**Documentação de Requisitos - Sistema de Reserva de Salas**

**1. Visão Geral do Projeto**

O Sistema de Reserva de Salas é um aplicativo desktop multiusuário desenvolvido em Python, utilizando a biblioteca **PyQt6** para a interface gráfica e **SQLite** para persistência de dados. O objetivo é permitir que usuários cadastrados visualizem a disponibilidade de salas em slots de horário fixos, façam agendamentos e gerenciem suas reservas, aplicando regras estritas de uso.

**2. Requisitos Funcionais (RF)**

Requisitos funcionais definem o que o sistema DEVE fazer para o usuário.

**2.1. Módulo de Autenticação e Cadastro (Login/Cadastro)**

|  |  |  |
| --- | --- | --- |
| **ID** | **Descrição** | **Status** |
| **RF-001** | O sistema deve exibir uma tela inicial com opções para **Cadastrar** ou **Entrar**. | Concluído |
| **RF-002** | O usuário deve poder se cadastrar fornecendo **Nome**, **Matrícula** e **Senha**. | Concluído |
| **RF-003** | O sistema deve validar que a **Matrícula** contém **apenas números**. | Concluído |
| **RF-004** | O sistema deve validar que a **Senha** contém **apenas letras e números (alfanumérica)**. | Concluído |
| **RF-005** | O sistema deve validar que a **Senha** possui um **mínimo de 8 caracteres**. | Concluído |
| **RF-006** | O sistema deve criptografar a senha (usando **Bcrypt**) antes de armazená-la no banco de dados. | Concluído |
| **RF-007** | O sistema deve permitir o **login** de usuários verificando a Matrícula e a Senha (usando a técnica de hash). | Concluído |
| **RF-008** | Deve existir um acesso administrativo via credenciais fixas (admin/12345678) para a tela de gerenciamento. | Concluído |

**2.2. Módulo de Reserva de Salas**

|  |  |  |
| --- | --- | --- |
| **ID** | **Descrição** | **Status** |
| **RF-009** | O sistema deve listar as salas disponíveis com base em **slots de horário fixos** (gerados a partir das 13:00, com 50 minutos de duração). | Concluído |
| **RF-010** | O usuário deve poder selecionar uma sala e escolher um **slot de horário disponível**. | Concluído |
| **RF-011** | O sistema deve registrar a reserva no banco de dados, associando-a à Matrícula do usuário. | Concluído |
| **RF-012** | O sistema deve exibir o **status de ocupação** de cada slot (Disponível, Ocupado por X, ou Minha Reserva). | Concluído |
| **RF-013** | **Regra 1:** O usuário não pode ter mais de **2 reservas ativas** em seu nome (limite total). | Concluído |
| **RF-014** | **Regra 2:** O usuário não pode reservar o **mesmo slot de horário** em duas salas diferentes. | Concluído |
| **RF-015** | O usuário deve poder **cancelar** uma reserva que esteja em seu nome. | Concluído |

**2.3. Módulo de Administração**

|  |  |  |
| --- | --- | --- |
| **ID** | **Descrição** | **Status** |
| **RF-016** | A tela de administração deve listar **todos os usuários cadastrados** em formato de tabela. | Concluído |
| **RF-017** | A tabela de administração deve mostrar o **Nome, Matrícula, o Hash da Senha** e as **Reservas Ativas** de cada usuário. | Concluído |
| **RF-018** | O administrador deve ter a opção de **Excluir** um usuário, removendo-o e todas as suas reservas associadas do sistema. | Concluído |

**3. Requisitos Não Funcionais (RNF)**

Requisitos não funcionais definem a qualidade e as restrições do sistema.

|  |  |  |
| --- | --- | --- |
| **ID** | **Descrição** | **Categoria** |
| **RNF-001** | **Persistência:** Todos os dados (usuários e reservas) devem ser persistentes, sendo mantidos após o fechamento do aplicativo. | Dados |
| **RNF-002** | **Sincronização:** O estado das reservas deve ser sincronizado através do banco de dados, permitindo que as reservas de um usuário sejam visíveis imediatamente para todos os outros. | Confiabilidade |
| **RNF-003** | **Performance:** O tempo de resposta para operações críticas (login, reserva) deve ser inferior a 1 segundo. | Performance |
| **RNF-004** | **Segurança:** As senhas devem ser armazenadas de forma irreversível usando **hashing (Bcrypt)**. | Segurança |
| **RNF-005** | **Usabilidade:** A interface gráfica deve ser **intuitiva e responsiva**, utilizando layouts que se ajustam ao tamanho da janela (PyQt6). | Usabilidade |
| **RNF-006** | **Manutenibilidade:** O código deve ser modularizado (classes em arquivos separados) para facilitar a leitura e futuras expansões. | Manutenibilidade |

**4. Arquitetura e Tecnologias**

|  |  |  |
| --- | --- | --- |
| **Componente** | **Tecnologia** | **Finalidade** |
| **Interface (Cliente)** | Python / **PyQt6** | Criação da interface gráfica de desktop (janelas, botões, tabelas). |
| **Persistência** | Python / **SQLite3** | Banco de dados em arquivo para armazenar usuários e reservas (sincronização multiusuário). |