Universidade Tiradentes
Ciência da Computação

IAGO VIEIRA ROCHA
JOÃO GABRIEL MENDONÇA RIBEIRO
JOÃO PAULO LORDÊLO PEDREIRA VIVAS
MARIANA MENEZES CORREIA
RAFAEL CALMON GONZAGA SANTOS

MÓDULO DE NOTIFICAÇÕES

Desenvolvimento do módulo de notificações para um sistema de gerenciamento de eventos

Aracaju - SE
2025

IAGO VIEIRA ROCHA
JOÃO GABRIEL MENDONÇA RIBEIRO
JOÃO PAULO LORDÊLO PEDREIRA VIVAS
MARIANA MENEZES CORREIA
RAFAEL CALMON GONZAGA SANTOS

MÓDULO DE NOTIFICAÇÕES

Desenvolvimento do módulo de notificações para um sistema de gerenciamento de eventos

ATIVIDADE sobre Sistema de Notificações 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

A cidade de Nova Aurora está se consolidando como um importante polo cultural e acadêmico, sediando ao longo do ano uma ampla variedade de eventos, como conferências, workshops, shows, competições esportivas e feiras. Em virtude desse crescimento, a prefeitura contratou o grupo para o desenvolvimento de um Sistema de Gestão de Eventos, destinado a gerenciar inscrições, pagamentos, agendas, notificações e feedbacks de forma automatizada e eficiente. Aqui será relatado o processo de desenvolvimento voltado para o módulo de notificações.

O presente relatório tem por finalidade apresentar, de maneira detalhada e organizada, as atividades realizadas semanalmente pelo Grupo 06 durante o desenvolvimento do referido sistema. O projeto está inserido no contexto do Projeto da disciplina Projeto de Programação, e busca aplicar na prática os conceitos aprendidos ao longo do curso, por meio da utilização de ferramentas modernas de desenvolvimento e boas práticas de engenharia de software.

A execução do projeto foi conduzida de forma colaborativa, seguindo etapas de planejamento, implementação e integração contínua. O relatório semanal, portanto, representa o registro formal do progresso técnico, das decisões de projeto e das tarefas executadas ao longo do período, possibilitando o acompanhamento do desempenho do grupo e a análise evolutiva do sistema desenvolvido.

2 JUSTIFICATIVA

Um sistema de gerenciamento de eventos será de extrema importância pois irá permitir maior organização, não só para os participantes de inscreverem, realizarem pagamentos e verem eventos de seu interesse, como também para os organizadores e administradores desses eventos que vão poder ter um maior controle sobre as inscrições realizadas em seus eventos e disponibilizar informações sobre eles de forma mais fácil para possíveis participantes. O módulo de notificações é de extrema importância pois é por meio dele que os usuários serão alertados sobre possíveis alterações nos eventos como mudança de data ou local, além de receberem confirmações referentes aos seus pagamentos e inscrições e lembretes quando os eventos estiverem se aproximando.

3 OBJETIVOS

3.1 OBJETIVO GERAL

Desenvolver o módulo de notificações, possibilitando o envio de notificações para vários usuários do sistema, persistindo os dados referentes a essas notificações, e permitindo aos usuários visualizarem suas respectivas notificações e marcá-las como lidas. Além da interface visual para integração com o mesmo.

3.2 OBJETIVOS ESPECÍFICOS

1. Elaborar endpoint para listagem de notificações novas;
2. Elaborar endpoint para listagem de todas as notificações do sistema;
3. Elaborar endpoint para listagem de um usuário específico;
4. Elaborar endpoint para atribuição de tags para uma notificação;
5. Elaborar endpoint para listagem de notificações filtradas por tipo;
6. Elaborar endpoint para realizar inscrição de usuário em um evento;
7. Elaborar endpoint para realizar pagamento de uma inscrição;
8. Elaborar endpoint para gerenciar as preferências de um usuário;
9. Elaborar endpoints para o gerenciamento de Tags, Usuários e Evento;
10. Elaborar métodos para enviar notificações de confirmação de pagamento;
11. Elaborar gerenciamento de permissão de usuário a partir do seu tipo (Administrador, Organizador ou Participante);
12. Elaborar métodos para registros de Logs referentes às notificações.

4 METODOLOGIA

Para o desenvolvimento do projeto nos baseamos numa arquitetura clean, dividindo o projeto em pastas de Service, Repository, Controller, DTO e Entity (Figura 1).

Principais atividades realizadas ao longo de cada semana no desenvolvimento do projeto:

SEMANA 01

1. Criação do repositório e das respectivas branches dos integrantes do grupo;
2. Divisão das classes por integrantes do grupo;
3. Início da criação das Classes principais;
4. Mudança no projeto para desenvolver em Spring Boot.

SEMANA 02

1. Criação da nova versão do projeto em Spring Boot;
2. Divisão em pacotes de Controller, Service, Repository e Entity para melhor organização;
3. Início do desenvolvimento agora em Spring Boot;
4. Primeira tentativa de envio de notificação para vários usuários.

SEMANA 03

1. Ajustes nas classes relativas a notificação;
2. Ajustes das branchs para realizar o merge na main;
3. Atualização das branches com a main atualizada;
4. Criação do banco para persistência dos dados.

SEMANA 04

1. Finalização das classes relativas a eventos e usuário;
2. Ajustes nas classes relativas a notificações;
3. Modificação da classe eventos, criando a DTO, o Repositório e o Service;
4. Início do digrama de classes embasado no projeto em Spring Boot;

5. Realização da integração do frontend e do backend.

SEMANA 05

1. Finalização do diagrama de classes embasado no projeto em Spring Boot;
2. Finalização da documentação;
3. Últimos testes e ajustes na funcionalidades do sistema.

5 RESULTADOS E DISCUSSÕES

Os resultados principais foram a construção de uma API de notificações ,para um aplicativo gerenciador de eventos, e a criação de uma interface para a utilização da API. O backend feito em Java 21 com o framework Spring Boot, contava com diversas entidades, DTO's, Services, Controllers e Repositorys, a fim de se comunicar com a interface e conseguir armazenar os dados informados por formulários em um banco de dados criado.

Dessa forma, cada entidade possui sua tabela e consegue se comunicar tranquilamente com o banco. Dessa forma, organizamos os pacotes da API separando por Controller, Service e Repository, para facilitar o entendimento do código. O **NotificacaoController.java** contém todos os endpoints relacionado à exibição, cadastro, atualização e deleção de notificações, permitindo o Frontend a se comunicar com o Backend da aplicação. Dessa forma, o Controller chama o **NotificacaoService.java** para tratar e aplicar regras de negócio sob os dados enviados pela interface criada, então, utiliza o NotificacaoRepository.java para salvar, alterar, buscar, ou deletar dados do banco de dados. Além disso, foram utilizados os DTO's **NotificacaoDTO.java**, **NotificacaoLidaDTO.java**, **NotificacaoListagemDTO.java** e **NotificacaoDeletarDTO.java** para formatar os dados de saída e entrada, além de evitarem pequenos erros na passagem de dados.

Por fim, para o registro de métodos REST dos tipos POST, PUT e DELETE, utilizamos a entidade LogsNotificacao.java, além de seu Repository, **LogsNotificacaoRepository.java**, e seu DTO, **CriarLogNotificacaoDTO.java**, para salvar qualquer requisição feita à API, além da definição de preferências com a entidade **Preferencia.java** e seus respectivos Controller, Service e Repository.

6 CONSIDERAÇÕES FINAIS

O presente relatório detalhou o processo de desenvolvimento do Módulo de Notificações para um Sistema de Gerenciamento de Eventos, um requisito parcial para a avaliação da disciplina Projeto de Programação. O projeto, inserido no contexto do Projeto Integrador, teve como Objetivo Geral o desenvolvimento deste módulo, com foco no envio, persistência, visualização e marcação de notificações como lidas por diversos usuários.

A metodologia adotada baseou-se em uma arquitetura clean, com a divisão do projeto nas pastas de Service, Repository, Controller, DTO e Entity. As atividades foram conduzidas em cinco semanas, englobando desde a criação do repositório e divisão de tarefas até a finalização do backend em Java 21 com o framework Spring Boot, a criação do banco de dados para persistência, e a integração com o frontend.

Como Resultado Principal, foi entregue uma API de notificações e uma interface para sua utilização. O backend foi estruturado em pacotes organizados (Controller, Service, Repository), utilizando DTOs específicos para formatar dados de entrada e saída e implementando a persistência de Logs para requisições REST (POST, PUT, DELETE). Os resultados demonstram o alcance dos objetivos propostos, como a construção de endpoints para gerenciamento de Tags, Usuários, Eventos e as próprias notificações, incluindo listagem e filtragem por tipo.

O desenvolvimento deste módulo permitiu a aplicação prática de conceitos de engenharia de software, proporcionando aos integrantes do Grupo 06 uma experiência valiosa em desenvolvimento colaborativo, planejamento e integração contínua. O sistema de notificações se mostra de extrema importância para o projeto, pois alerta os usuários sobre alterações, pagamentos e lembretes de evento.

7 REFERÊNCIAS

SPRING. Spring Boot. Disponível em: https://spring.io/projects/spring-boot. Acesso em: 10 nov. 2025.

ORACLE. Java SE 21 Development Kit Documentation. [S. l.]: Oracle, c2023. Disponível em: https://www.oracle.com/java/technologies/javase-jdk21-doc-downloads.html. Acesso em: 10 nov. 2025.

8 ANEXOS

Figuras 1

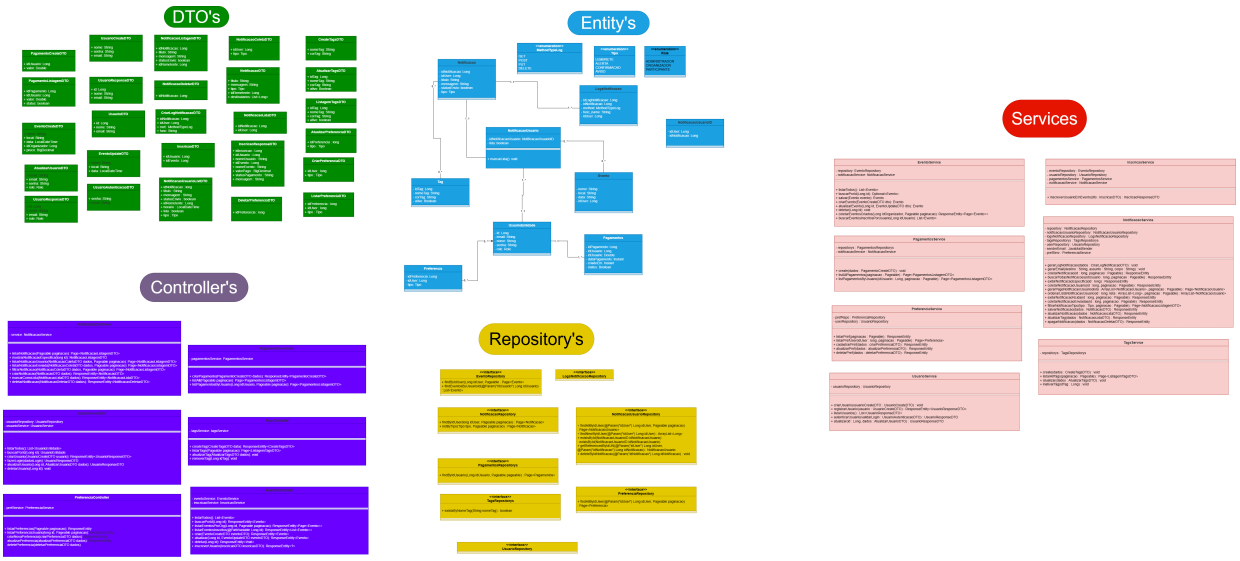

Figura 1 – Diagrama de Classes atualizado

https://drive.google.com/file/d/1m3tvI96z_jrta8mGjrDWVpk5FikkJHrZ/view?usp=drive_link