elas: **Visibilidade do Estado do Sistema, Estética e Design Minimalista e Prevenção de Erros.** As telas apresentadas abaixo seguirão a ordem da tela no desktop e posteriormente a versão mobile.

TELA DE INÍCIO

Desktop - 1



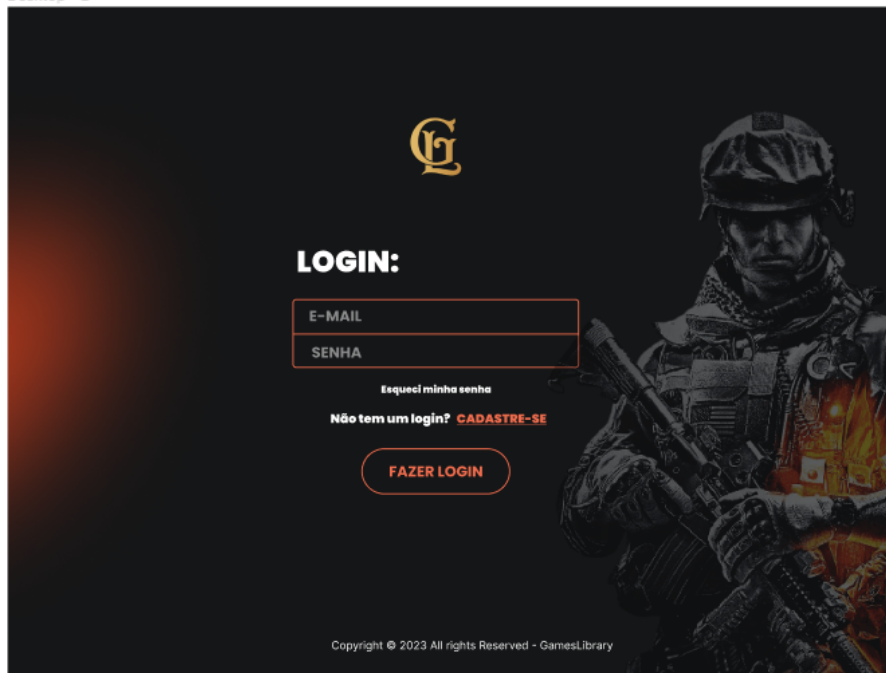
iPhone 14 & 15 Pro Max - 1



Salvador - BA
2023

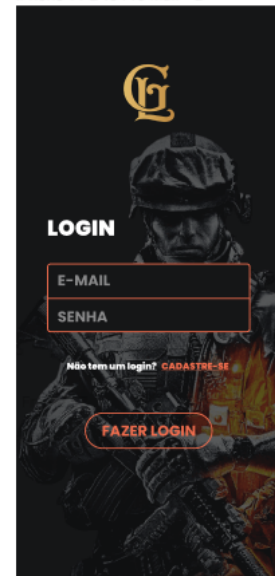
TELA LOGIN

Desktop - 2



The desktop login form is centered on a dark background with a soldier in tactical gear. At the top is a gold 'GL' logo. Below it, the text 'LOGIN:' is in white. There are two input fields: 'E-MAIL' and 'SENHA'. Below the 'SENHA' field is a link 'Esqueci minha senha'. Below that is a link 'Não tem um login? CADASTRE-SE'. At the bottom is a rounded button labeled 'FAZER LOGIN'. The footer contains the text 'Copyright © 2023 All rights Reserved - GamesLibrary'.

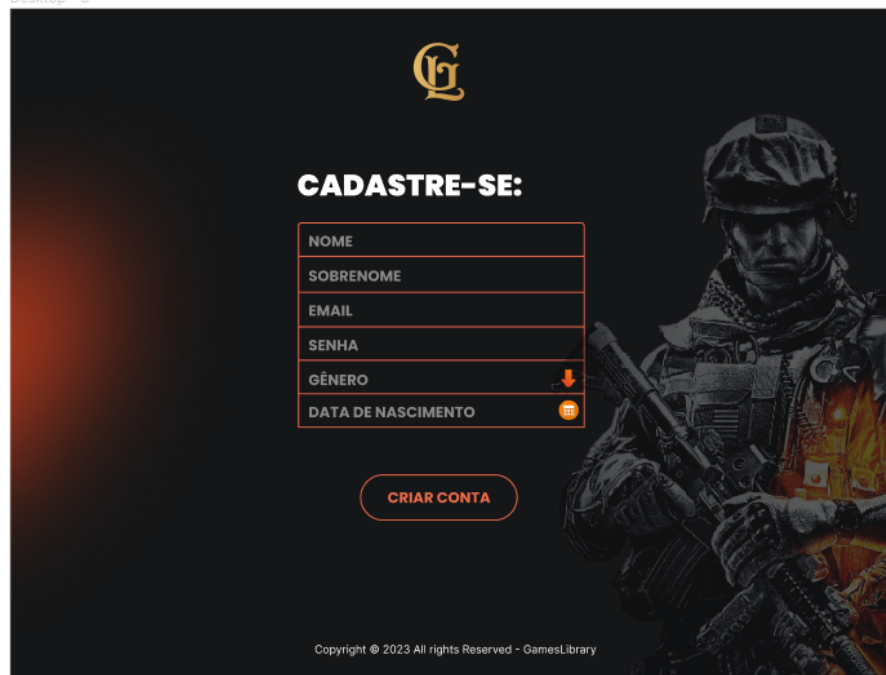
iPhone 14 & 15 Pro Max - 2



The iPhone login form is a mobile-optimized version of the desktop one. It features the same 'GL' logo, 'LOGIN:' text, and input fields for 'E-MAIL' and 'SENHA'. The links for 'Esqueci minha senha' and 'Não tem um login? CADASTRE-SE' are present. The 'FAZER LOGIN' button is rounded and centered. The footer text 'Copyright © 2023 All rights Reserved - GamesLibrary' is at the bottom.

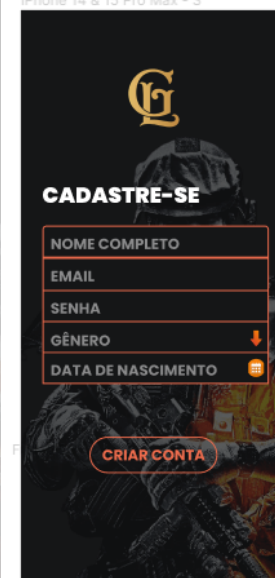
TELA CADASTRO

Desktop - 3



The desktop registration form is centered on a dark background with a soldier in tactical gear. At the top is a gold 'GL' logo. Below it, the text 'CADASTRE-SE:' is in white. There are six input fields: 'NOME', 'SOBRENOME', 'EMAIL', 'SENHA', 'GÊNERO' (with a dropdown arrow), and 'DATA DE NASCIMENTO' (with a calendar icon). Below the fields is a rounded button labeled 'CRIAR CONTA'. The footer contains the text 'Copyright © 2023 All rights Reserved - GamesLibrary'.

iPhone 14 & 15 Pro Max - 3

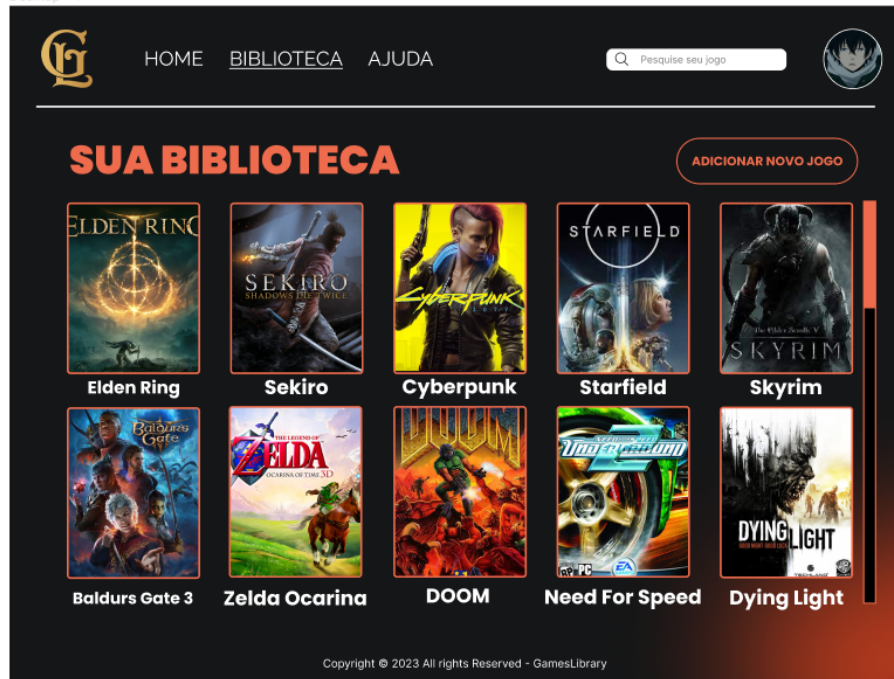


The iPhone registration form is a mobile-optimized version of the desktop one. It features the same 'GL' logo, 'CADASTRE-SE:' text, and input fields for 'NOME COMPLETO', 'EMAIL', 'SENHA', 'GÊNERO' (with a dropdown arrow), and 'DATA DE NASCIMENTO' (with a calendar icon). The 'CRIAR CONTA' button is rounded and centered. The footer text 'Copyright © 2023 All rights Reserved - GamesLibrary' is at the bottom.

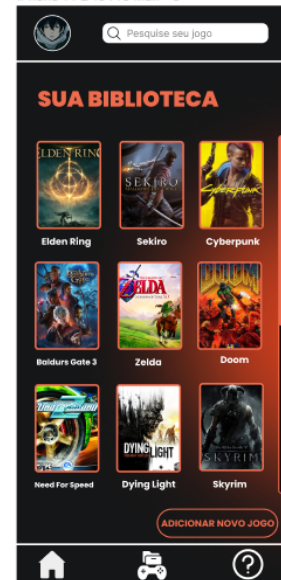
Salvador - BA
2023

BIBLIOTECA

Desktop - 4

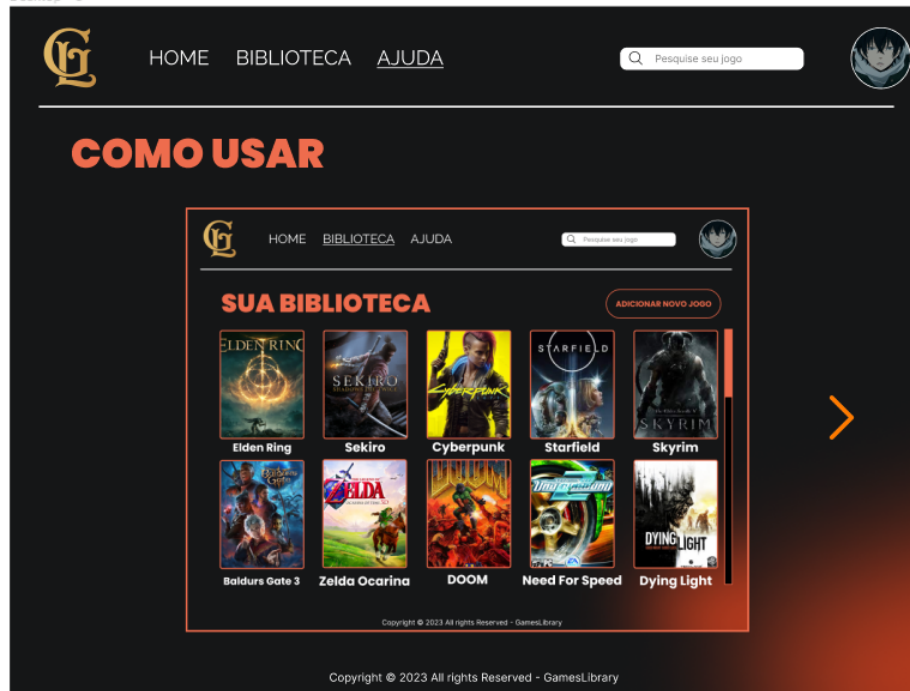


iPhone 14 & 15 Pro Max - 5

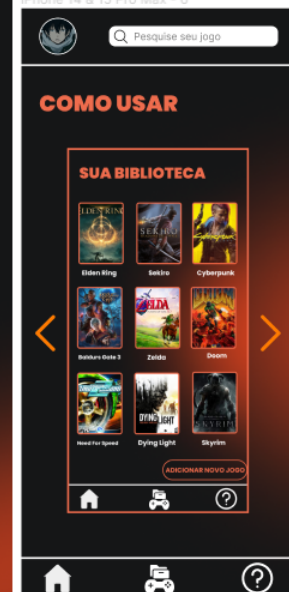


COMO USAR

Desktop - 5



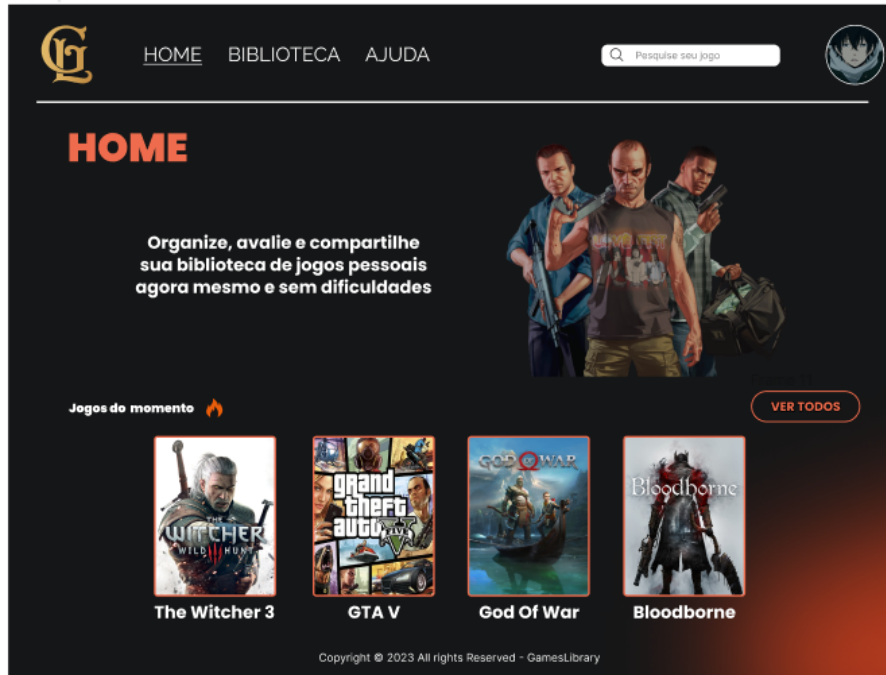
iPhone 14 & 15 Pro Max - 6



Salvador - BA
2023

TELA HOME

Desktop - 6



iPhone 14 & 15 Pro Max - 4



TELA ERRO 404

Desktop - 7



iPhone 14 & 15 Pro Max - 7

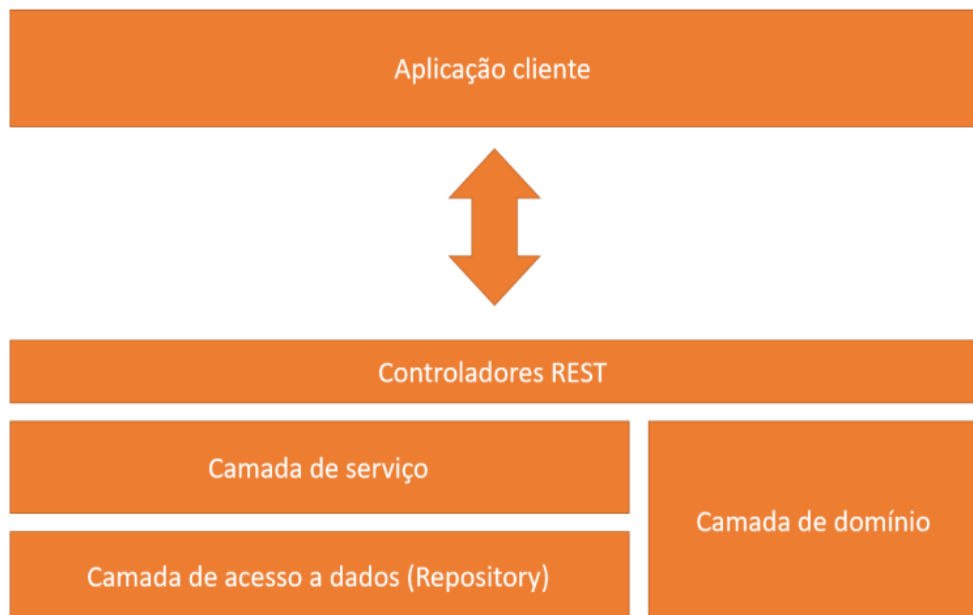


Salvador - BA
2023

Arquitetura do projeto

O relatório descreve o desenvolvimento do GamersLibrary utilizando a arquitetura REST (Representational State Transfer). A arquitetura REST é um estilo arquitetural amplamente adotado para o design de sistemas distribuídos, proporcionando uma abordagem simples e eficiente para comunicação entre componentes.

O sistema foi desenvolvido com o objetivo de fornecer uma plataforma eficiente e escalável para gerenciar o catálogo de jogos do usuário.



Implementação da Arquitetura REST no Sistema

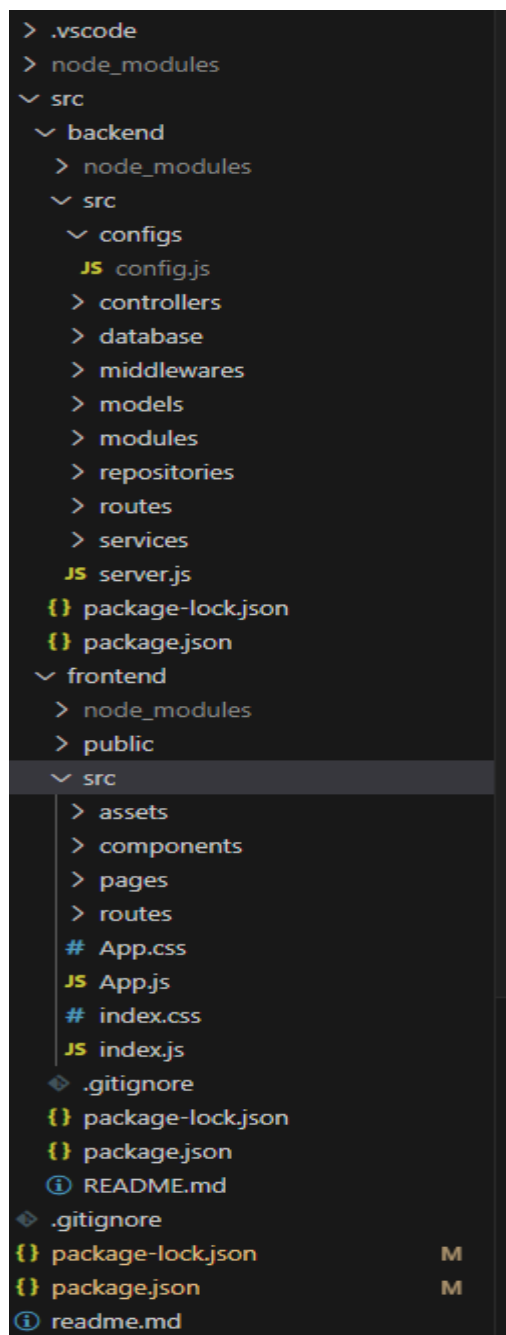
1- Recursos: Identificamos os principais recursos do sistema e atribuímos URIs significativas a cada um deles.

2- Operações HTTP: Utilizamos as operações HTTP padrão para manipulação de recursos - GET para leitura, POST para criação, PUT para atualização e DELETE para exclusão.

3- Representação de Dados: Optamos por representar os dados em formato JSON devido à sua simplicidade, legibilidade e ampla aceitação.

4- Endpoints e Controladores: Criamos endpoints RESTful e controladores que mapeiam as solicitações HTTP para as operações correspondentes nos recursos.

A arquitetura REST provou ser uma escolha eficaz para o desenvolvimento deste sistema, proporcionando simplicidade, flexibilidade e escalabilidade. A abordagem centrada em recursos e a comunicação stateless contribuíram para a criação de um sistema robusto e de fácil manutenção.



Exemplificação da nossa arquitetura.

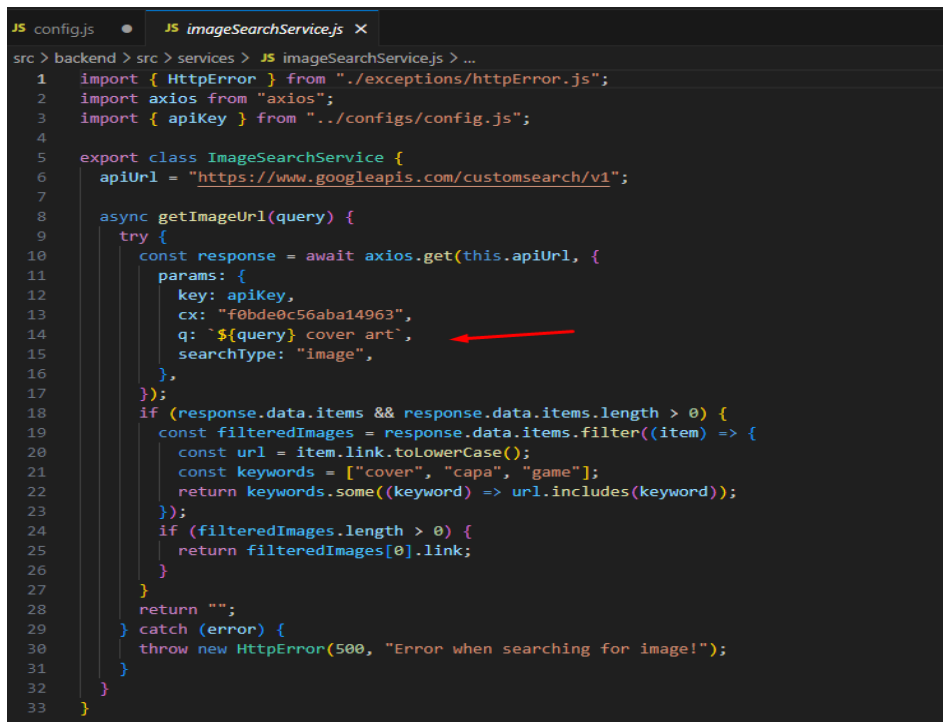
Comunicação do frontEnd com o backEnd

Em uma arquitetura REST, a comunicação entre o frontend e o backend ocorre por meio de requisições HTTP. O frontend envia solicitações para URIs específicas que representam recursos no backend, utilizando métodos HTTP como GET, POST, PUT ou DELETE. O backend processa essas solicitações e responde com códigos de status HTTP e dados no formato desejado, frequentemente JSON. Essa abordagem desacoplada e baseada em padrões facilita a interoperabilidade e escalabilidade do sistema, permitindo uma comunicação eficiente e flexível entre as camadas do aplicativo.

Descrição API Google Custom Search

Para realizar o upload de imagens do jogo adicionado a nossa biblioteca, utilizamos a API Google Custom Search.

A API Google Custom Search é um serviço oferecido pela Google que nos permitiu criar uma experiência de pesquisa personalizada em seus próprios aplicativos ou sites. Ao integrar essa API, os desenvolvedores podem utilizar a infraestrutura de busca poderosa da Google, aplicando filtros específicos para personalizar os resultados de pesquisa de acordo com suas necessidades, ou seja, ao usuário escrever o nome do jogo em que ele irá adicionar a nossa biblioteca, a API faz uma pelo Google Imagens do nome do jogo + “cover art”, o primeiro resultado desta pesquisa será a imagem a ser adicionada no respectivo jogo.



```
JS config.js • JS imageSearchService.js X
src > backend > src > services > JS imageSearchService.js > ...
1 import { HttpError } from "../exceptions/httpError.js";
2 import axios from "axios";
3 import { apiKey } from "../configs/config.js";
4
5 export class ImageSearchService {
6   apiUrl = "https://www.googleapis.com/customsearch/v1";
7
8   async getImageUrl(query) {
9     try {
10       const response = await axios.get(this.apiUrl, {
11         params: {
12           key: apiKey,
13           cx: "f0bde0c56aba14963",
14           q: `${query} cover art`,
15           searchType: "image",
16         },
17       });
18       if (response.data.items && response.data.items.length > 0) {
19         const filteredImages = response.data.items.filter((item) => {
20           const url = item.link.toLowerCase();
21           const keywords = ["cover", "capa", "game"];
22           return keywords.some((keyword) => url.includes(keyword));
23         });
24         if (filteredImages.length > 0) {
25           return filteredImages[0].link;
26         }
27       }
28       return "";
29     } catch (error) {
30       throw new HttpError(500, "Error when searching for image!");
31     }
32   }
33 }
```