TRABALHO COMPLETO
FULL PAPER
XXVII SEPE
Simpósio de Ensino, Pesquisa e Extensão
Teaching, Research and Extension Symposium
24 A 26 DE OUTUBRO 2023
OCTOBER 24th to 26th, 2023
ISSN: 2316-9745
CIÊNCIA E DESENVOLVIMENTO SUSTENTÁVEL
SCIENCE AND SUSTAINABLE DEVELOPMENT
UFN Universidade Franciscana Franciscan University
https://doi.org/10.48195/sepe2023.27196

CHECKCHAP: PLATAFORMA INTUITIVA E ÁGIL PARA CRIAÇÃO E COMPARTILHAMENTO DE LISTA DE TAREFAS ONLINE

Bernardo de Souza Viero[1]; Leonardo Peripolli Pereira[2]; Mauricio Pereira Braga[3]; Herysson Rodrigues Figueiredo[4]

Universidade Franciscana (UFN)

Santa Maria – RS

bernardo.viero@ufn.edu.br, leonardo.ppereira@ufn.edu.br,

mauricio.pereira@ufn.edu.br, herysson.figueiredo@ufn.edu.br

RESUMO

Este artigo apresenta o desenvolvimento sobre a criação da aplicação CheckChap, uma plataforma online dedicada à gestão de listas de tarefas. Além de elucidar conceitos fundamentais do desenvolvimento web, o texto detalha a metodologia empregada no desenvolvimento desta aplicação.

Palavras-chave: Plataforma online; Desenvolvimento de software; Compartilhamento de tarefas; Acessibilidade; Padrão de projeto MVC

ABSTRACT

This article aims to introduce a concise overview of the development of the CheckChap application, an online platform dedicated to task list management. In addition to elucidating fundamental concepts of web development, the text outlines the methodology employed in the development of this application. We conclude with some final considerations about the project.

Keywords: Online platform; Software development; Task sharing; Usability; MVC design pattern.

Eixo Temático: Tecnologia, Inovação e Desenvolvimento Sustentável (TIDS).

1. INTRODUÇÃO

Este artigo descreve o desenvolvimento do CheckChap, uma plataforma online que simplifica a gestão de tarefas em ambientes profissionais e acadêmicos. A

[1] Autor/Apresentador - Instituição do(s) autor(es) e-mail(s).
[2] Demais Autores - Instituição do(s) autor(es) e-mail(s).

plataforma permite a criação e compartilhamento de listas de tarefas de forma simples e intuitiva, sem a necessidade de cadastro. Uma característica distintiva é a capacidade de os usuários criarem URLs exclusivas para compartilhamento e colaboração em diferentes contextos.

Uma URL, Uniform Resource Locator (localizador padrão de recursos), desempenha um papel crucial no funcionamento do CheckChap. Quando os usuários criam listas de tarefas ou as compartilham, o HTTP (Hypertext Transfer Protocol) entra em cena. O HTTP é o protocolo que possibilita a comunicação entre o dispositivo do usuário e os servidores da web onde o CheckChap está hospedado. A URL atua como um endereço único que direciona o navegador da web para a página específica onde as tarefas estão armazenadas (Osório, 2020). Isso torna o compartilhamento e a colaboração em tarefas mais simples e acessíveis.

O desenvolvimento do CheckChap foi motivado pela necessidade da equipe de robótica da Universidade Franciscana de superar desafios na gestão de peças e na observância das normas de segurança durante a construção de robôs de combate. O CheckChap foi projetado para abordar esses problemas, permitindo o registro em tempo real das necessidades da equipe e promovendo uma comunicação eficiente e organização eficaz. Embora tenha sido inicialmente desenvolvido para atender às necessidades da equipe de robótica da Universidade Franciscana, o CheckChap tem o potencial de beneficiar um público mais amplo.

O projeto tem como objetivo geral otimizar a gestão de tarefas em diversos contextos por meio da criação da plataforma online CheckChap, que se destaca por sua interface intuitiva e acessibilidade. A plataforma foi desenvolvida para aprimorar a gestão de tarefas em ambientes pessoais, profissionais e acadêmicos, eliminando a necessidade de cadastro e permitindo um compartilhamento eficiente de listas de tarefas. Para alcançar esse objetivo principal, foram estabelecidos objetivos específicos, incluindo o desenvolvimento de uma interface de usuário simples, a criação de URLs exclusivas para compartilhamento e a seleção de tecnologias confiáveis para garantir a confiabilidade e a segurança da plataforma.

2. METODOLOGIA

As metodologias ágeis têm como objetivo reduzir os riscos associados à imprevisibilidade e incertezas dos projetos, abordando contratempos de forma mais

fluída e acelerada. Elas favorecem uma abordagem flexível e adaptativa no desenvolvimento de software (Prikladnicki, 2014).

Neste projeto, foi adotada a metodologia de desenvolvimento XP (Extreme Programming) para aproveitar os benefícios das abordagens ágeis. O XP é conhecido por enfatizar a comunicação eficaz, colaboração intensa e feedback contínuo. Ele se baseia em práticas como programação em pares, testes automatizados, integração contínua e refatoração para garantir a qualidade do software em todas as etapas do desenvolvimento (Beck, 2004).

No XP, a equipe trabalha em iterações curtas, entregando incrementos funcionais do software em intervalos regulares. Através da programação em pares, onde dois programadores trabalham juntos em uma tarefa, promove-se a troca de conhecimento e a revisão de código em tempo real, resultando em soluções mais robustas e com menos erros (Beck, 2004).

2.1 Ferramentas utilizadas

Para a gestão de dados, escolheu-se o MySQL, que é uma escolha sólida de SGBDR para desenvolvedores e organizações que precisam armazenar, recuperar e gerenciar dados de maneira eficaz e confiável. Sua natureza de código aberto, desempenho, segurança e escalabilidade o tornam uma opção popular para uma ampla gama de aplicações. Ele é responsável por armazenar todas as informações relacionadas às tarefas e aos usuários, garantindo a segurança e a consistência dos dados (“MySQL Oracle”, 2023).

Durante a construção do projeto, utilizamos o framework Java Springboot, é um framework de desenvolvimento de aplicativos Java que simplifica a criação de aplicativos robustos e escaláveis. Ele oferece um conjunto de ferramentas e convenções que aceleram o processo de desenvolvimento, permitindo que os desenvolvedores se concentrem nas funcionalidades do aplicativo, em vez de configurações complexas. O Spring Boot integra facilmente com outras tecnologias do ecossistema Spring e é amplamente adotado na criação de aplicativos web, microserviços e aplicativos empresariais. Isso facilitou a implementação de diversas funcionalidades e a integração eficaz com outros componentes do sistema (“Springboot”, 2023).

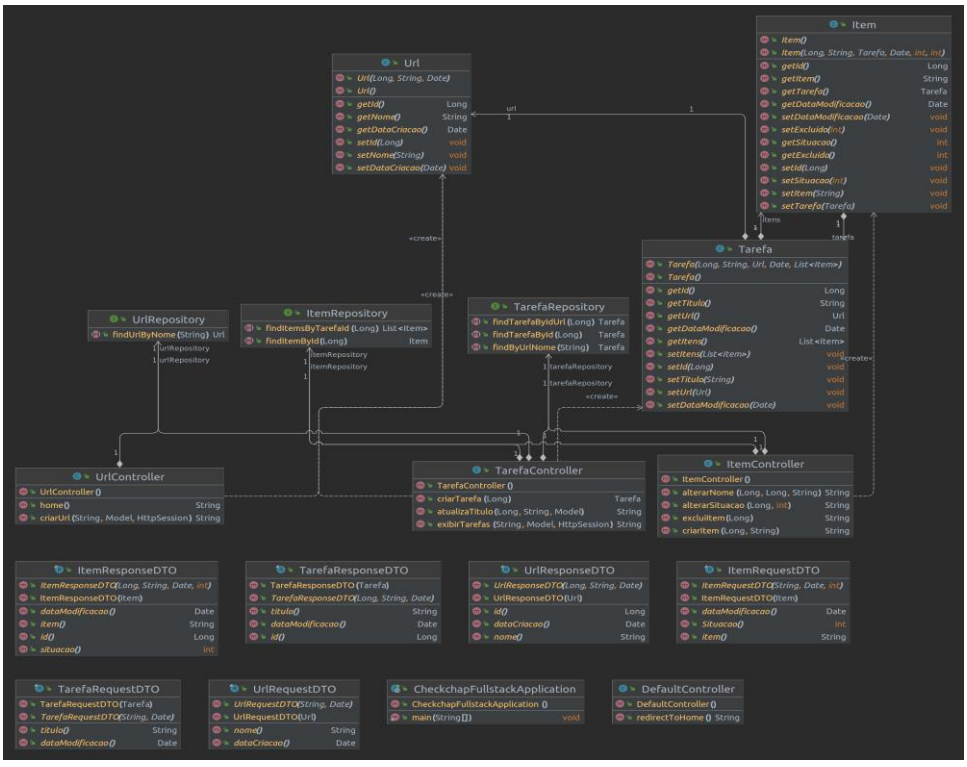
ISSN: 2316-9745

No que se refere ao frontend, utilizou-se uma combinação de Thymeleaf, JavaScript e TailwindCSS. O Thymeleaf, um motor de templates Java, permitiu a criação de páginas web dinâmicas. O JavaScript foi aplicado para aumentar a interatividade na plataforma, tornando a experiência do usuário mais envolvente. Quanto à estilização, o TailwindCSS foi escolhido devido à sua flexibilidade e eficiência na criação de interfaces de usuário modernas e responsivas, contribuindo para um design atraente e adaptado a diferentes dispositivos (“TailwindCSS”, 2023).

2.2 Planejar funcionalidades

Neste projeto, o planejamento das funcionalidades esteve diretamente relacionado com o diagrama de caso de uso e, consequentemente, com o diagrama de classes, visualizado na Figura 1. A partir desses diagramas, foi possível compreender as principais funcionalidades do sistema.

Figura 1. Diagrama de Classes da aplicação já desenvolvida.



Viero, Bernardo. 2023.

Isso tornou as funcionalidades mais compreensíveis e fáceis de serem desenvolvidas. Cada regra de negócio ou funcionalidade foi associada ao padrão MVC, com cada parte do sistema dividida em seu controlador específico. Isso facilitou

a construção da aplicação e a futura manutenção, tornando cada funcionalidade mais compreensível.

O uso do padrão MVC (Model-View-Controller) foi fundamental para tornar as funcionalidades da aplicação mais compreensíveis e mais fáceis de serem desenvolvidas. O padrão MVC é uma abordagem de arquitetura de software que divide uma aplicação em três componentes interconectados: Model (Modelo), View (Visualização) e Controller (Controlador). Cada um desses componentes tem um papel específico na organização e operação da aplicação (“Spring MVC”, 2023).

O Model representa a lógica de negócios e os dados da aplicação. Ele lida com as regras de negócios e o acesso aos dados subjacentes. A View, por outro lado, cuida da apresentação dos dados ao usuário, incluindo a interface do usuário e a forma como as informações são exibidas. Enquanto isso, o Controller age como intermediário entre o Modelo e a Visualização. Ele controla o fluxo de dados, gerencia as interações do usuário e coordena as ações apropriadas no Modelo e na Visualização (“Spring MVC”, 2023).

Dessa forma, conseguimos planejar cada funcionalidade com base nos requisitos funcionais da aplicação, dividindo-as em três partes principais para a execução das tarefas. Cada funcionalidade estava alocada em seu respectivo controlador, e o armazenamento e manipulação dos dados eram realizados por meio dos modelos correspondentes.

Isso permitiu que as solicitações para a manipulação e execução das tarefas e funcionalidades fossem feitas por meio das telas de visualização dos templates.

2.3 Desenvolvimento coletivo

Neste projeto, adaptamos a prática tradicional de Programação em Duplas (Pair Programming) de maneira inovadora para atender às necessidades do desenvolvimento da plataforma. A Pair Programming envolve, normalmente, dois programadores trabalhando juntos no mesmo código, no mesmo computador, alternando entre os papéis de "piloto" e "navegador". No entanto, nesta situação, a equipe modificou essa abordagem para funcionar em um ambiente de desenvolvimento distribuído (Williams, 2002).

Cada desenvolvedor tinha seu próprio repositório de código e trabalhava em sua própria máquina, mas a colaboração e a revisão de código ainda ocorriam em

tempo real. Isso significava que, embora cada desenvolvedor estivesse trabalhando fisicamente de forma independente, ainda era revisado e corrigido o código um do outro, mantendo a essência da Programação em Duplas.

Essa adaptação para um ambiente distribuído trouxe várias vantagens. Primeiramente, permitiu que a equipe aproveitasse os benefícios da Programação em Duplas, como a redução de erros de código e maior eficiência, mesmo trabalhando em locais diferentes. Além disso, permitiu a cada programador manter um foco individual, enquanto ainda se beneficiava da revisão e do feedback do seu parceiro.

Em relação ao desenvolvimento da interface, foi adotado uma estratégia diferenciada, usando um submódulo separado para o desenvolvimento de interfaces. Esse submódulo permitiu que os desenvolvedores se concentrassem exclusivamente na interface do usuário antes de integrar suas alterações ao repositório principal do código. Essa prática fortaleceu o fluxo de trabalho da programação em duplas e ajudou a manter o foco na experiência do usuário durante o desenvolvimento do projeto.

Essa prática é um excelente exemplo de como a modularização do código pode melhorar a eficiência e a eficácia do desenvolvimento de software, permitindo que diferentes aspectos de um projeto sejam tratados de forma independente e, posteriormente, integrados em uma solução completa.

2.4 Arquitetura e desenvolvimento de funcionalidades

A arquitetura e o desenvolvimento da aplicação foram baseados nos diagramas construídos, resultando em uma aplicação estruturada com padrões MVC, visualizado na Figura 2.

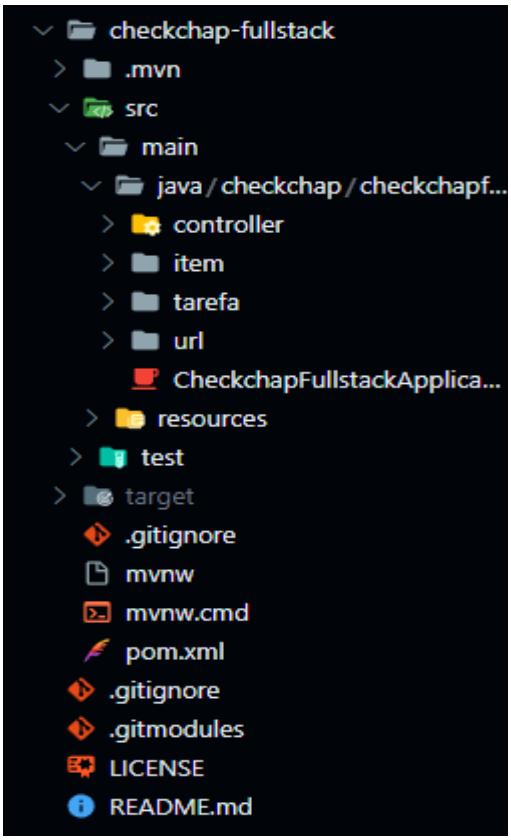
Model: O modelo representa a camada de dados da aplicação, contendo classes que representam as entidades de negócio, regras de negócio e acesso aos dados. Exemplos incluem as classes "Url" e "UrlRepository".

View: A visão é responsável por exibir informações ao usuário e interagir com ele. Geralmente é representada por arquivos HTML retornados pelos métodos do controlador, como o arquivo "index" retornado pelo método "home()".

Controller: O controlador recebe as requisições dos usuários, coordena a interação entre o modelo e a visão, e executa a lógica de negócio. A classe

UrlController lida com as requisições relacionadas à criação e redirecionamento de URLs, conectando o controlador com os modelos para manipulação de dados.

Figura 2. Arquitetura MVC do projeto.



Viero, Bernardo. 2023.

2.5 Usabilidade, experiência do usuário e acessibilidade

No desenvolvimento da aplicação, a prioridade foi assegurar a usabilidade e proporcionar uma experiência agradável aos usuários. Foi buscada a criação de uma interface simplificada e intuitiva para a criação e compartilhamento de listas de tarefas, levando em consideração a acessibilidade, garantindo que pessoas com diferentes habilidades e limitações pudessem utilizá-la facilmente.

Durante o processo de desenvolvimento, foram adotadas técnicas de acessibilidade e seguidas diretrizes reconhecidas de usabilidade. Uma das abordagens fundamentais foi a aplicação das 10 heurísticas de Nielsen, que auxiliaram no design e na arquitetura do site, permitindo aos usuários encontrar facilmente informações relevantes sobre as tarefas, independentemente de suas habilidades ou limitações (Nielsen, 2013).

Para melhorar a compreensão e o uso da plataforma, foram usados textos descritivos em botões e links, conforme recomendado pelas heurísticas de Nielsen. Isso facilitou a interação dos usuários com a plataforma ao estabelecer uma correspondência entre o sistema e o mundo real.

Além disso, foram seguidas as diretrizes do Web *Content Accessibility Guidelines* (Diretrizes de acessibilidade de conteúdo) para garantir a acessibilidade. Foram implementados recursos como descrições alternativas (alt text) para todas as imagens do site, permitindo que usuários com deficiência visual entendessem o conteúdo por meio de leitores de tela. Também foi dada atenção ao contraste adequado entre o texto e o fundo, assegurando que as informações fossem apresentadas de forma clara e legível para usuários com deficiência visual (Nielsen, 2013).

3. RESULTADOS E DISCUSSÕES

Como resultado de um trabalho dedicado, foi desenvolvida uma plataforma web para a criação de listas de tarefas, que oferece URLs personalizáveis e a opção de compartilhamento (Viero, 2023).

A página inicial, apresenta um formulário para que o usuário crie uma URL personalizada para sua página de tarefas, visualizado na Figura 3. Além disso, fornece instruções claras e detalhadas sobre como utilizar a plataforma, tornando a experiência do usuário mais fácil.

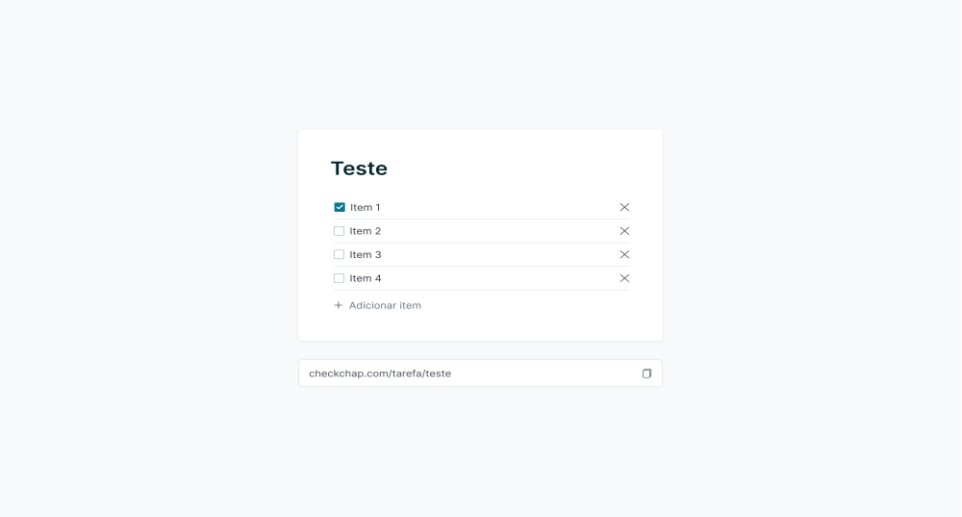
Figura 3. Tela inicial da plataforma.



Viero, Bernardo. 2023.

A Figura 4 apresenta a página de criação de listas de tarefas apresenta campos para atribuição de título e adição de tarefas, cada uma com um checkbox correspondente. A página também exibe a URL personalizada para compartilhamento, também acessível diretamente pelo campo de busca navegador, contribuindo para uma gestão eficiente das tarefas.

Figura 4. Tela de tarefas criada



Viero, Bernardo. 2023.

4. CONCLUSÃO

A gestão eficaz de tarefas é fundamental tanto em ambientes profissionais quanto acadêmicos, sendo a sua execução eficiente um fator determinante para o sucesso de projetos. Para atender a essa necessidade, desenvolvemos a plataforma CheckChap, uma ferramenta online que permite a criação e o compartilhamento de listas de tarefas de forma simplificada e intuitiva.

Essa solução mostrou-se eficaz e versátil, uma vez que, embora tenha sido desenvolvida com base na experiência de uma equipe de robótica universitária, revelou-se aplicável a uma variedade de contextos, sejam eles pessoais, profissionais ou acadêmicos. O recurso de criação de URLs exclusivas para cada lista de tarefas no CheckChap demonstrou ser especialmente útil, facilitando a colaboração e o compartilhamento de tarefas entre os usuários.

REFERÊNCIAS

Beck, Kent. Extreme Programming Explained: Embrace Change. Addison-Wesley Professional; 2ª ed. 2004.

Nielsen, Jakob. Usabilidade Móvel. GEN LTC. 27 de novembro de 2013.

MySQL Oracle. MySQL HeatWave Database Service. Em: 2023. Disponível em: https://www.oracle.com/br/mysql/. Acessado em: 15 de set. 2023.

Osório, Victor. Anatomia do Protocolo HTTP: A internet se faz de protocolos e não de frameworks. eBook Kindle. Edição 2014.

Prikladnicki, Rafael. Métodos Ágeis para Desenvolvimento de Software. Bookman; eBook Kindle. Edição 2014.

SPRING MVC. Serving Web Content with Spring MVC. Em: (2023). Disponível em https://spring.io/guides/gs/serving-web-content. Acessado em 15 de set. 2023.

Spring Boot. Spring Boot 3.1.3. Em: 2023. Disponível em: https://spring.io/projects/spring-boot. Acessado em: 15 de set. 2023.

TailwindCSS. Rapidly build modern websites without ever leaving your HTML. Em: (2023). Disponível em https://tailwindcss.com/. Acessado em: 18 de jun. de 2023.

Viero, Bernardo. Checkchap-springboot. Checkchap. Em: (2023). Disponível em https://github.com/bernardoviero/checkchap-springboot. Acessado em 20 de jun. 2023.

Williams, Laurie Pair Programming Illuminated. Addison-Wesley Professional; 1st Revised ed. 2002.

W3ORG. Shawn Lawton Henry (WCAG). "WCAG 2 Overview". Em: (1994). Disponível em https://www.w3.org/WAI/standards-guidelines/wcag. Acessado em 18 de jun. 2023.