



**ABRAÃO MARTINS DOS SANTOS NETO  
ANTONIO CALU DE OLIVEIRA NETO  
JOÃO GABRIEL TRINDADE ALVES  
JOÃO GUILHERME MENDES CRUZ DOS SANTOS  
JOÃO PEDRO WEBER DE ARAUJO VIEIRA**

**Entrega Parcial  
Grupo 5 - Agenda e Programação**

**Aracaju  
2025**

**ABRAÃO MARTINS DOS SANTOS NETO  
ANTONIO CALU DE OLIVEIRA NETO  
JOÃO GABRIEL TRINDADE ALVES  
JOÃO GUILHERME MENDES CRUZ DOS SANTOS  
JOÃO PEDRO WEBER DE ARAUJO VIEIRA**

**Entrega Parcial  
Grupo 5 - Agenda e Programação**

Entrega Final equivalente ao sistema de Agenda e Programação da matéria de Projeto Programação a respeito do projeto da 2ª Unidade apresentado à Universidade Tiradentes

Orientador(a): Layse Santos Souza

## **RESUMO**

O presente trabalho apresenta o desenvolvimento completo do sistema Agenda e Programação, criado para a disciplina de Projeto de Programação da Universidade Tiradentes. O sistema visa oferecer uma plataforma eficiente de organização de eventos, compromissos e tarefas, com persistência em banco de dados MySQL, notificações personalizadas e controle de acesso por diferentes níveis de usuários (Administrador, Usuário e Visualizador). A implementação foi realizada em Java, seguindo o padrão em camadas (MVC), com uso de JavaFX para interface gráfica, JDBC para conexão ao banco e arquitetura modular que separa responsabilidades entre model, repository, service, controller e ui. A entrega final compreende as fases de planejamento, modelagem, implementação e testes, consolidando uma aplicação funcional, escalável e aderente aos requisitos especificados.

## **ABSTRACT**

This document presents the complete development of the Agenda and Scheduling System, created for the Programming Project course at Universidade Tiradentes. The system aims to provide an efficient platform for organizing events and appointments, using a MySQL database for data persistence, personalized notifications, and access control for multiple user roles (Administrator, User, and Viewer). The system was implemented in Java, following the MVC layered architecture, with JavaFX for the user interface and JDBC for database integration.

## SUMÁRIO

<b>1. INTRODUÇÃO.....</b>	<b>4</b>
<b>2. OBJETIVOS.....</b>	<b>5</b>
2.1. Objetivos Gerais.....	5
2.2. Objetivos Específicos.....	5
<b>3. ESCOPO DO PROJETO.....</b>	<b>5</b>
3.1. Funcionalidades Principais.....	5
3.2. Funcionalidade Secundárias.....	6
<b>4. DESCRIÇÃO GERAL DO SISTEMA.....</b>	<b>6</b>
4.1. Pública-alvo.....	6
4.2. Tecnologias Utilizadas.....	6
<b>5. ARQUITETURA DO SISTEMA.....</b>	<b>7</b>
Arquitetura segue o modelo MVC, sendo dividido em:.....	7
<b>6. REQUISITOS DO SISTEMA.....</b>	<b>8</b>
6.1 Requisitos Funcionais.....	8
6.2 Requisitos não Funcionais.....	9
<b>7. BANCO DE DADOS.....</b>	<b>10</b>
<b>8. INTERFACE DO USUÁRIO.....</b>	<b>10</b>
<b>9. PRINCIPAIS FLUXOS DO SISTEMA.....</b>	<b>10</b>
9.1. Fluxo de Relatórios.....	10
9.2. Notificações.....	10
<b>10. EXECUÇÃO.....</b>	<b>10</b>
<b>11. CONCLUSÃO.....</b>	<b>10</b>

## **1. INTRODUÇÃO**

O projeto Agenda e Programação tem como propósito criar uma aplicação que auxilie na gestão de eventos e tarefas, permitindo que os usuários visualizem, cadastrem e gerenciem compromissos de maneira prática e intuitiva.

A plataforma também visa oferecer recursos como notificações automáticas, filtros personalizados, alertas de lembrete e controle de acesso conforme o tipo de usuário. A escolha pelo Java se deu por sua robustez e ampla compatibilidade com bibliotecas de interface e persistência.

## **2. OBJETIVOS**

### **2.1. Objetivos Gerais**

Desenvolver um sistema desktop em Java que permita o gerenciamento de agendas e eventos de forma dinâmica, com armazenamento persistente e integração entre usuários.

### **2.2. Objetivos Específicos**

- Implementar interface gráfica com JavaFX;
- Criar banco de dados MySQL com suporte a múltiplos usuários e eventos;
- Desenvolver sistema de notificações automáticas e personalizadas;
- Garantir controle de acesso por níveis de permissão;
- Permitir filtragem e busca de eventos.

## **3. ESCOPO DO PROJETO**

### **3.1. Funcionalidades Principais**

- Gerenciamento de agenda;
- Envio e configuração de notificações;
- Emissão de relatórios de eventos;
- Integração com banco de dados via JDBC;

- Interface JavaFx e JavaCalendar(Classe AgendaControllerFx);
- Filtrar eventos por dia, semana, mês e ano;

### **3.2. Funcionalidade Secundárias**

- Validação de erros via exceções customizadas;
- MVC dividido em model, service, repository e controller.

## **4. DESCRIÇÃO GERAL DO SISTEMA**

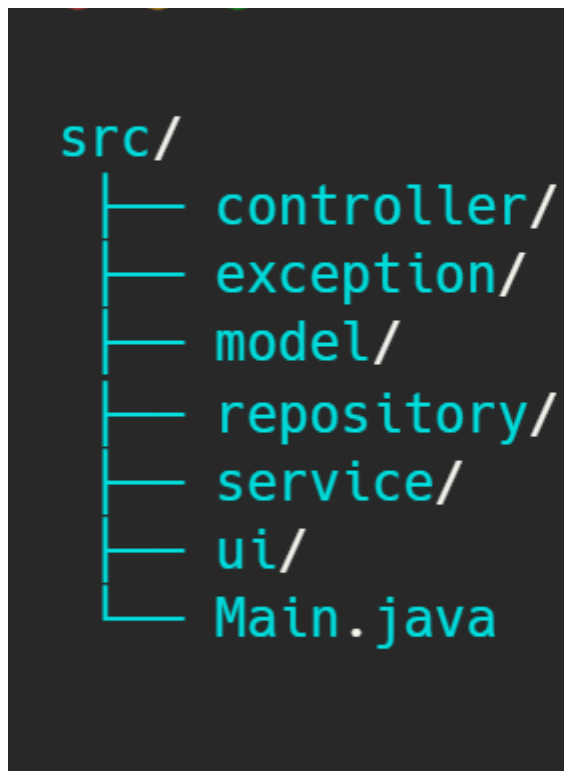
### **4.1. Pública-alvo**

- Usuário que deseje visualizar agendas e eventos;
- Administrados Internos.

### **4.2. Tecnologias Utilizadas**

- Java 17+;
- JavaFx;
- JDBC para acesso ao banco de dados;
- IDE: IntelliJ, VS Code e MySQL Workbench.

## 5. ARQUITETURA DO SISTEMA



Arquitetura segue o modelo MVC, sendo dividido em:

- Controller → recebe solicitações da UI e chama serviços;
- Service → regras de negócio (AgendaService, UsuarioService, etc.);
- Repository → acesso ao banco via JDBC;
- Model → classes de domínio, possui os modelos das principais classes do sistema;
- UI → camada JavaFX (AgendaControllerFx);
- Main → Execução principal para inicialização do sistema



## 6. REQUISITOS DO SISTEMA

### 6.1 Requisitos Funcionais

Requisito Funcional	Descrição
RF01	O sistema deve permitir ao admin cadastrar, modificar e remover eventos.
RF02	O sistema deve permitir ao usuário visualizar agenda diária, semanal e mensal.
RF03	O sistema deve permitir ao usuário visualizar sua agenda de compromissos.
RF04	O sistema deve permitir ao usuário visualizar todos os eventos disponíveis no calendário.
RF05	O sistema deve permitir ao usuário visualizar eventos passados, presentes e futuros.
RF06	O sistema deve permitir filtrar por categoria e status de eventos divididos entre confirmado, pendente e cancelado.
RF7	O sistema deve possuir persistência de dados como eventos e dados de usuário no banco de dados.
RF8	O sistema deve possuir persistência de dados em relação às alterações dos eventos.
RF9	O sistema deve permitir configurar notificações
RF10	O sistema deve enviar notificações acerca da confirmação de presença dos participantes em eventos.

RF11	O sistema deve permitir ao admin controle de acesso a eventos privados e confidenciais.
------	---

## 6.2 Requisitos não Funcionais

Requisito não Funcional	Descrição
RNF01	O sistema deve conter uma interface simples e intuitiva.
RNF02	O sistema deve cumprir todas as leis e regulamentos aplicáveis em relação a LGPD acerca da segurança de dados.
RNF03	O sistema deve assegurar que os dados sensíveis dos usuários não sejam expostos a terceiros .
RNF05	O software deve suportar até 500 usuários simultâneos sem degradação de desempenho .
RNF05	O sistema deve estar disponível em português .
RNF06	O sistema deve oferecer feedback claro para as ações do usuário mostrando confirmações, erros e alertas.
RNF08	O código deve seguir boas práticas de desenvolvimento para facilitar a .manutenção
RNF09	O sistema deve armazenar os dados dos usuários em um banco de dados MySql.
RNF11	O sistema deve permitir a busca de eventos especificados pelo usuário.
RNF12	O sistema deve enviar notificações ao usuário de eventos em que o usuário está cadastrado num período de 24 horas
RNF13	O sistema deve permitir a visualização de eventos sem a necessidade do usuário estar cadastrado.

## 7. BANCO DE DADOS

[Modelo do banco de dados aqui](#)

## 8. INTERFACE DO USUÁRIO

O sistema utiliza JavaFx e JCalendar para interação, com:

- Tela principal da agenda
- Tabela de visualização de eventos
- Botões de filtro de mês e ano
- Botões de relatório e configuração de notificações

## 9. PRINCIPAIS FLUXOS DO SISTEMA

### 9.1. Fluxo de Relatórios

- Usuário Solicita Relatório na ui (AgendaControllerFx);
- Ui chama RelatorioService;
- RelatorioService formata e devolve à interface;
- Interface exibi informações para o usuário.

### 9.2 Notificações

- Usuário seleciona botão notificação da ui;
- UI chama NotificacaoService que trata a solicitação verificando os próximos eventos do usuário nas próximas 24 horas;
- NotificacaoService devolve um StringBuilder contendo todos os eventos próximos.

## 10. EXECUÇÃO

Rodar a classe principal:

Main.java

## 11. CONCLUSÃO

O sistema apresenta uma arquitetura baseada em MVC, com camadas bem definidas e foco na organização de eventos e agendas. A modularização permite fácil expansão para futuras melhorias.