5422 - Interação de sistemas de informação - ferramentas

Contrarian Report - Análise do Projeto Django

Trabalho realizado por: António Morais

Introdução

"Contrarian Report" é uma aplicação web desenvolvida em Django que implementa uma plataforma de conteúdo baseada em subscrições. O projeto segue um padrão de arquitetura multi-aplicação típico de projetos Django e foi concebido com dois papéis de utilizador distintos: clientes (leitores) e escritores. A aplicação apresenta um modelo de conteúdo premium onde os utilizadores podem subscrever para aceder a conteúdo exclusivo, com a gestão de subscrições realizada através da integração com o PayPal. O projeto utiliza extensivamente as capacidades assíncronas do Django, indicando uma ênfase na performance e responsividade.

Desenvolvimento

Estrutura do Projeto

O projeto segue a arquitetura baseada em aplicações recomendada pelo Django, com vários componentes principais:

1. **Pasta do Projeto Principal (projetopython)**
    - Contém ficheiros de configuração principais como settings.py, urls.py, asgi.py e wsgi.py
    - Gere URLs globais, configurações e inicialização da aplicação
    - Utiliza o manage.py para tarefas administrativas e inicialização do servidor
2. **Aplicações**
    - account: Gere a autenticação e gestão de utilizadores (modelo CustomUser)

- client: Gere funcionalidades específicas para subscritores
- writer: Gere a criação e gestão de conteúdo para escritores
- admin: Gere as subscrições e pode criar, alterar informações dos users e clientes.
- common: Contém utilitários partilhados e decoradores entre aplicações

3. **Sistema de Autenticação**
    - Utiliza um modelo de utilizador personalizado (CustomUser) que inclui uma flag is_writer para distinguir entre escritores e clientes
    - Implementa controlo de acesso baseado em funções através de decoradores personalizados
4. **Configuração de Ambiente**
    - Utiliza o ficheiro. env para armazenar informações sensíveis como credenciais do PayPal
    - Integra a biblioteca decouple para aceder a variáveis de ambiente de forma segura
5. **Base de Dados**
    - Utiliza SQLite com migrações Django para gestão do esquema de dados

Funcionalidades Principais

Gestão de Utilizadores

O sistema suporta dois papéis de utilizador distintos:

- **Clientes (Leitores)**: Utilizadores que podem navegar por artigos e subscrever conteúdo premium
- **Escritores**: Utilizadores que podem criar, editar e eliminar artigos

Decoradores de autenticação em common/auth.py impõem permissões baseadas em funções:

- aclient_required: Garante que apenas clientes podem aceder a determinadas vistas
- awriter_required: Restringe o acesso a funcionalidades específicas para escritores
- ensure_for_current_user: Valida a propriedade do objeto antes de permitir operações

Sistema de Subscrição

Implementado na aplicação client com estes componentes:

- Modelo Subscription: Acompanha subscrições ativas para utilizadores
- Modelo PlanChoices: Define planos de subscrição disponíveis
- Integração com PayPal para processamento de pagamentos via paypal.py

O fluxo de subscrição inclui:

1. Navegação pelos planos disponíveis
2. Criação de uma subscrição via PayPal
3. Gestão e cancelamento de subscrições quando necessário

As credenciais do PayPal são armazenadas no ficheiro. env.

Gestão de Conteúdo

A aplicação writer gere a criação e gestão de conteúdo:

- Modelo Article: Armazena conteúdo de artigos com flag premium
- Operações CRUD para artigos (criar, ler, atualizar, eliminar)
- Marcação de conteúdo premium para restringir acesso apenas a subscritores

Características Técnicas

1. **Processamento Assíncrono**
    - Uso extensivo da sintaxe async/await em todas as vistas
    - Invólucros assíncronos personalizados para formulários Django e operações de modelo (AsyncFormMixin, AsyncModelFormMixin)
    - Renderização assíncrona com o utilitário arender
2. **Integração com PayPal**
    - Conexão API para gestão de subscrições
    - Fluxo de autenticação para proteger chamadas à API do PayPal
    - Configuração de variáveis de ambiente via decouple
3. **Frontend**
    - Integração Bootstrap via crispy_forms e crispy_bootstrap5 para estilização de formulários
    - Estrutura de templates organizada por aplicação com templates partilhados em common/templates

Fluxo da Aplicação

1. **Fluxo de Autenticação**
    - Os utilizadores registam-se/iniciam sessão e são direcionados para os painéis apropriados com base na função
    - Utilizadores anónimos são redirecionados para o login quando tentam aceder a vistas protegidas
2. **Fluxo de Utilizador Cliente**
    - Vista de painel mostra o estado da subscrição
    - Navegação por artigos com acesso filtrado com base no nível de subscrição

    - Subscrição de planos e gestão do estado da subscrição
    - Atualização de informações pessoais
3. **Fluxo de Utilizador Escritor**
    - Painel para gestão de artigos
    - Criar, editar, eliminar artigos com verificações de propriedade
    - Alternar estado premium para artigos
    - Gestão de conta (atualizar/eliminar)

Conclusão

O projeto Contrarian Report demonstra uma aplicação Django bem estruturada com uma clara separação de preocupações entre a funcionalidade de cliente e escritor. O uso de vistas assíncronas em toda a aplicação indica um foco na performance, particularmente importante para plataformas de entrega de conteúdo.

Os principais pontos fortes da implementação incluem:

- Controlo de acesso baseado em funções com decoradores personalizados
- Integração com PayPal para gestão de subscrições
- Clara separação entre conteúdo premium e standard
- Operações CRUD abrangentes para ambos os tipos de utilizador

O projeto implementa com sucesso uma plataforma de conteúdo baseada em subscrições com papéis de utilizador distintos e controlo de acesso a conteúdo premium. A arquitetura segue as melhores práticas do Django com aplicações modulares e uma clara separação de preocupações, tornando-a manutenível e extensível para futuras adições de funcionalidades.