Trabalho Avançado – Desenvolvimento de API REST para Gestão de Eventos com Inscrições e Pagamentos

Objetivo O objetivo deste trabalho é desenvolver uma API RESTful utilizando Django e Django REST Framework. Você irá criar um sistema capaz de gerenciar eventos, inscrições, pagamentos e notificações, exercitando conceitos avançados de backend, autenticação, permissões, relacionamento entre modelos e endpoints customizados.

Descrição do Sistema O sistema deverá conter:

1. Usuários - Perfis: - Organizador: cria eventos, gerencia inscrições e recebe relatórios. - Participante: pode se inscrever em eventos, pagar e receber notificações. - Campos: nome completo, email, senha, telefone (opcional), tipo de usuário.

2. Eventos - Cada evento deve ter: título, descrição, data e hora de início/fim, local, número máximo de participantes e preço. - Apenas organizadores podem criar ou editar eventos. - Deve ser possível listar eventos passados e futuros.

3. Inscrições - Um participante pode se inscrever em vários eventos. - Um evento pode ter vários participantes. - Cada inscrição deve armazenar: status (pendente, confirmada, cancelada), data de inscrição e valor pago. - Limite de vagas: não permitir inscrições além do máximo definido. - Apenas inscrições pagas ou confirmadas permitem acesso aos recursos do evento.

4. Pagamentos - Cada inscrição pode ter um pagamento vinculado. - Campos: valor, método (cartão, pix, boleto), data de pagamento e status. - Apenas inscrições pendentes podem gerar pagamento. - Endpoint para confirmar pagamento.

5. Notificações - Endpoint para gerar notificações para participantes, por exemplo: - “Seu pagamento foi confirmado” - “O evento X começa amanhã” - As notificações devem ser registradas no banco, não precisam ser enviadas de fato.

6. Endpoints REST Recomendados | Endpoint | Função | |----------|--------| | /users/me/ | Retorna perfil do usuário logado | | /events/ | Listar/criar eventos (somente organizadores podem criar) | | /events/<id>/register/ | Permite que um participante se inscreva no evento | | /events/<id>/participants/ | Lista participantes do evento (somente organizadores) | | /registrations/<id>/pay/ | Endpoint para registrar pagamento de inscrição | | /events/upcoming/ | Retorna eventos futuros | | /notifications/ | Lista notificações de um usuário | | /reports/events/ | Relatório de eventos com número de inscritos e receita |

7. Regras e Permissões - Organizadores só podem alterar seus próprios eventos. - Participantes só podem visualizar eventos e se inscrever. - Apenas usuários autenticados podem interagir com a API. - Inscrições não podem ser duplicadas. - Pagamentos só podem ser confirmados se a inscrição estiver pendente.

8. Funcionalidades Extras (Opcional) - Rankings de eventos por número de participantes ou receita. - Filtro de eventos por data, local ou preço. - Endpoint para histórico completo do participante (eventos passados e pagos). - Endpoint para estatísticas: receita total, eventos com mais inscrições, média de participantes por evento.

Dificuldades Técnicas - Relacionamentos avançados entre modelos (ForeignKey, ManyToMany, OneToOne). - Serializers aninhados e customizados. - Permissões personalizadas no DRF. - Validações complexas (limite de vagas, status de pagamento, cancelamento de inscrições). - Endpoints agregados e relatórios usando annotate e aggregate.

Instruções para Desenvolvimento 1. Crie a aplicação somente em Django e Django REST Framework, sem front-end. 2. Organize o projeto com apps separados para usuários, eventos, inscrições, pagamentos e notificações. 3. Garanta que a API seja testável via Postman ou Insomnia. 4. Documente todos os endpoints, incluindo: - Método HTTP - URL - Parâmetros esperados - Exemplo de resposta 5. Faça commits regulares usando Git para acompanhar a evolução do projeto.

Critérios de Avaliação | Item | Peso | |------|------| | Estrutura de models e relacionamentos | 25% | | Endpoints e lógica REST | 25% | | Autenticação e permissões | 20% | | Funcionalidades extras e custom endpoints | 20% | | Organização do projeto e documentação | 10% |

Observações Finais - Planeje a estrutura de modelos antes de criar endpoints. - A documentação dos endpoints é tão importante quanto a implementação. - Este projeto é ideal para consolidar conceitos avançados de backend e API REST.