Relatório da API e Cliente

Nome: Elke Macline de Oliveira Sabino

Disciplina: Tópicos Especiais de Programação

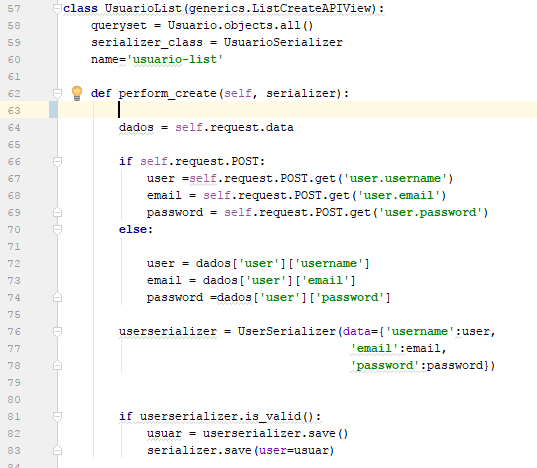
**1 - A API deve possuir pelo menos 4 entidades relevantes e relacionadas via mapeamento objeto relacional**

R: As entidades usadas no projeto foram: Usuario,Autor,Livro,Estante e Operacao. O sistema baseia-se em troca e compra de livros entre usuários de diversas regiões.

**2 - Pelo menos uma entidade deve ser integrada ao esquema de autenticação do Django;**

R: Na api, a entidade Usuário está integrada com o User do Django, ao cadastrar um usuário, durante o cadastro são realizadas duas ações é criado um User no Django e após isso, é criado o usuário inserindo uma chave estrangeira OneToOneFiel. No arquivo Models pode ser verificado e a implementação do cadastro está na Views

Views linha 57:



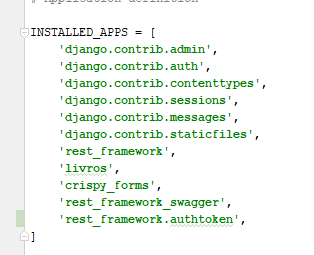
**3 - Parte da API deve ser somente leitura e parte deve ser acessível apenas para usuários autenticados.**

R: Esse requisito também foi realizado, algumas ações como cadastramento de livros, cadastramento de autores, efetuação de transações entre usuários, verificação das transações realizadas, só ocorrem se o usuário estiver autenticado

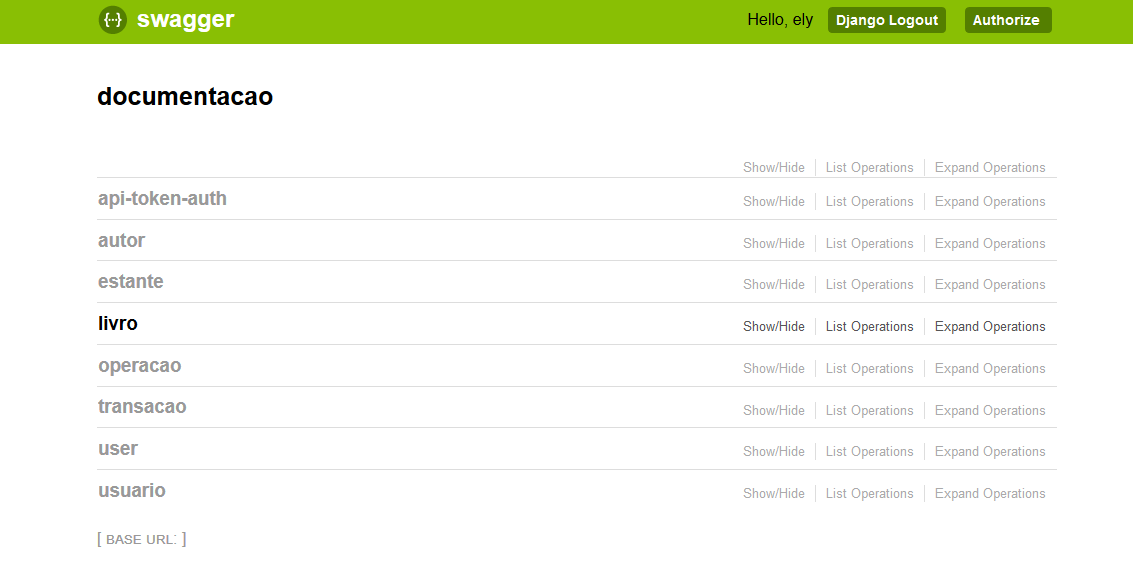
**4 - A API deve ser documentada com Swagger ou alguma outra sugestão da página: http://www.django-rest-framework.org/topics/documenting-your-api**

R: A documentação foi feita usando o Swagger, abaixo alguns prints do que fora feito:

No settings em APPs:

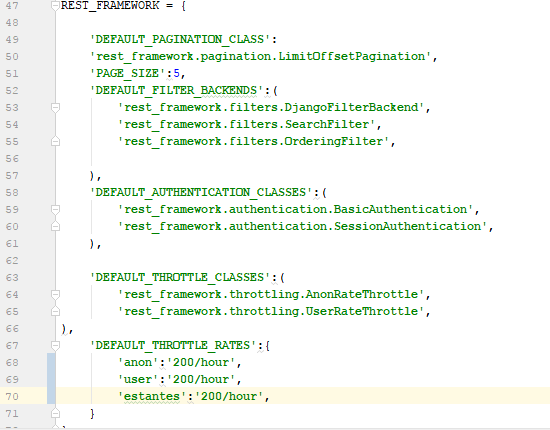


Resultado na API:

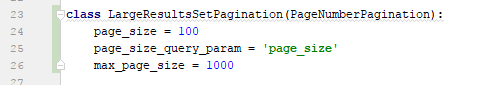


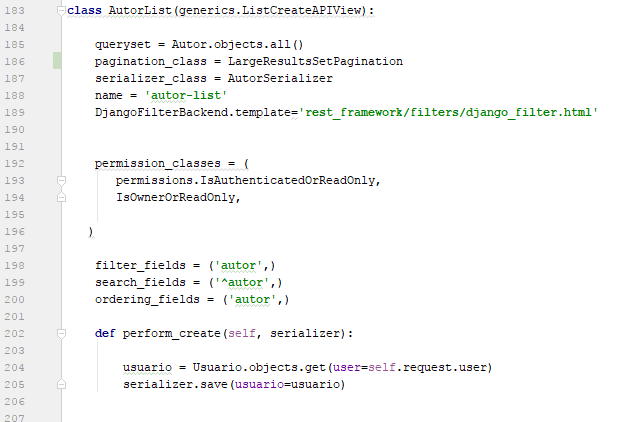
**5 - Definir e usar critérios de paginação e Throttling. Esse último deve diferenciar usuários autenticados de não autenticados**

R: Este requisito também foi realizado, em settings na linha 63 podemos ver as configurações realizadas.



Na views, também foi implementado de forma personalizada a paginação para classe de autores.



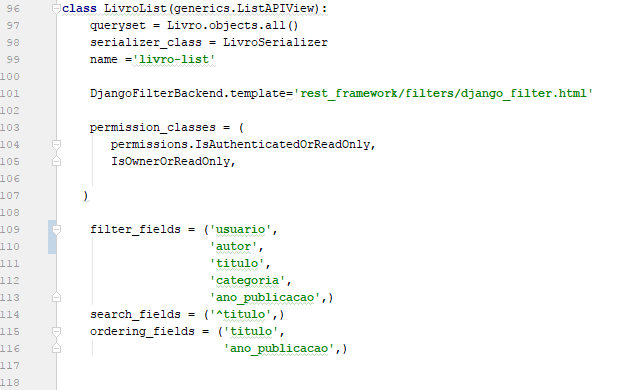


**6 - Implementar para pelo menos 2 entidades: filtros, busca e ordenação**

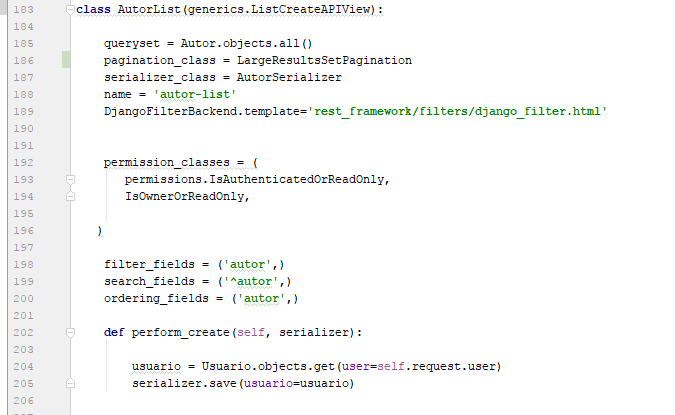
R: Esse requisito foi realizado nas classes de Livros e Autores, onde podem ser feitas buscas, ordenação e filtrar.

Foi utilizado a importação **from** rest\_framework.filters **import** DjangoFilterBackend

Para a utilização dos filtros, na views pode ser constatado na linha 109, os campos escolhidos para filtragem, como mostra a figura abaixo:



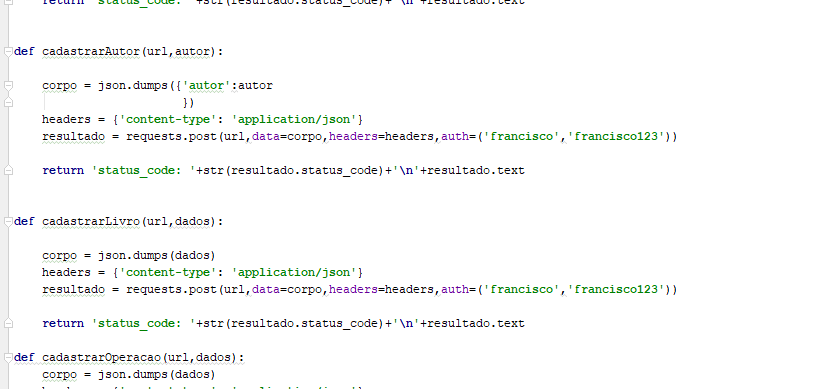
Na classe AutorList também é outro exemplo onde o filtro foi usado, como mostra abaixo



**OBS: No lado Cliente, feito em java, quando é aberto o formulário de cadastro de livros, o combobox com os nomes dos autores foi também usado o recurso do filtro através de uma requisição para que o mesmo recebesse os autores em ordem alfabética**

**7 - Criar testes unitários e de cobertura;**

R: Os testes que foram feitos foram apenas usando um arquivo externo com a biblioteca Requests, para fazer as requisições e verificar se tudo estava ocorrendo bem, como mostra abaixo uma parte do arquivo em imagem, sendo que o arquivo original com as requisições está no repositório com a API e o cliente



**Observações finais do relatório:**

**O cliente foi feito usando linguagem java com algumas bibliotecas importantes como** Jersey e GSON, **foram criados os formulários usando o swing.**

**No cliente é possível fazer as seguintes operações:**

* **Efetuar o cadastro de um novo usuário**
* **Efetuar um login, permanecendo conectado, como uma sessão, foi implementado uma classe sessão em que o cliente se comunica com a API verificando o usuário e senha, confirmando o login no lado cliente é exibido o nome do usuário na tela e dessa forma o botão do LOGOF fica ativado para quando o usuário quiser encerrar a sessão.**
* **Pode ser realizado uma consulta verificando todos os livros cadastrados no sistema**
* **Pode ser realizado uma consulta verificando autores**
* **Pode ser realizado cadastros de novos livros, sendo exigido a autenticação, caso contrário retorna uma mensagem.**
* **É possível verificar os livros do usuário que está logado, que o mesmo cadastrou.**
* **É possível também excluir livros do usuário logado no sistema**
* **É possível fazer a consulta de todas as estantes no sistema verificando a disponibilidade das mesmas para compra ou troca**
* **É possível efetuar uma transação escolhendo o livro que o usuário logado se interessa, assim a estante deixa de ser listada como disponível**
* **O usuário pode a qualquer tempo se estiver logado verificar as suas transações realizadas.**