REVISTA CONEXÃO ELETRÔNICA

PLATAFORMA DE GERENCIAMENTO DE BIBLIOTECA INTEGRADO AO APLICATIVO *WHATSAPP*

Breno Protz Dias[1]; Henrique Tufano Alvarenga[1]; Richard Vieira do Espirito Santo[2]*

[1] Graduando em Engenharia da Computação, Faculdades Integradas de Três Lagoas – FITL/AEMS; [2]Esp. em Engenharia de software – Universidade Estácio de Sá; [3] Esp. em Segurança da Informação – Instituto de Gestão e Tecnologia da Informação, docente das Faculdades Integradas de Três Lagoas – FITL/AEMS
* autor correspondente: richard_ves@hotmail.com

RESUMO
Neste artigo apresentaremos um aplicativo de gerenciamento de biblioteca que se integra ao WhatsApp, transformando a experiência com a biblioteca em algo mais acessível e eficiente para os usuários. Esse aplicativo foi concebido por uma equipe de alunos de engenheiros de computação e seus respectivos orientadores, com o propósito de aprimorar o acesso aos recursos da biblioteca e facilitar a comunicação entre os usuários e os bibliotecários. Este mesmo oferece uma ampla gama de funcionalidades, incluindo catalogação de livros, pesquisa avançada, opções para empréstimos e devoluções, reservas online, notificações diretamente pelo WhatsApp, assistência direta de bibliotecários, avaliações e comentários sobre os livros e informações sobre os eventos da biblioteca. O seu objetivo primordial é simplificar o processo de localização e acesso aos livros desejados, promovendo a interação dentro da comunidade de leitores e proporcionando um suporte eficaz através do WhatsApp. Além disso, esse aplicativo contribui para a relevância contínua das bibliotecas na era digital, tornando-as mais acessíveis e alinhadas com as necessidades dos usuários modernos. Em outras palavras, essa inovação representa um grande avanço no universo das bibliotecas, unindo a riqueza do conhecimento impresso com as facilidades da tecnologia atual, tornando as bibliotecas um recurso vital e dinâmico em nossa era digital.

Palavras-chave: gerenciamento virtual biblioteca; livros; WhatsApp; gerenciamento; comunicação.

1 INTRODUÇÃO

Uma biblioteca integrada ao WhatsApp representa uma ferramenta fundamental para otimizar e expandir as capacidades desta popular plataforma de comunicação. Trata-se de um conjunto de recursos e interfaces de programação de aplicativos (APIs, do inglês *application programming interfaces*) projetados para permitir a integração do WhatsApp com outros sistemas, aplicativos de terceiros e serviços diversos (CORTEZ, 2021).

Essa integração é crucial para atender às crescentes demandas dos usuários e das empresas que utilizam o WhatsApp como canal de comunicação. A biblioteca oferece uma variedade de aplicações e permite a personalização das interações com os usuários e a automação de processos de negócios, tornando possível fornecer um atendimento ao cliente mais eficaz (CORTEZ, 2021).

1.1 Biblioteca tradicional

Uma biblioteca, no contexto tradicional, é uma coleção de livros e documentos organizada de tal forma para facilitar o acesso a consultas a fim de estudo e leitura. Ela personifica uma inesgotável riqueza de informações e de sabedoria que desempenha um papel crucial na educação e na preservação da cultura. Dentro de seus recintos, cada livro é mais do que uma simples obra; é um condutor de pensamentos, um elo para o passado e uma janela para o mundo (ANDRADE,1942).

A biblioteca se manifesta como um porto seguro para aqueles que buscam respostas, oferecendo uma vasta coleção de recursos que abrange desde as páginas de obras impressas até o vasto universo de recursos digitais e preciosos arquivos históricos (ANDRADE,1942).

A biblioteca, com seu ambiente sereno e acolhedor é um ambiente onde o silêncio, em vez de ser vazio, é a força motriz que nutre a criatividade e amplia a compreensão. As aplicações de uma biblioteca são verdadeiramente variadas ela não apenas se destina a ser um depósito de informações para pesquisa acadêmica, mas também se apresenta como um espaço aberto à troca de ideias, um palco para o encontro de mentes inquisitivas e ávidas por conhecimento. Com a passagem do tempo, as bibliotecas modernas evoluíram para incorporar tecnologia de ponta, oferecendo acesso a recursos digitais e espaços multifuncionais que fomentam o trabalho colaborativo e a inovação. Assim, a biblioteca tradicional se reinventa continuamente, mantendo-se como uma luz-guia para a busca do saber e o enriquecimento intelectual (LESSA; LINS,2021).

1.2 Aplicativo *WhatsApp*

O WhatsApp, uma das aplicações de mensagens mais populares do mundo, é amplamente conhecido por sua interface de usuário intuitiva e recursos de comunicação versáteis. No entanto, muitos usuários talvez não saibam que o WhatsApp oferece uma API poderosa que permite a integração da plataforma em uma variedade de aplicativos e serviços (NUVENS, 2018).

A API do WhatsApp é uma ferramenta versátil e valiosa para empresas que desejam aprimorar sua comunicação com os clientes. Ela permite a integração da plataforma de mensagens diretamente em aplicativos empresariais, possibilitando uma gama de recursos e benefícios. Por meio da API, as empresas podem enviar mensagens modelo para automatizar tarefas de rotina e melhorar a eficiência operacional. Além disso, *chatbots* personalizados podem ser desenvolvidos para interagir de forma natural com os clientes (ZENVIA, 2023).

Notificações de transações, suporte multimídia e grupos de conversa com vários participantes são recursos adicionais que tornam a API do WhatsApp uma ferramenta de comunicação completa. Ela oferece ainda relatórios e análises que auxiliam as empresas a avaliar o desempenho de suas campanhas e a eficácia das interações com os clientes (ZENVIA, 2023).

1.3 *Back-end*

O *back-end* é a parte de um sistema que não é visível ao usuário. Ele é responsável por lidar com as operações que acontecem "nos bastidores", como o armazenamento de dados, o processamento de informações e a comunicação com outros sistemas (EWALLY, 2021).

O *back-end* de uma biblioteca integrada via WhatsApp é responsável por armazenar os dados da biblioteca, processar as solicitações dos usuários e se comunicar com o aplicativo WhatsApp. Para isso, é necessário um banco de dados para armazenar informações sobre os livros, como título, autor, gênero, ano de publicação e disponibilidade. Também é necessário que *o back-end* seja capaz de entender as mensagens

enviadas pelos usuários e realizar as ações solicitadas. Por exemplo, se um usuário enviar uma mensagem perguntando sobre a disponibilidade de um livro, o *back-end* precisa verificar o banco de dados para encontrar a resposta. Por fim, é necessário que o *back-end* seja capaz de enviar e receber mensagens do WhatsApp. Isso permite que o usuário interaja com a biblioteca diretamente pelo aplicativo de mensagens. Com um *back-end* bem desenvolvido, é possível criar uma biblioteca integrada via WhatsApp que seja eficiente e fácil de usar. As ferramentas que mais se adequaram para o projeto, foram as tecnologias Java, *Springboot, hibernate, Docker* e *Kubernetes* (EWALLY, 2021).

1.3.1 JAVA

Java é uma linguagem de programação popular e versátil que pode ser usada para desenvolver uma variedade de sistemas, incluindo sistemas integrados ao WhatsApp. Para usar o Java em um sistema integrado ao WhatsApp, você precisará instalar o *Java Development Kit* (traduzindo para o português seria Kit de desenvolvimento java), criar um projeto Java e implementar as funcionalidades do sistema integrado ao WhatsApp, essas funcionalidades podem incluir o armazenamento de dados, o processamento de solicitações e a comunicação com o front-end. Para armazenar dados, você pode usar um banco de dados relacional, o *PostgreSQL* e *Flyway*. Para permitir que o front-end se comunique com o back-end, você pode usar um *API RESTful* (JAVA, 2001).

1.4 *Framework*

Um "*framework*" (ou estrutura, em português) é um conjunto de conceitos, práticas e ferramentas que fornece uma base para o desenvolvimento de software. Ele é uma espécie de esqueleto ou estrutura que pode ser expandida para criar aplicativos específicos (PRESSMAN, 2021).

Os *frameworks* são criados para facilitar o desenvolvimento de software, fornecendo funcionalidades genéricas que podem ser adaptadas para atender às necessidades específicas de um projeto. Eles são geralmente compostos por bibliotecas de código, padrões de design e convenções de desenvolvimento (PRESSMAN, 2021). Ajudam os desenvolvedores ao fornecerem uma estrutura predefinida para lidar com tarefas comuns, como interações com banco de dados, manipulação de dados, gerenciamento de sessões, entre outros. Eles aceleram o processo de desenvolvimento, promovem boas práticas de programação e ajudam na manutenção do código ao longo do tempo (PRESSMAN,2021).

Os *frameworks* fornecem uma estrutura organizada que permite que os desenvolvedores se concentrem mais nas características específicas do seu aplicativo, em vez de recriar repetidamente a infraestrutura básica, os frameworks levantados para o projeto foram *Hibernate JPA, Spring Boot* e para segurança o *Spring Security* (PRESSMAN, 2021).

1.4.2 *Hibernate JPA*

O *Hibernate JPA* é um *framework* de persistência de dados que permite aos desenvolvedores acessar dados de um banco de dados relacional usando uma API orientada a objetos. Ele fornece uma abstração do banco de dados, o que torna mais fácil para os desenvolvedores escrever o código, que é independente da implementação específica do banco de dados (COIMBRA, 2022).

1.4.3 *Spring boot*

Spring Boot é um framework de desenvolvimento de aplicativos Java que facilita a criação de aplicativos Java *stand-alone*, escaláveis e produtivos. Ele fornece uma variedade de recursos que podem ajudar a reduzir o tempo e o esforço necessários para desenvolver

aplicativos *Java. Spring Boot* é uma boa opção para desenvolver sistemas integrados ao WhatsApp. Ele fornece uma variedade de recursos que podem ajudar a reduzir o tempo e o esforço necessários para desenvolver aplicativos Java (JAVA, 2001)

1.4.4 *SpringSecurity*

Spring Security é um framework de segurança de código aberto para Java que pode ser usado para proteger aplicações web, mobile e de nuvem. Ele oferece uma variedade de recursos para autenticação, autorização, criptografia e monitoramento de segurança (CANDIOLLI, 2020).

1.4.5 *Next.JS*

Next.js é um *framework React* que permite criar aplicativos web e móveis com facilidade. Ele oferece uma variedade de recursos que podem ser úteis para desenvolver sistemas integrados ao WhatsApp (GEEKHUNTER, 2021).

1.4.6 *TailWind CSS*

Tailwind CSS é um *framework CSS* utilitário que permite aos desenvolvedores criar interfaces de usuário responsivas e escaláveis de forma rápida e fácil. Ele oferece uma ampla gama de classes CSS que podem ser usadas para estilizar elementos HTML (ROY,2022).

1.4.7 *UI React*

UI React é um *framework* de interface do usuário *React* que fornece uma variedade de componentes pré-construídos que podem ser usados para criar interfaces de usuário responsivas e escaláveis. Ele pode ser usado para criar sistemas integrados ao WhatsApp que são fáceis de usar e atraentes (UFSM, 2022).

1.4.8 *Kubernet*

Kubernetes é uma plataforma de orquestração de contêineres que permite aos desenvolvedores gerenciar e escalar aplicativos distribuídos. Ele pode ser usado para executar sistemas integrados ao WhatsApp em um ambiente de produção (REDHAT, 2023).

1.4.9 *Docker*

O *docker* é uma plataforma de virtualização de container que permite aos desenvolvedores empacotar aplicativos e suas dependências em unidades portáteis chamadas de containers. Os containers são executados em qualquer ambiente que tenha o *Docker* instalado, o que torna mais fácil desenvolver, implantar e gerenciar aplicativos (DANIELA, 2023).

1.5 *Front-end*

A parte de um sistema que é visível ao usuário é chamada de *front-end*. Ela é responsável por lidar com a interface do usuário, como o design, o layout e a interação.

Em uma biblioteca integrada via WhatsApp, o front-end é responsável por apresentar as informações da biblioteca ao usuário, receber as solicitações dos usuários e enviar mensagens para os usuários. Para integrar uma biblioteca via WhatsApp, o front-end precisa ser capaz de se comunicar com o *back-end*. Por exemplo, o front-end poderia apresentar uma lista de livros disponíveis, permitir que o usuário reserve um livro ou enviar uma notificação de evento. O *front-end* é uma parte essencial de qualquer sistema que interage com usuários. Ele é responsável por criar uma experiência de usuário agradável e eficiente. As ferramentas que mais se adequaram para a situação, foram *React*, *Next.JS*, *TailWind CSS* e *UI React* (TOTVS, 2021).

1.5.1 *React*

React é uma biblioteca de interface do usuário JavaScript que pode ser usada para criar interfaces de usuário responsivas e escaláveis. Ele pode ser usado para criar sistemas integrados ao WhatsApp que são fáceis de usar e

atraentes (DURÃES, 2021).

1.6 Banco de Dados

Um banco de dados é uma coleção estruturada de dados que é armazenada de forma organizada para que possa ser facilmente acessada e gerenciada. Ele é uma parte essencial de qualquer sistema de biblioteca integrado ao WhatsApp, pois é usado para armazenar dados sobre usuários, livros, empréstimos e outras informações. As ferramentas que mais se adequaram para a situação, foram o *Postgres* e o *flyway* (SOUZA; MOITAS; CUNHA,2018).

1.6.1 *Postgres*

O *PostgreSQL* é um banco de dados relacional de código aberto que é popular para uma variedade de aplicações, incluindo sistemas de biblioteca. Ele oferece uma ampla gama de recursos que podem ser úteis para sistemas de biblioteca integrados ao WhatsApp, incluindo recursos avançados, escalabilidade e segurança (REMESSA ONLINE, 2021).

1.6.2 *Flyway*

Flyway é uma ferramenta de gerenciamento de migrações de banco de dados de código aberto que pode ser usada para automatizar a aplicação de alterações no banco de dados. Ele é popular para sistemas de biblioteca integrados ao WhatsApp, pois pode ajudar a garantir que os dados do banco de dados estejam sempre atualizados e consistentes (VINICIUS, 2018).

1.7 Camada de segurança

A camada de segurança é uma parte essencial de qualquer sistema de biblioteca integrado ao WhatsApp. É responsável por proteger os dados dos usuários (informações pessoais, dados financeiros e empréstimos) e projetar uma camada de segurança para um sistema de biblioteca integrado ao WhatsApp, bem como as medidas específicas que podem ser implementadas para melhorar a segurança do sistema. Foi utilizado o *SpringSecurity* e *Bcrypt* (GOCACHE, 2017).

1.8 *Bcrypt*

O *Bcrypt* é um algoritmo de *hash* (um código criado a partir de um bloco de dados usando um algoritmo criptográfico) de senhas baseado no *Blowfish* (cifra simétrica de blocos que pode ser usado em substituição ao DES, algoritmo que possuía em torno de 19 anos de uso, e era vulnerável a ataques por força bruta devido ao tamanho de sua chave). É considerado um algoritmo seguro para armazenar senhas, pois é resistente a ataques de força bruta. Este trabalho utiliza o *Bcrypt* integrados ao WhatsApp para gerar e armazenar os hashes das senhas dos usuários do sistema (KERLYN, 2017).

1.9 Integração da biblioteca ao whatsapp

As aplicações de uma biblioteca são verdadeiramente diversas e dignas de destaque. Ela não é apenas um depósito de informações e recursos valiosos para pesquisa acadêmica, mas também atua como um espaço de convergência para a livre troca de ideias e o encontro de mentes inquisitivas. Nas bibliotecas modernas, a convergência entre o analógico e o digital é notável, com a incorporação de tecnologia que abre novos horizontes. Elas disponibilizam não apenas obras impressas, mas também oferecem acesso a vastos recursos digitais, permitindo uma pesquisa ampla e aprofundada, além de disponibilizarem espaços dedicados ao trabalho colaborativo, promovendo a cocriação e a inovação (WEINBERGER,2007).

No contexto em constante evolução das bibliotecas digitais e sistemas de gerenciamento de livros, a busca por recursos que simplifiquem a vida dos leitores e proporcionem uma experiência de empréstimo de livros mais eficaz se torna uma prioridade imperativa. Um

espaço virtual, um santuário de conhecimento que disponibiliza uma vasta coleção de obras e otimiza a pesquisa por livros específicos por meio de filtros avançados. Este cenário virtual se estende ao âmbito da comunicação instantânea e personalizada em que o aplicativo *WhatsApp* desempenha um papel fundamental, pois os usuários são notificados sobre a disponibilidade do livro que desejam e a iminência do vencimento do prazo de devolução de uma obra já em suas mãos (WEINBERGER, 2007).

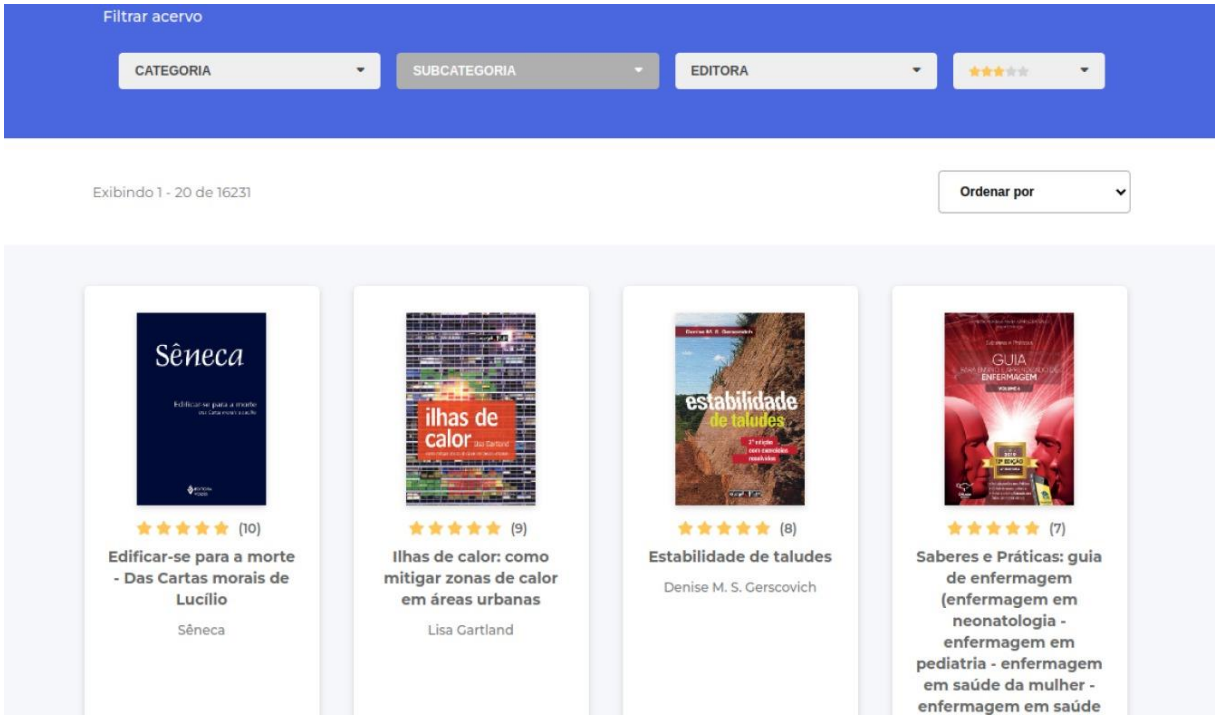
2 OBJETIVOS

O objetivo deste artigo consiste na elaboração de uma plataforma de gestão bibliotecária que se integra de forma coesa ao aplicativo *WhatsApp*. Pretende-se conduzir uma análise detalhada acerca das diversas funcionalidades inerentes a essa aplicação, avaliar seu impacto positivo na acessibilidade da biblioteca, examinar a dinâmica de interação entre os usuários e os bibliotecários, e elucidar de que maneira essa inovação contribui de forma substancial para a preservação da relevância das bibliotecas na era digital. Por meio deste estudo, almeja-se ressaltar de que modo essa iniciativa inovadora representa um avanço de magnitude expressiva no âmbito das bibliotecas, ao efetuar uma conexão entre a vastidão do conhecimento contido em suportes impressos e as ferramentas tecnológicas contemporâneas.

3 MATERIAL E MÉTODOS

Inicialmente, desenvolveu-se um componente *index* com as ferramentas *react* e *tailwind* para representar a tela que mostra todos os livros. Essas ferramentas foram escolhidas devido à facilidade do *react* e a ampla biblioteca de CSS que o *tailwind* fornece (Figura 1).

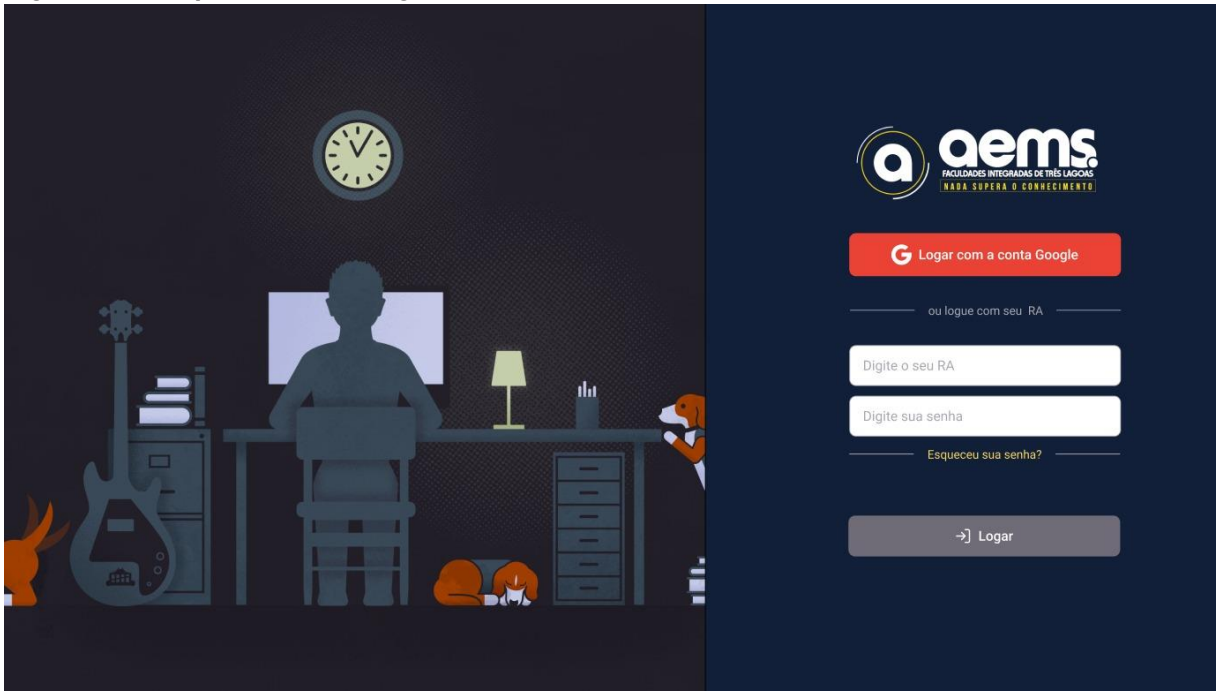
Figura 1. Página *Index*, um exemplo da página principal.



Fonte: Elaborado pelos autores.

Usou-se a mesma estrutura da tela index para desenvolver a tela de *login*. Para tanto, alterou-se o background e os tons de corres e gerou-se uma alteração modelada da tela base (Figura 2).

Figura 2. Exemplo de tela de login.



Fonte: Elaborado pelos autores.

Figura 3. Exemplo do código em *React* – sessão do usuário.

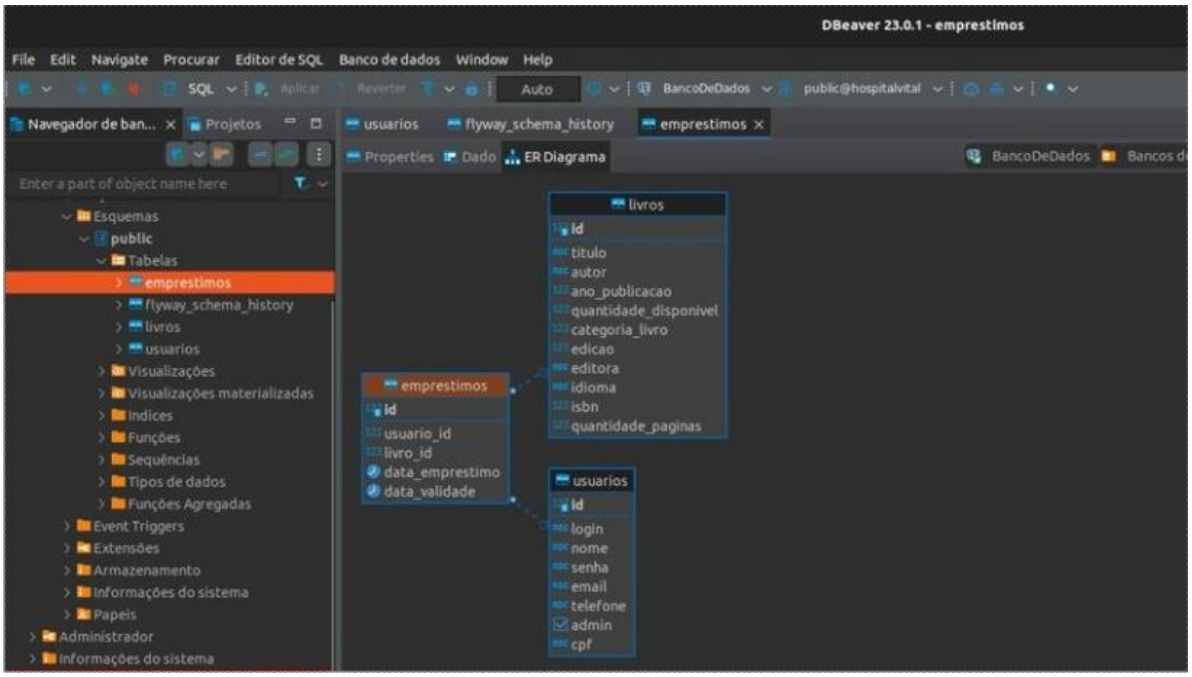
```
src > services > auth > session.tsx > useSession > React.useEffect() callback > then() callback
'use client';

import React from 'react';
import { authService } from './authService';
import { useRouter } from 'next/navigation';

export function useSession() {
  const [session, setSession] = React.useState(null);
  const [loading, setLoading] = React.useState(true);
  const [error, setError] = React.useState(null);

  React.useEffect(() => {
    authService.getSession()
      .then((userSession) => {
        console.log(userSession);
        setSession(userSession);
      })
      .catch((err) => {
        setError(err);
      })
      .finally(() => {
        setLoading(false);
      });
  }, []);

  return {
    data: {
      session,
    },
    error,
    loading,
  }
}
```

Fonte: Elaborado pelos autores.

A seguir, criou-se da sessão do usuário que informa se o mesmo está ativo ou quando vai desconectar, como mostra o código na Figura 3.

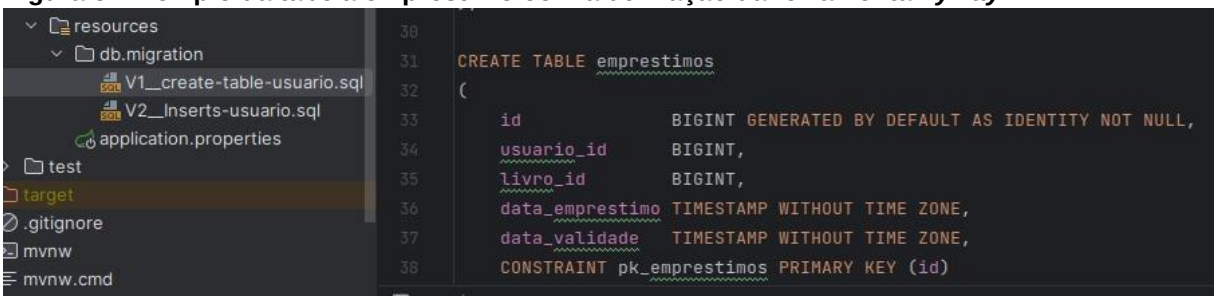
A próxima etapa foi a criação do banco de dados em três tabelas com a ferramenta *postgres*. A primeira foi para conter as informações pessoais como, a identificação (id), nome, login de usuário, e-mail e se essa pessoa tem permissão de administrador. A outra tabela (livro), a id do livro, o título, autor e produtora. Já a última tabela, a de empréstimos, na qual as anteriores se relacionam com ela. A Figura 4 mostra que a tabela empréstimo vincula as informações de data de retirada e de validade do empréstimo com as ids das tabelas usuários e livros. Deste modo, gera o dado do usuário x com a data da retirada do livro e validade da entrega.

Figura 4. Exemplo do fluxograma do banco.



Fonte: Elaborado pelos autores.

Figura 5. Exemplo da tabela empréstimo com a utilização da ferramenta *flyway*.



Fonte: Elaborado pelos autores.

A seguir, utilizou-se o *Flyway* para permitir o versionamento e gerenciamento do banco de dados para se poder controlar a evolução dos elementos que compõem uma determinada base de dados. A Figura 5 mostra o exemplo da tabela empréstimos.

Criada a tabela, utilizou-se a ferramenta *Springsecurity* para realizar a proteção das rotas do sistema, para que seja garantido o acesso autorizado somente para o nível designado ao usuário, como por exemplo não dar permissões de administrador para um usuário comum, representado na Figura 6.

Figura 6. Código para assegurar o acesso do usuário.

```
@Configuration
@EnableWebSecurity
public class SecurityConfigurations {
    //Ao criar essa configuração ele desabilita o formulário de login e senha para poder fazer requisições da aplicação
    //Adicionar a seguinte anotação caso na classe de segurança caso eu queira fazer a checagem de segurança como o shiro faz
    //Encima dos metodos de requisição eu adiciona a role necessaria @Secured("ROLE_ADMIN")
    //@EnableMethodSecurity(securedEnabled = true)

    @Autowired
    private SecurityFilter securityFilter;
     Breno
    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        //CSRF Cross-Site Request Forgery
        //sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS) é para indicar que eu estou usando autenticação Stateless
        //Exemplo para bloquear requisição caso o usuário não tenha permissão
        return http.csrf().disable() HttpSecurity
                .cors().and()
                .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS) SessionManagementConfigurer<HttpSecurity>
                .and().authorizeHttpRequests() AuthorizationManagerRequestMat...
                .requestMatchers(HttpMethod.POST, "/login").permitAll()
                .requestMatchers("/v3/api-docs/**", "/swagger-ui.html", "/swagger-ui/**").permitAll()
                .anyRequest().authenticated()
                .and().addFilterBefore(securityFilter, UsernamePasswordAuthenticationFilter.class) HttpSecurity
                .build();
    }

     Breno
    @Bean
    public AuthenticationManager authenticationManager(AuthenticationConfiguration configuration) throws Exception {
        return configuration.getAuthenticationManager();
    }
```

Fonte: Elaborado pelos autores.

Figura 7. Exemplo de código da classe *bean* para coleta de dados.

```
package lhb.api.domain.usuario;

import ...

/**
 * @author Breno
 */
32 usages  Breno
@Table(name = "usuarios")
@Entity(name = "Usuario")
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
@EqualsAndHashCode(of = "id")
public class Usuario implements UserDetails {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String login;
    private String nome;
    private String senha;
    private String email;
    private String telefone;
    private boolean admin;
    private String cpf;

     Breno
    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
        //Caso o usuario tenha controle de perfis ou grupos de acesso é aqui que você configura essa gracinha
        if(admin){
            return List.of(new SimpleGrantedAuthority( role: "ROLE_USER"));
        }
        return List.of(new SimpleGrantedAuthority( role: "ROLE_USER"));
    }
```

Fonte: Elaborado pelos autores.

Após a criação do banco, desenvolveu-se a classe *bean*, local de coleta de dados de cada variável e se ela vai ter permissão de administrador, para enviar para o banco de dados (Figura 7). Para tanto, utilizou-se a ferramenta *bycript* para criptografar as senhas dentro do banco (Figura 8).

Posteriormente, utilizou-se as ferramentas *bycript* e *springsecurity* para desenvolver um *token* para o usuário e validar se a senha está correta (Figura 9).

Figura 8. Exemplo de *bycript* utilizado para criptografia.

```
@Bean
public PasswordEncoder passwordEncoder(){
    return new BCryptPasswordEncoder();
}
```

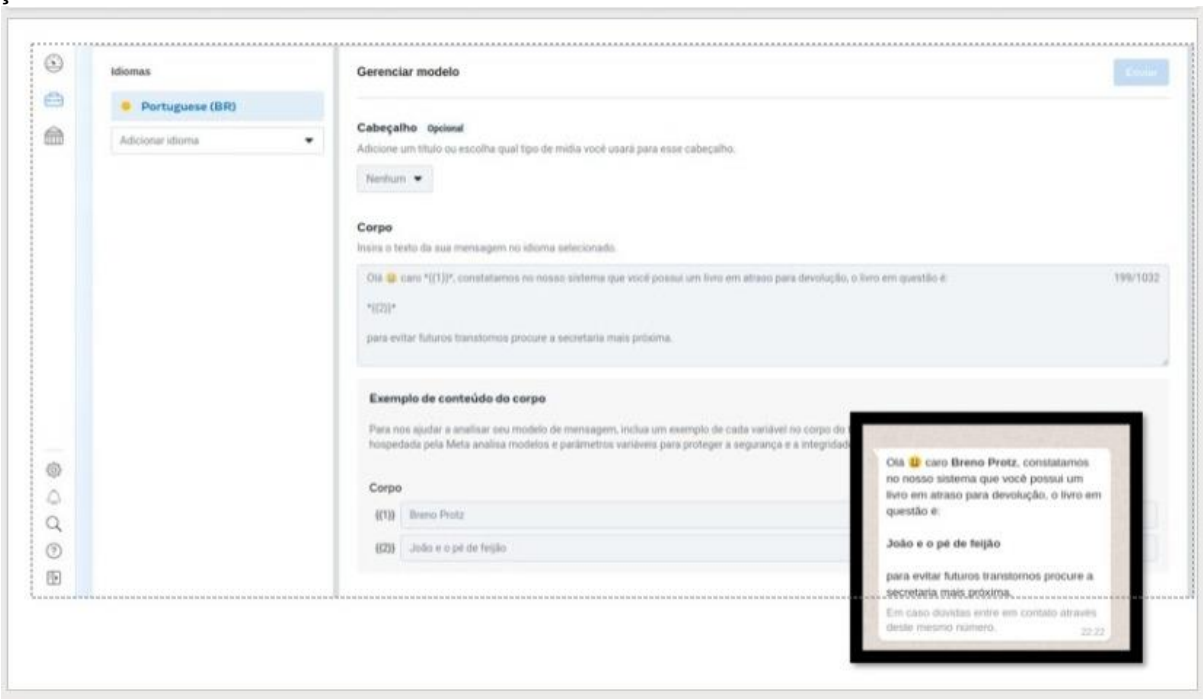
Fonte: Elaborado pelos autores.

Figura 9. Código de geração do token da sessão do usuário.

```
@PostMapping
public ResponseEntity efetuarLogin(@RequestBody @Valid DadosAutenticacao dados){
    var authenticationToken = new UsernamePasswordAuthenticationToken(dados.login(), dados.senha());
    var authentication = manager.authenticate(authenticationToken);

    var tokenJWT = tokenService.gerarToken((Usuario) authentication.getPrincipal());
    return ResponseEntity.ok(new DadosTokenJWT(tokenJWT, ((Usuario) authentication.getPrincipal()).getId(), ((Usuario) authentication.getPrincipal()).getNome()));
}
```

Fonte: Elaborado pelos autores.

Figura 10. Mensagem recebida no aplicativo Whatsapp sobre a proximidade da data de devolução.



Fonte: Elaborado pelos autores.

4 RESULTADOS E DISCUSSÕES

A ampliação da disponibilidade de reservas e consultas online representou um avanço significativo na acessibilidade aos serviços oferecidos pela biblioteca. A capacidade dos usuários de realizar transações remotas eliminou obstáculos físicos e conferiu uma maior conveniência às operações, refletindo diretamente na satisfação e no envolvimento do público. A flexibilidade proporcionada pela capacidade de adicionar e remover livros contribuiu para uma administração dinâmica e eficiente do acervo.

Além disso, as notificações automatizadas desempenharam um papel crucial ao facilitar uma comunicação eficaz com os usuários. A entrega oportuna de mensagens relativas ao status das reservas e alertas de expiração promoveu uma interação mais proativa e garantiu que os usuários estivessem bem-informados. Essa abordagem fortaleceu os laços entre a biblioteca e sua comunidade de usuários, como exemplificado pelo caso em que a aplicação enviava uma mensagem para o usuário, informando sobre a iminente expiração do prazo de devolução do livro (Figura 10). A prontidão e proatividade na comunicação contribuíram para uma relação mais estreita e eficiente entre a biblioteca e seus usuários.

5 CONCLUSÕES

O processo de desenvolvimento do sistema integrado de biblioteca resultou em uma série de conquistas notáveis, evidenciando seu impacto extremamente positivo tanto na dinâmica operacional quanto na experiência dos usuários. A incorporação de novas aquisições desempenhou um papel fundamental no enriquecimento da oferta bibliográfica, ao passo que a eliminação de obras desatualizadas assegurou a pertinência e a qualidade do acervo, culminando em um ambiente bibliotecário mais alinhado às crescentes exigências do público.

No âmbito da conveniência para os usuários, é possível observar que as mensagens automáticas provenientes da biblioteca exercem uma influência multifacetada. Elas se revelam instrumentais ao manter os usuários devidamente informados acerca dos serviços disponíveis na biblioteca. A título exemplificativo, as bibliotecas podem empregar mensagens automáticas para lembrar os usuários sobre a necessidade de devolver livros emprestados, renovar empréstimos ou participar de eventos promovidos pela instituição. Tal abordagem visa não apenas evitar possíveis multas e atrasos, mas também proporcionar aos usuários a oportunidade de explorar plenamente as atividades e recursos oferecidos pela biblioteca. Este método de comunicação automatizada, ao fornecer informações relevantes de forma oportuna, contribui significativamente para otimizar a interação entre os usuários e a biblioteca, aprimorando assim a experiência geral dos frequentadores.

REFERÊNCIAS

CANDIOLLI, S. Começando com Spring Security. Disponível em: <https://cwi.com.br/blog/comecando-com-spring-security/>. Acesso em: 25 maio 2023.

CORTEZ, M. G. C.; SILVA, M. F.; SILVA, J. R. Sistema de descoberta e entrega de livros em bibliotecas universitárias. Disponível em: <https://repositorio.ufrn.br/bitstream/123456789/33413/1/Sistemadescobertaentrega_Cortez_2021.pdf>. Acesso em: 30 nov. 2023.

CUNHA, J; MOITAS, T; SILVA, G. Programação de banco de dados. 1 ed. São Paulo: 2018.

DANIELA, D. O que é Docker e como ele

funciona? Disponível em: <https://www.hostinger.pt/tutoriais/o-que-e-docker>. Acesso em: 05 mar. de 2023.

DURÃES, G. Veja o que é React e como ele pode facilitar a sua vida de programador! Disponível em: <https://blog.tecnospeed.com.br/o-que-e-react/>. Acesso em: 17 set. 2023.

GEEKHUNTER. O que é Next.js. Disponível em: <https://blog.geekhunter.com.br/o-que-e-next-js/>. Acesso em: 30 out. 2022.

GOCACHE. Melhores práticas de segurança para APIs. Disponível em: <https://www.gocache.com.br/seguranca/melhores-praticas-de-seguranca-para-apis/>. Acesso em 12 jan. 2023

JAVA. What is java 2001. Disponível em: <https://www.java.com/en/download/help/whatis_java.html>. Acesso em: 25 fev. 2023.

KERLYN. Uma breve introdução sobre bcrypt. Disponível em: <https://medium.com/reprogramabr/uma-breve-introdu%C3%A7%C3%A3o-sobre-bcrypt-f2fad91a7420>. Acesso em: 17 jan. 2023.

PACHER, T.; NEVES, E. Análise e comparação de frameworks de persistência. Disponível em: <https://riut.utfpr.edu.br/jspui/bitstream/1/16876/4/PG_COADS_2012_1_05.pdf>. Acesso em: 04 jan. 2023.

PRESSMAN, R. Engenharia de Software: Uma Abordagem Profissional. 9 ed. São Paulo: McGraw-Hill Education, 2022.

REDHAT. Wha tis Kubernetes?. Disponivel em: <https://www.redhat.com/en/topics/containers/what-is-kubernetes>. Acesso em: 20 fev. 2023.

REMESSA ONLINE. Tutorial PostgreSQL: Entenda como funciona este gerenciador. Disponível em: <https://www.remessaonline.com.br/blog/tutorial-postgresql-entenda-como-funciona-este-gerenciador/>. Acesso em: 14 fev. 2023.

ROY, S. O que é Taiwind CSS?. Disponível em: <https://www.freecodecamp.org/portuguese/news/o-que-e-tailwind-css-um-guia-para-iniciantes/>. Acesso em: 16 jul. 2023.

SILVA, M. P. S.; SILVA, R. B.; ALMEIDA, J. P. C. Avaliação de sistemas de recomendação: uma revisão sistemática. Revista Digital de Biblioteconomia e Ciência da Informação (RDBI). Disponível em: <https://periodicos.sbu.unicamp.br/ojs/index.php/rdbci/article/download/8634630/3397/5079>. Acesso em: 30 maio 2023.

SOMMERVILE, I. Engenharia de Software. 9 ed. São Paulo: Universidade de Pearson 2011.

TOTVS. *Front end*: O que é, como funciona e qual a importância. Disponivel em: <https://www.totvs.com/blog/developers/front-end/>. Acesso em: 03 ago. 2023.

UFSM. Framework React UI. Disponível em: <https://www.ufsm.br/pet/sistemas-de-informacao/2022/08/14/material-ui-framework-react-ui>. Acesso em: 02 fev. 2023.

VINICIUS, C. Versionamento de banco dados com Flyway. Disponível em: <https://blog.cvinicius.com.br/2018/02/versionamento-de-banco-dados-com-flyway.html>. Acesso em: 16 mar. de 2023.