Universidade Tiradentes
Ciências da Computação

Pedro Faria Barreto
Alan Rocha Barbosa Soares
Arthur de Oliveira Ferreira
Andrey Fellipe Conrado
Luiz Fernando Santos Sobral
João Victor Argôlo Gomes

Gerenciador de eventos

Um gerenciador de eventos desenvolvido com JavaFX para a interface, Spring Boot e JPA para a lógica e persistência de dados, utilizando MySQL como banco de dados e Docker para containerização e fácil implantação.

Aracaju - SE
2025

Pedro Faria Barreto
Alan Rocha Barbosa Soares
Arthur de Oliveira Ferreira
Andrey Fellipe Conrado
Luiz Fernando Santos Sobral
João Victor Argôlo Gomes

Gerenciador de eventos

Um gerenciador de eventos desenvolvido com JavaFX para a interface, Spring Boot e JPA para a lógica e persistência de dados, utilizando MySQL como banco de dados e Docker para containerização e fácil implantação.

ATIVIDADE sobre o projeto de gerenciamento de eventos apresentado como requisito parcial da avaliação da disciplina Projeto de programação, ministrada pela Prof. Layse Santos Souza, no 2º semestre de 2025.

Aracaju - SE
2025

Sumário

1 INTRODUÇÃO

Este relatório apresenta o desenvolvimento de um sistema de gerenciamento de eventos voltado para o controle e organização de informações relacionadas a eventos, participantes e agendas. O projeto tem como objetivo oferecer uma solução eficiente e escalável, combinando boas práticas de engenharia de software com tecnologias amplamente utilizadas no mercado.

A aplicação foi construída utilizando JavaFX para a criação de uma interface gráfica intuitiva e responsiva, proporcionando uma experiência agradável ao usuário. No back-end, o Spring Boot foi adotado como framework principal, garantindo agilidade no desenvolvimento, padronização e integração com o JPA (Java Persistence API), responsável pelo mapeamento e persistência dos dados em um banco MySQL. Para assegurar maior portabilidade e facilidade de implantação, a aplicação foi containerizada com Docker, permitindo sua execução em diferentes ambientes de forma consistente.

Este documento detalha o processo de concepção, desenvolvimento e implementação do sistema, abordando suas principais funcionalidades, a arquitetura adotada e as tecnologias empregadas, além de apresentar os resultados alcançados e as possibilidades de aprimoramento futuro.

2 JUSTIFICATIVA

A escolha das ferramentas e tecnologias utilizadas no desenvolvimento do sistema de gerenciamento de eventos foi pautada pela busca de eficiência, robustez e facilidade de manutenção. Cada recurso adotado contribuiu para a criação de uma aplicação moderna, escalável e de fácil integração entre suas camadas.

O **JavaFX** foi escolhido para o desenvolvimento da interface gráfica por oferecer uma experiência visual rica e responsiva, além de integração nativa com a linguagem Java. Essa escolha possibilitou a construção de telas intuitivas e com boa comunicação com a camada de negócios.

No backend, o framework **Spring Boot** foi adotado por sua modularidade e simplicidade na configuração, permitindo o desenvolvimento de APIs REST bem estruturadas e compatíveis com o padrão de arquitetura em camadas.

A persistência de dados foi implementada com o **JPA (Java Persistence API)** em conjunto com o **MySQL**, garantindo um mapeamento objeto-relacional eficiente, padronização nas consultas e armazenamento seguro das informações.

Por fim, o uso do **Docker** proporcionou um ambiente de execução isolado e consistente, facilitando o processo de implantação e evitando incompatibilidades entre diferentes sistemas. Essa abordagem também assegura maior portabilidade e praticidade na manutenção do projeto.

Em conjunto, essas tecnologias formam uma base sólida para o desenvolvimento de um sistema completo, confiável e preparado para futuras expansões e melhorias.

3 OBJETIVOS

3.1 OBJETIVO GERAL

O objetivo principal do sistema é permitir o cadastro eficiente e organizado de participantes em eventos, oferecendo uma plataforma organiza todas as informações relacionadas a inscrições, dados pessoais e presença. Com essa funcionalidade, o sistema busca facilitar o gerenciamento de participantes, permitindo consultas rápidas, atualizações precisas e controle efetivo da participação em diferentes eventos. Além disso, ao automatizar o processo de cadastro, a aplicação reduz erros manuais, otimiza o tempo dos organizadores e contribui para uma experiência mais intuitiva e acessível tanto para administradores quanto para os próprios participantes. Dessa forma, o sistema não apenas registra informações, mas também suporta a organização e o planejamento de eventos de forma estruturada e confiável, garantindo que todos os dados sejam armazenados de maneira segura e possam ser consultados ou analisados sempre que necessário.

3.2 OBJETIVOS ESPECÍFICOS

1. Organizar os usuários por cargos.
2. Deixar a interface intuitiva.
3. Garantir segurança nos dados dos usuários.

4 METODOLOGIA

A metodologia adotada para o desenvolvimento do sistema foi o desenvolvimento incremental, na qual cada componente é implementado e testado de forma independente, garantindo maior controle sobre a qualidade e o funcionamento de cada módulo. O sistema foi organizado em camadas, com responsabilidades bem definidas, conforme descrito a seguir:

- **Model:** define as entidades do sistema, como *Evento*, *Usuário* e *Seguro*, representando a estrutura de dados utilizada.
- **Repository:** interface responsável pela comunicação com o banco de dados por meio da JPA, abstraindo operações de persistência e consultas.
- **Controller:** camada que recebe requisições HTTP, processa dados de entrada e retorna respostas, geralmente no formato JSON, servindo como ponto de interação com o cliente.
- **Service:** camada intermediária encarregada da lógica de negócios, validando regras, coordenando operações entre *Controller* e *Repository* e garantindo o funcionamento consistente do sistema.

Essa estrutura em camadas permite maior modularidade, manutenção facilitada e escalabilidade, além de promover a separação clara entre a interface, a lógica de negócios e a persistência de dados.

5 RESULTADOS E DISCUSSÕES

O sistema desenvolvido demonstrou-se plenamente funcional, atendendo de forma satisfatória aos requisitos estabelecidos durante a fase de planejamento. As principais funcionalidades implementadas incluem:

- **Cadastro de eventos:** permite o registro detalhado de eventos, incluindo informações como nome, data, local, descrição e responsáveis;
- **Consulta e atualização de dados:** possibilita a listagem, edição e exclusão de eventos previamente cadastrados, garantindo a consistência das informações armazenadas;
- **Persistência de dados:** utiliza o banco de dados **MySQL** em conjunto com o **Spring Data JPA**, assegurando integridade relacional e armazenamento permanente das informações;
- **Implantação automatizada:** emprega o **Docker Compose** para orquestrar os contêineres do backend e do banco de dados, simplificando o processo de inicialização e integração entre os serviços.

Durante os testes, foi possível comprovar o correto funcionamento das operações CRUD, evidenciando a comunicação eficiente entre o backend e o banco de dados. As requisições enviadas via **Postman** retornaram respostas consistentes, confirmando que os dados eram persistidos, recuperados e atualizados adequadamente. Dessa forma, o sistema mostrou-se estável, modular e pronto para futuras expansões.

6 CONSIDERAÇÕES FINAIS

O desenvolvimento do **Sistema de Gerenciamento de Eventos** possibilitou aplicar, na prática, conceitos de programação, banco de dados e virtualização. O sistema atendeu aos objetivos propostos, demonstrando eficiência no cadastro e gerenciamento de eventos.

1. utilização do **Spring Boot** para criação de APIs REST de forma organizada e modular;
2. integração com o **MySQL** por meio do **Spring Data JPA**, garantindo persistência dos dados;
3. uso de contêineres **Docker** para facilitar a execução e a portabilidade do sistema;
4. validação das rotas e operações via **Postman**, assegurando o correto funcionamento da API.

O projeto mostrou-se funcional e escalável, servindo como base para futuras melhorias e expansão do sistema.

7 REFERÊNCIAS

1. Documentação do Spring Boot;
2. Documentação do JavaFx;
3. Documentação do MySQL;
4. Slides fornecidos em sala de aula.
5. https://docs.spring.io/spring-boot/documentation.html
6. https://www.devmedia.com.br/mysql-tutorial/33309
7. https://docs.oracle.com/javase/8/javase-clienttechnologies.htm

8 ANEXOS

Prints de saída: staff participando de evento, saída no terminal e criação de um staff.


```
[
    {
        "nome": "Luiz",
        "cpf": "11122233344",
        "email": "luiz@example.com",
        "senha": "1234",
        "dataNasc": "2005-03-12",
        "telefone": null,
        "perfil": null,
        "idStaff": 1,
        "especializacao": "mixagem",
        "eventoAuxiliado": {
            "idEvento": 1,
            "nomeEvento": "Workshop Java",
            "descricaoEvento": "Aprenda Spring Boot",
            "dataInicio": "2025-10-20",
            "dataFim": "2025-10-21",
            "capacidade": 50
        }
    }
]
```


```
mysql> SELECT * FROM staffs LIMIT 10;
+----------+-------------+------------+------------------+-------+-------+---------------+----------------------------+
| id_staff | cpf         | data_nasc  | email            | nome  | senha | especializacao | evento_auxiliado           |
+----------+-------------+------------+------------------+-------+-------+---------------+----------------------------+
|        1 | 11122233344 | 2005-03-12 | pedro@example.com | Pedro | 1234  | mixagem        | Taylor_Swift:_The_Eras_Tour |
+----------+-------------+------------+------------------+-------+-------+---------------+----------------------------+
1 row in set (0.00 sec)
```