**PROJETO.....**

**Requisitos da disciplina Modelagem de Software e Arquitetura de Sistemas**

**INTEGRANTES DO PROJETO e RA’S**

Caio dos Santo Gomes - 24026876

Vitor Rodrigues dos Santos 23025502

São Paulo

2025

Sumário

[1 INTRODUÇÃO 3](#_Toc179916120)

[2. DOCUMENTO DE ABERTURA DO PROJETOS 3](#_Toc179916121)

[3. REQUISITOS DE SISTEMA 4](#_Toc179916122)

[3.1 REQUISITOS FUNCIONAIS DE SOFTWARE 4](#_Toc179916123)

[3.2 REQUISITOS NÃO FUNCIONAIS DE SOFTWARE 5](#_Toc179916124)

[4. CASOS DE USO 5](#_Toc179916125)

[5. DIAGRAMA DE CLASSE 6](#_Toc179916126)

[6. ARQUITETURA DO SISTEMA 6](#_Toc179916127)

[7. REFERÊNCIAS BIBLIOGRÁFICAS 6](#_Toc179916128)

# 1 INTRODUÇÃO

Instruções de Entrega: Documento de abertura do projeto, Contextualização e

objetivo da solução, Identificação de

Requisitos funcionais e não funcionais.

Descrição: Identificar 6 requisitos

funcionais e 6 não funcionais do sistema.

# 2. DOCUMENTO DE ABERTURA DO PROJETOS

**Prefácio**

Este documento destina-se a:

* **Desenvolvedores**: Para implementação e manutenção do sistema.
* **Gerentes de Projeto**: Para acompanhamento do escopo e progresso.
* **Administradores do Sistema**: Para suporte técnico e operacional.
* **Stakeholders**: Para compreensão das funcionalidades e objetivos do sistema.

Primeira versão, criada no dia 24/02/2025, código e implementação de desing.

**Introdução**

Administrador: Usuário responsável por gerenciar os eventos e informações do site  
Conjunto de rotinas e padrões de programação para acesso a um aplicativo ou plataforma. Back-end: Parte do sistema responsável pela lógica de negócios, banco de dados e comunicação com o servidor. Banco de Dados: Estrutura que armazena informações utilizadas pelo sistema.

Ocasião beneficente cadastrada no banco de dados, com informações como data, horário, local e descrição.

Painel de Controle: Interface exclusiva para administradores gerenciarem eventos e informações.

**Glossário**

* **CRUD (Create, Read, Update, Delete)** – Conjunto de operações básicas utilizadas para manipular dados em um sistema, como criar, visualizar, atualizar e excluir registros.
* **Backend** – Parte do sistema responsável pelo processamento de dados e lógica de negócios, geralmente rodando em um servidor.
* **Frontend** – Interface gráfica do sistema com a qual os usuários interagem.
* **API (Application Programming Interface)** – Conjunto de regras que permite a comunicação entre diferentes sistemas ou módulos.
* **Banco de Dados** – Estrutura que armazena e organiza os dados do sistema, como usuários, eventos e ingressos.
* **QR Code** – Código bidimensional usado para armazenar informações e facilitar a validação de ingressos.
* **Autenticação** – Processo de verificação da identidade do usuário ao acessar o sistema.
* **Validação de Ingresso** – Procedimento que confirma a autenticidade do ingresso para entrada no evento.

**Definição de requisitos de usuário**

Para Administradores**:**

Login e autenticação segura e criação de novos eventos com as seguintes informações:

* Nome do evento
* Data e horário
* Descrição
* Endereço completo
* Imagem representativa (opcional)
* Contato do organizador
* Edição e exclusão de eventos existentes.
* Painel de controle para gerenciar todos os eventos.

Para Sub Administradores:

* Validação de ingressos
* Lista de nomes cadastrados para o evento
* Gerar novos ingressos

Para Usuários:

* Visualização do evento
* Abrir fotos em Fullscreen
* Criar login com email e CPF
* Gerar ingresso para o evento selecionado

**Arquitetura do sistema**

* O usuário acessa o sistema via navegador e interage com a interface.
* As ações são enviadas ao backend, que processa os dados e executa as regras de negócio.
* O backend consulta ou atualiza o banco de dados conforme necessário.
* O resultado é retornado ao frontend para exibição ao usuário.

**Especificação de requisitos do sistema**

**Requisitos Funcionais**

* Criar conta informando nome, e-mail e CPF.
* Fazer login no sistema.
* Visualizar eventos disponíveis.
* Gerar ingresso para eventos.
* Baixar o ingresso gerado em formato digital (PDF ou QR Code).

**Requisitos Não Funcionais**

* O sistema deve ser responsivo para desktop e dispositivos móveis.
* O processo de cadastro e geração de ingressos deve ser intuitivo e rápido.
* As senhas devem ser armazenadas de forma segura.

**Administrador**

O administrador é responsável por gerenciar os eventos e usuários no sistema.

**Requisitos Funcionais**

* Criar, editar e excluir eventos.
* Definir data, horário, local e descrição dos eventos.
* Gerenciar usuários, podendo habilitar ou desabilitar contas.
* Gerenciar permissões de subadministradores.
* Visualizar relatórios sobre os eventos, incluindo número de ingressos gerados e participantes confirmados.

**Requisitos Não Funcionais**

* O acesso ao painel administrativo deve ser protegido por autenticação reforçada.
* As ações administrativas devem ser registradas para auditoria.
* O sistema deve permitir a escalabilidade para adicionar novos administradores.

**Subadministrador**

O subadministrador auxilia na organização dos eventos, principalmente na validação de ingressos.

**Requisitos Funcionais**

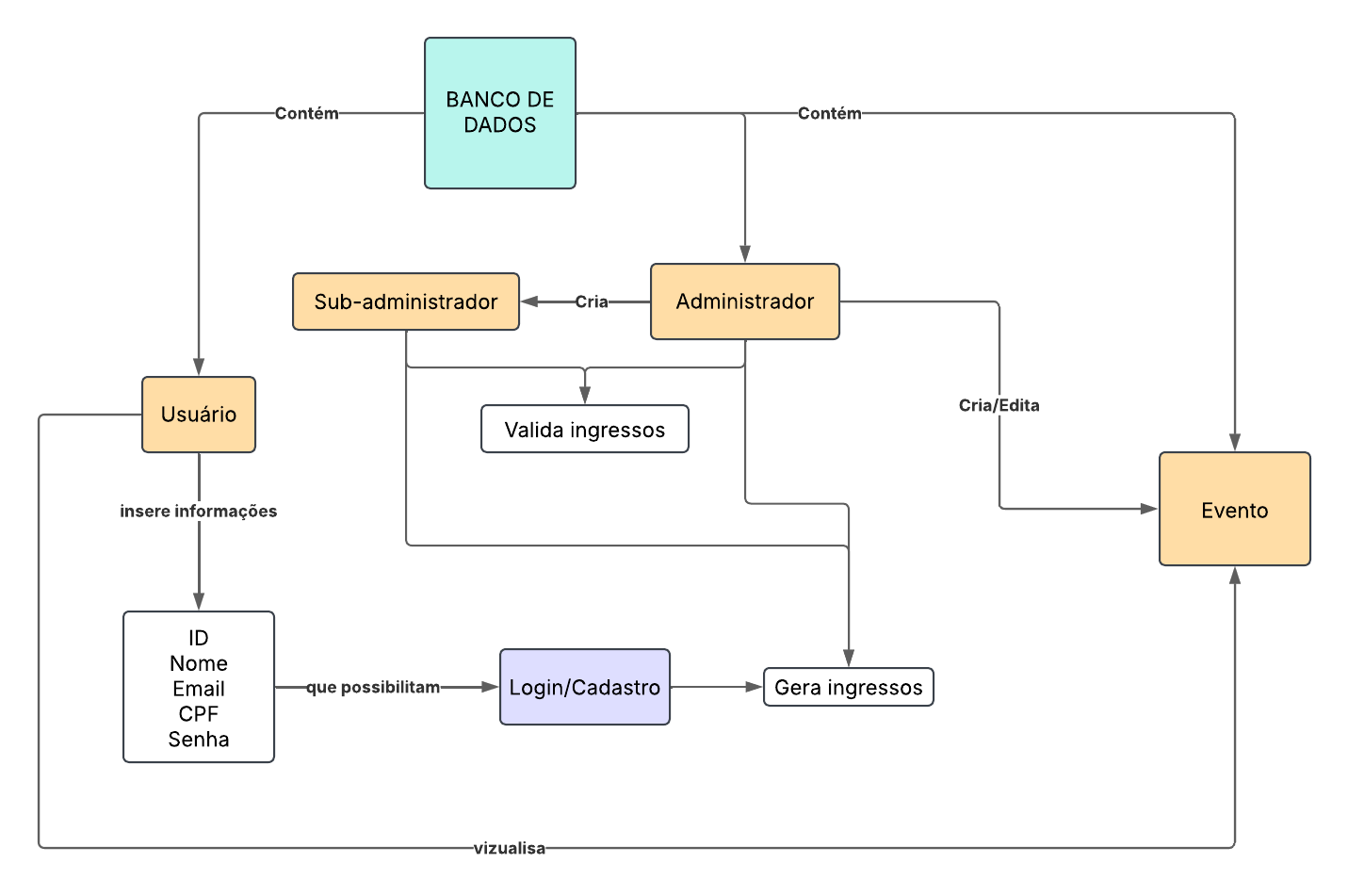
* Validar ingressos dos participantes no dia do evento.
* Fazer a leitura de QR Code ou digitar código manualmente.
* Confirmar presença dos visitantes ao evento.

**Requisitos Não Funcionais**

* A validação de ingressos deve ocorrer de forma rápida para evitar filas.
* O subadministrador não pode modificar eventos, apenas validar ingressos.
* O sistema deve garantir que cada ingresso só possa ser validado uma única vez.

**Modelos do sistema**

O administrador terá acesso a tudo, tanto na edição visual como na de dados. Todos os participantes do sistema podem visualizar os eventos.

**Evolução do sistema**

O sistema será projetado com base nos seguintes pressupostos:

* O site terá um **ciclo de vida curto**, sendo utilizado para eventos específicos.
* O sistema funcionará majoritariamente **online**, mas algumas funcionalidades, como a validação de ingressos, podem ser otimizadas para **uso offline**.( Armazenando em disco os ingressos gerados junto com suas informações)
* O banco de dados armazenará apenas **informações essenciais**, evitando sobrecarga e garantindo eficiência.
* A autenticação de usuários será feita de forma **simples e segura**, ( login e senha)

pode evoluir para suportar múltiplos eventos simultâneos, integração com pagamentos, aplicativo mobile e ferramentas de análise. Para garantir flexibilidade, a arquitetura é modular, com banco de dados escalável e APIs padronizadas. No futuro, melhorias na segurança, automação de relatórios e integração com redes sociais podem ser implementadas sem grandes reestruturações.

**Apêndices**

O sistema mínimo para rodar a aplicação é definido pela necessidade de garantir o funcionamento adequado das funcionalidades essenciais, sem sobrecarregar os recursos do ambiente. De acordo com as características da aplicação, as configurações mínimas e ideais para o sistema são as seguintes:

* **Configuração Mínima**:  
  O sistema pode ser executado em um ambiente básico com os seguintes requisitos:
* **Processador:** Intel Core i3 ou equivalente
* **Memória RAM:** 4 GB
* **Armazenamento:** 20-40 GB em HD ou SSD
* **Sistema Operacional:** Windows 10 ou Linux
* **Configuração Ideal**:  
  Para um desempenho mais eficiente e para suportar um maior número de usuários ou processos simultâneos, a configuração ideal seria:
* **Processador:** Intel Core i7 ou equivalente
* **Memória RAM:** 8 GB ou superior
* **Armazenamento:** 100 GB SSD
* **Sistema Operacional:** Windows 10 ou Linux (versões de servidor para ambientes maiores)

# 3. REQUISITOS DE SISTEMA

## 3.1 REQUISITOS FUNCIONAIS DE SOFTWARE

Necessários 6 requisitos

|  |  |
| --- | --- |
| **RFS01** | |
| **Função** | Cadastro de Eventos |
| **Descrição** | O sistema deve validar credenciais para permitir o acesso a áreas administrativas e pessoais. |
| **Entradas** | Título, descrição, data, horário, endereço, imagens. |
| **Fonte** | Administrador. |
| **Saídas** | Evento registrado na base de dados e disponível para visualização. |
| **Ação** | O administrador acessa o painel de eventos e preenche o formulário, o sistema valida os campos obrigatórios (ex.: título não pode ser vazio, data deve estar no futuro), Se os dados forem válidos, o sistema armazena as informações no banco de dados. novo evento é listado na interface pública para os visitantes visualizarem. |
|  |  |
| **RFS02** | |
| **Função** | Permitir que visitantes gerem ingressos para os eventos. |
| **Descrição** | O visitante poderá solicitar um ingresso único e intransferível para um evento disponível. |
| **Entradas** | Seleção do evento e confirmação de solicitação. |
| **Fonte** | Visitante. |
| **Saídas** | Código QR e código alfanumérico do ingresso. |
| **Ação** | O visitante acessa a página do evento e clica em "Gerar ingresso".  O sistema verifica se o evento permite novas inscrições.  Se permitido, o sistema gera um ingresso único contendo um código QR e um código alfanumérico.  O ingresso é vinculado ao usuário e armazenado no banco de dados.  O visitante recebe a confirmação do ingresso via e-mail e no painel de usuário.  O ingresso é gerado apenas se o usuário estiver logado. Caso não esteja ele é direcionado a criação de uma login. |
|  |  |

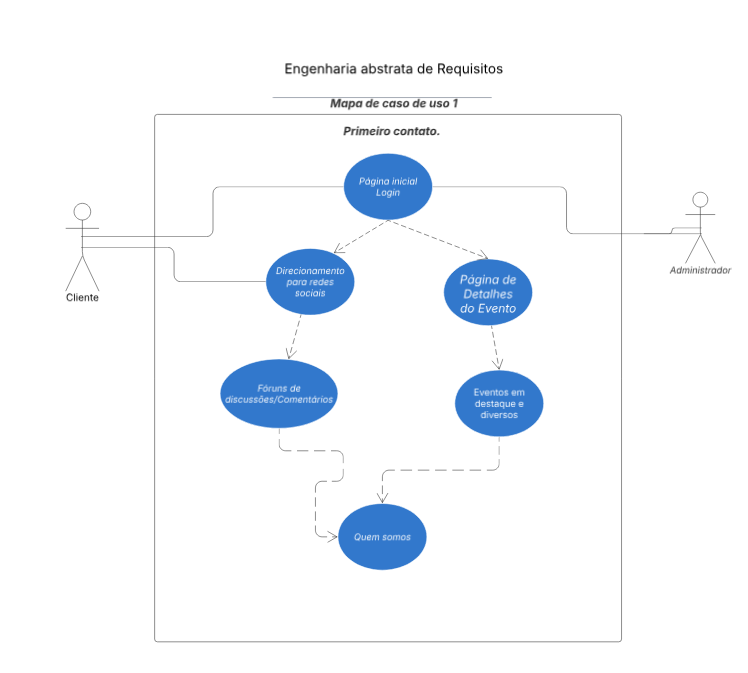
## 3.2 REQUISITOS NÃO FUNCIONAIS DE SOFTWARE

Necessários 6 requisitos

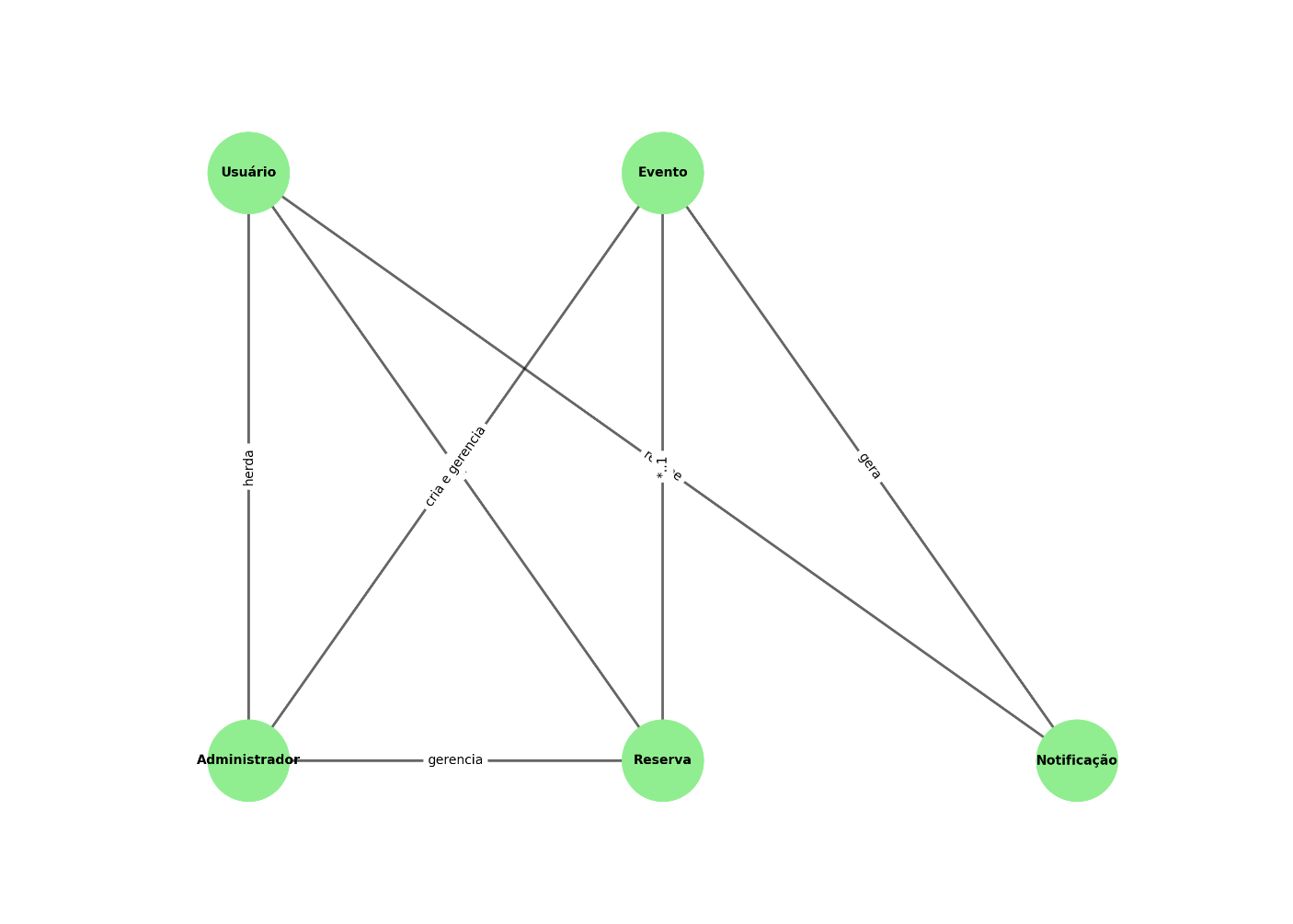
|  |  |
| --- | --- |
| **RFS01** | |
| **Função** | Desempenho |
| **Descrição** | O sistema deve ser capaz de lidar com até 500 visitantes simultâneos sem perda de desempenho. |
| **Entradas** | Acesso simultâneo de 500 usuários ao site. |
| **Fonte** | Visitantes do Site |
| **Saídas** | O site deve carregar de forma rápida, sem travamentos ou lentidão. |
| **Ação** | O sistema deve ser otimizado para uso eficiente de recursos, incluindo balanceamento de carga e caches para melhorar o tempo de resposta. |
|  |  |
| **RFS02** | |
| **Função** | Usabilidade |
| **Descrição** | O site deve ser intuitivo e fácil de usar, com navegação clara e design responsivo. |
| **Entradas** | Navegação e interações do usuário. |
| **Fonte** | Usuários acessando o site. |
| **Saídas** | Experiência de usuário otimizada, com fácil acesso a eventos, registros e informações. |
| **Ação** | Implementação de interface limpa e responsiva, adaptável a diferentes dispositivos (desktop, tablet, celular). |

# 4. CASOS DE USO

Apresentar 3 casos de uso do sistema



# 5. DIAGRAMA DE CLASSE



# 6. ARQUITETURA DO SISTEMA

* **Tecnologias**: HTML, CSS, JavaScript (React, Vue.js, Angular)
* **Função**:

Exibição de eventos.

Formulários para cadastro, login e visualização de detalhes de eventos.

Tela de administração para os administradores gerenciarem os eventos.

**1.2. Camada de Lógica de Negócio (Backend)**

Essa camada lida com o processamento de dados e as regras de negócios do sistema.

* **Tecnologias**: Node.js, Java (Spring Boot), Python (Django, Flask), Ruby on Rails
* **Função**:

Processamento das ações do usuário (login, registro de eventos, inscrições).

Comunicação com a base de dados.

Implementação de lógicas de autenticação e autorização.

Geração de relatórios e notificações.

**1.3. Camada de Persistência (Banco de Dados)**

Essa camada é responsável por armazenar e recuperar dados relacionados ao sistema.

* **Tecnologias**: MySQL, PostgreSQL, MongoDB, SQLite
* **Função**:

Armazenamento dos eventos cadastrados.

Armazenamento das informações de usuários e reservas.

Controle de estados de eventos e reservas.

**API: API da meta integralmente para atomatizar posts do site nas redes sociais**

# 7. REFERÊNCIAS BIBLIOGRÁFICAS

SOMMERVILLE, I. **Engenharia de Software.** 11ª Edição. São Paulo: Pearson Addison-Wesley, 2017.

**Código Fonte TV**: Canal que aborda temas de desenvolvimento de software, programação, arquitetura e infraestrutura de sistemas.

[Filipe Deschamps](https://www.youtube.com/c/FilipeDeschamps): Youtube