Documento de Requisitos

Sistema de Reservas

1. Introdução

1.1 Propósito

Este sistema tem como objetivo substituir o processo manual de reserva de salas, auditórios e veículos, garantindo que conflitos sejam evitados e que a gestão das reservas seja mais eficiente.

1.2 Tecnologias Utilizadas

- **Backend:** Django (Django Rest Framework para API)
- **Frontend:** React
- **Banco de Dados:** SQLite
- **Autenticação:** a definir

---

2. Requisitos Funcionais

2.1 Cadastro e Autenticação de Usuários

- Permitir o cadastro de professores e técnicos da UFAM.
- Exigir **SIAPE**, **CPF**, **Nome**, **Email**, **Telefone** e **Senha** no cadastro.
- O SIAPE e o CPF devem ser **únicos** no sistema.
- Validar se o usuário pertence à UFAM antes da aprovação.
- Enviar email de confirmação após o cadastro.
- Permitir que o administrador valide ou rejeite o cadastro.
- O status do usuário pode ser:
    - **Pendente**: Se ainda não foi validado pelo administrador.
    - **Aprovado**: Pode acessar e fazer reservas.
    - **Reprovado**: Não pode acessar.

2.2 Gerenciamento de Reservas

- O sistema deve permitir que professores e técnicos solicitem reservas dos seguintes itens:
    - **Salas de reunião**
    - **Auditórios**

    - **Veículos**
- O sistema deve permitir que cada reserva contenha:
    - **Data de início** e **Data de fim**
    - **Horário de início** e **Horário de término**
    - **Recurso solicitado**
    - **Tipo de reserva (Aula, Exposição, Evento, Colação de Grau, etc)-substituído por um campo de string onde o usuário vai informar a descrição da atividade**
    - **Justificativa**
- Garantir prioridade à primeira solicitação, exceto em eventos prioritários.
- Permitir que o administrador aprove ou rejeite reservas.
- Exigir verificação das condições dos veículos antes da aprovação.

2.3 Painel Administrativo

- Permitir a visualização de todas as reservas.
- Permitir a aprovação ou rejeição de cadastros e reservas.
- Permitir a criação de eventos prioritários que sobreponham reservas comuns.

---

3. Requisitos Não Funcionais

- O sistema deve utilizar uma **API RESTful** para comunicação entre Django e React.
- A autenticação deve ser baseada em **JWT** para segurança.
- O banco de dados deve garantir a **unicidade** de SIAPE e CPF.
- O sistema deve enviar **emails automáticos** para confirmações.
- Utilizar criptografia para senhas.
- Ser acessível via navegador web e responsivo para dispositivos móveis.
- Garantir tempo de resposta inferior a 3 segundos.
- Permitir múltiplos administradores.
- O sistema deve permitir recuperação de senha via email.

---

4. Regras de Negócio

- O acesso ao sistema exige login com CPF ou SIAPE.
- SIAPE e CPF devem ser únicos no sistema, sem cadastros duplicados.
- Apenas usuários "Aprovados" podem acessar e fazer reservas.
- A senha deve ter de 8 a 12 caracteres, incluindo pelo menos uma letra maiuscula, uma minúscula ou um símbolo.
- A confirmação de cadastro é feita pelo administrador. Professores na lista prévia são aprovados automaticamente; os demais ficam pendentes.
- O administrador pode aprovar ou rejeitar cadastros e reservas.
- Reservas de veículos só são confirmadas após inspeção do administrador.

- Em agendamentos simultâneos, apenas o primeiro é registrado; os demais ficam pendentes a não ser que haja uma solicitação com prioridade superior à já alocada.

---

5. Fluxo do Sistema

1. Cadastro do Usuário

1.1. O usuário acessa a página de cadastro.
1.2. Insere os seguintes dados:

- Nome completo
- E-mail institucional
- CPF
- SIAPE (para servidores)
- Senha
- Confirmação de senha
  1.3. O sistema verifica se o CPF e o SIAPE já existem no banco de dados:
- **Se já existirem**, retorna um erro informando que os dados já estão cadastrados.
- **Se forem únicos**, prossegue para a próxima etapa.

2. Confirmação de E-mail

2.1.O sistema deve ser capaz de enviar um e-mail para o usuário contendo 6 dígitos para a confirmação do email.
2.2. o sistema autentica o código inserido.
2.3. Ao solicitar cadastro muda o status para "Pendente de Aprovação pelo Administrador".

3. Aprovação do Cadastro pelo Administrador

3.1. O administrador acessa o painel de administração.
3.2. Visualiza a lista de usuários pendentes e verifica se constam na base de dados da UFAM.
3.3. O administrador pode:

- **Aprovar** o cadastro → O usuário recebe um e-mail de confirmação e pode acessar o sistema.
- **Reprovar** o cadastro → O usuário recebe um e-mail informando a reprovação e o motivo.

4. Login e Solicitação de Reservas

4.1. O usuário aprovado acessa o sistema com e-mail e senha.
4.2. No painel, ele pode visualizar as reservas disponíveis.
4.3. O usuário preenche um formulário de solicitação de reserva, informando:

- Data e horário desejados
- Local da reserva

- Motivo da reserva
  4.4. O sistema verifica se há disponibilidade no calendário.
- **Se houver**, a solicitação é enviada para aprovação do administrador.
- **Se não houver**, exibe um aviso informando a indisponibilidade.

5. Aprovação/Rejeição de Reservas pelo Administrador

5.1. O administrador acessa o painel de reservas pendentes.
5.2. Avalia a solicitação e pode:

- **Aprovar** → A reserva é confirmada, e o usuário recebe um e-mail de confirmação.
- **Rejeitar** → O usuário recebe um e-mail informando a rejeição e o motivo.

6. Atualização do Calendário

6.1. As reservas **aprovadas** aparecem no calendário do sistema.
6.2. As datas e horários reservados ficam bloqueados para outros usuários.

6. Diagramas

Diagrama do banco de dados

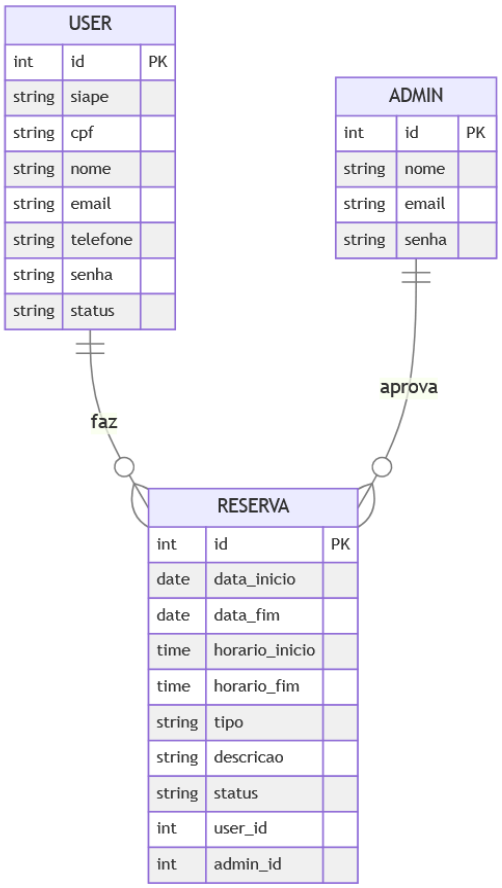


Diagrama de classe:

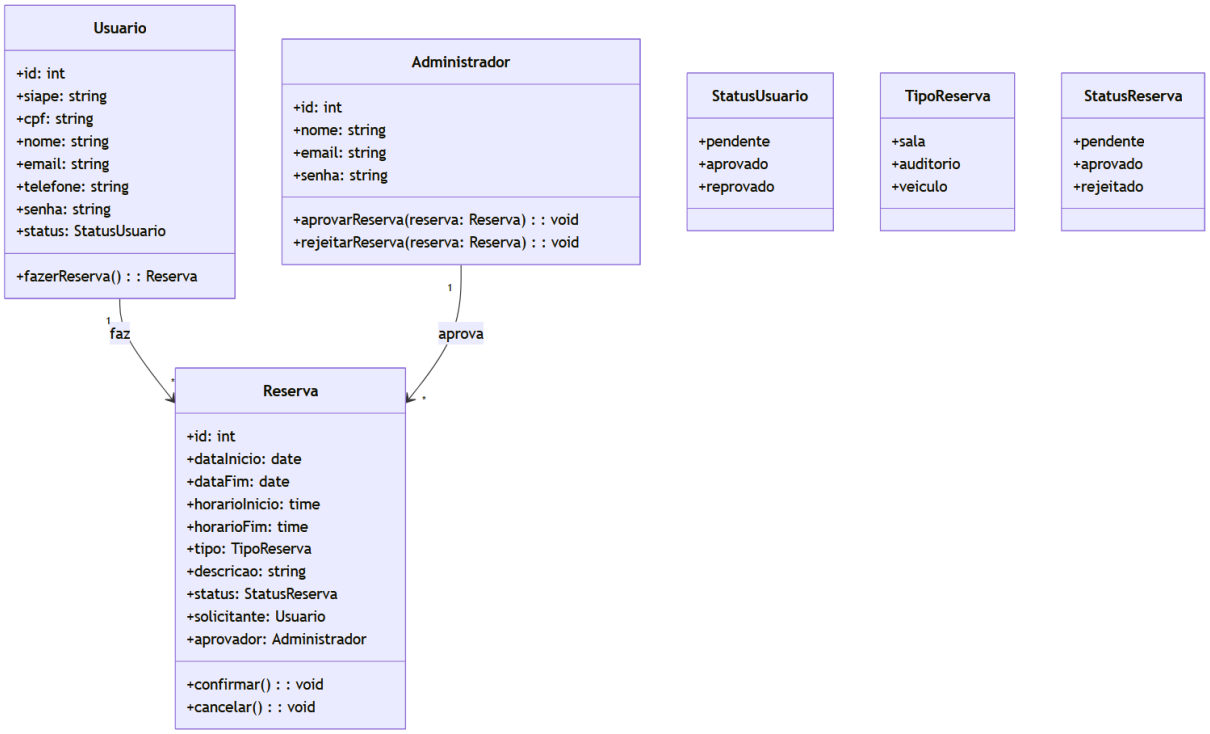
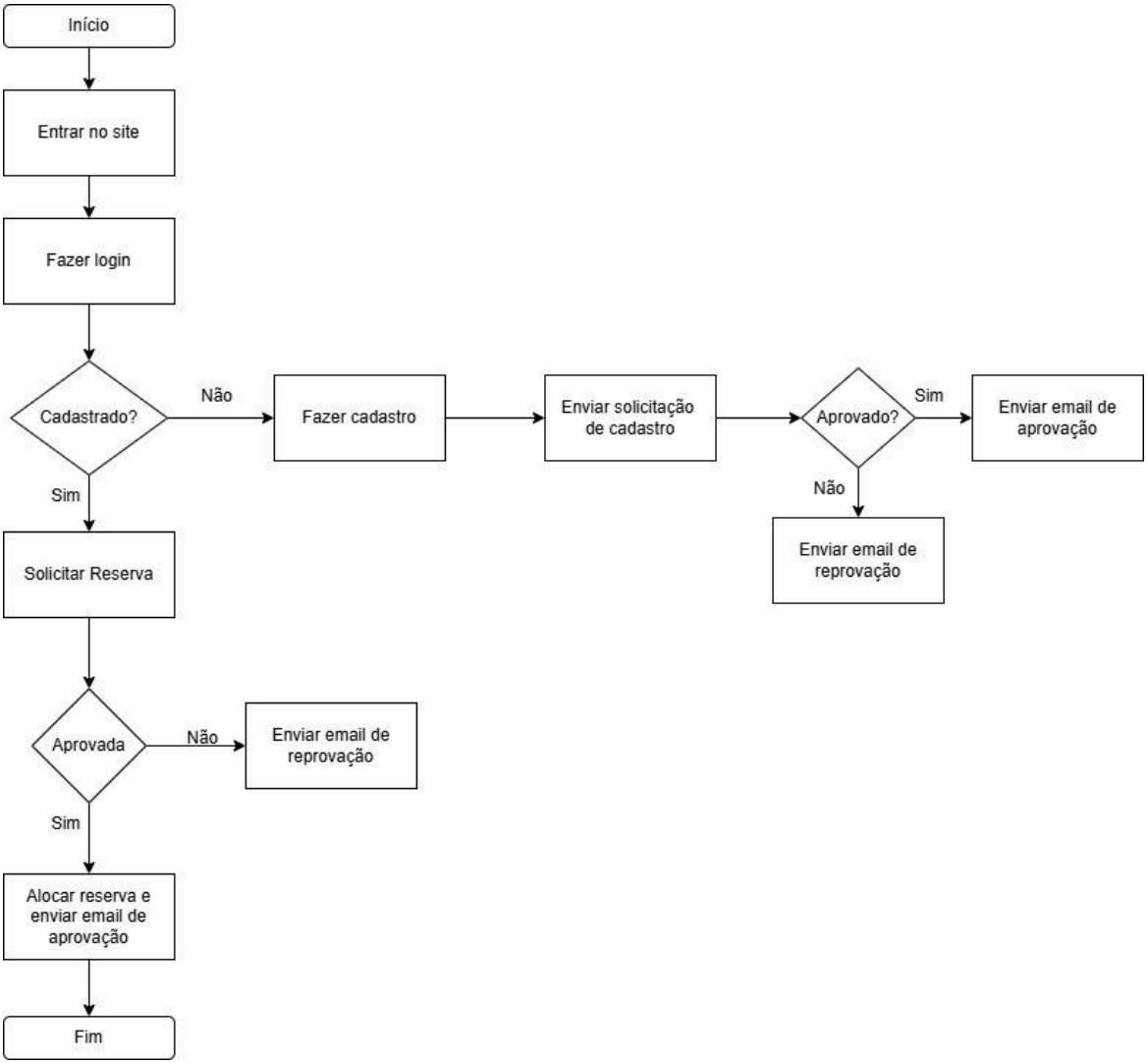
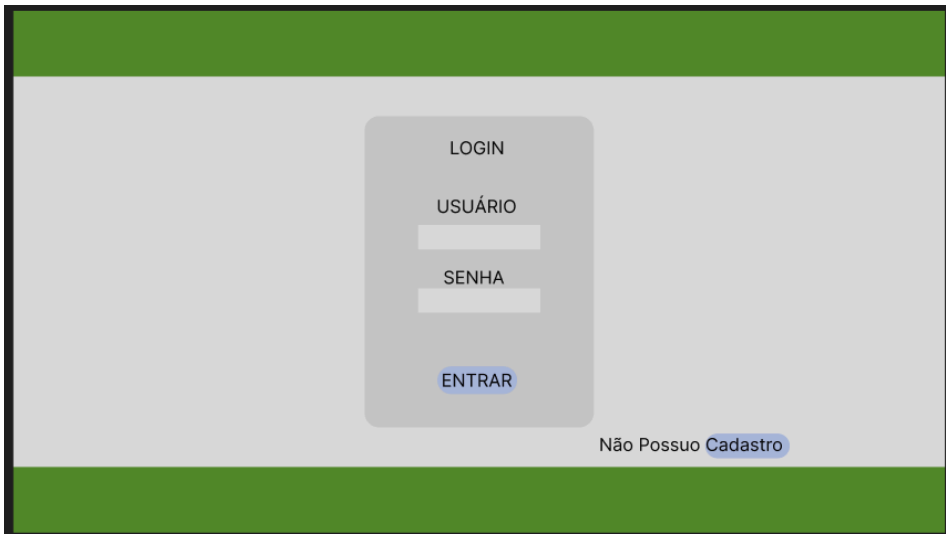


Diagrama de Sequência:



7. Telas


LOGIN
USUÁRIO
SENHA
ENTRAR
Não Possuo Cadastro


CADASTRO
NOME
EMAIL
SIAPE
TELEFONE
CPF
SENHA
CANCEL
SALVAR