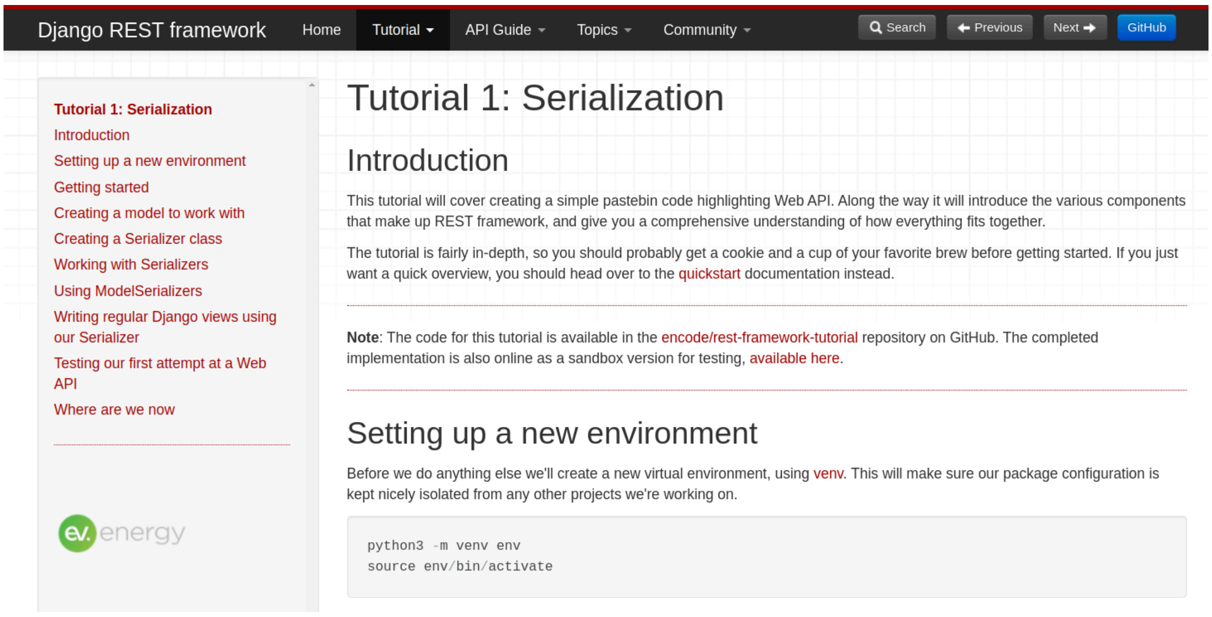
Django RESTful Framework (DRF) Tutorial

- https://github.com/SCKIMOSU/DRF_tutorial-_ch1_Serializer
-
- https://www.django-rest-framework.org/tutorial/1-serialization/
- https://github.com/encode/rest-framework-tutorial
-
- 이 튜토리얼에서는 Web API를 강조하는 간단한 페이스트빈 코드 생성을 다룹니다.
- 그 과정에서 REST 프레임워크를 구성하는 다양한 구성 요소를 소개하고 모든 것이 어떻게 조화를 이루는지에 대한 포괄적인 이해를 제공합니다.


Django REST framework
Home
Tutorial
API Guide
Topics
Community
Search
Previous
Next
GitHub
Tutorial 1: Serialization
Introduction
Setting up a new environment
Getting started
Creating a model to work with
Creating a Serializer class
Working with Serializers
Using ModelSerializers
Writing regular Django views using our Serializer
Testing our first attempt at a Web API
Where are we now
ev.energy

Tutorial 1: Serialization

Introduction

This tutorial will cover creating a simple pastebin code highlighting Web API. Along the way it will introduce the various components that make up REST framework, and give you a comprehensive understanding of how everything fits together.

The tutorial is fairly in-depth, so you should probably get a cookie and a cup of your favorite brew before getting started. If you just want a quick overview, you should head over to the quickstart documentation instead.

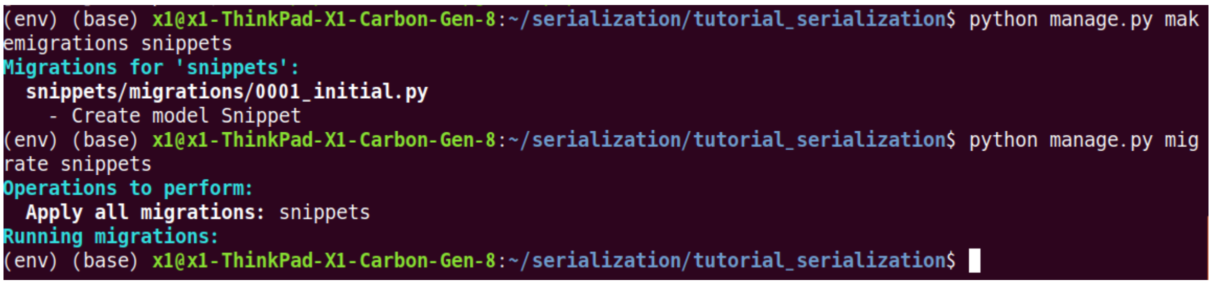
Note: The code for this tutorial is available in the encode/rest-framework-tutorial repository on GitHub. The completed implementation is also online as a sandbox version for testing, available here.

Setting up a new environment

Before we do anything else we'll create a new virtual environment, using venv. This will make sure our package configuration is kept nicely isolated from any other projects we're working on.

```
python3 -m venv env
source env/bin/activate
```

1. Serializer 구현



```
(env) (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/serialization/tutorial_serialization$ python manage.py mak
emigrations snippets
Migrations for 'snippets':
  snippets/migrations/0001_initial.py
    - Create model Snippet
(env) (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/serialization/tutorial_serialization$ python manage.py mig
rate snippets
Operations to perform:
  Apply all migrations: snippets
Running migrations:
(env) (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/serialization/tutorial_serialization$ 
```

- 웹 API를 시작하기 위해 가장 먼저 해야 할 일은 snippet 인스턴스를 json과 같은 표현으로 직렬화 및 역직렬화하는 방법을 제공하는 것입니다. Django의 형식과 매우 유사하게 작동하는 직렬 변환기를 선언하여 이를 수행할 수 있습니다. serializers.py라는 snippets 디렉터리에 파일을 만들고 다음을 추가합니다.

```
from rest_framework import serializers
from snippets.models import Snippet, LANGUAGE_CHOICES, STYLE_CHOICES
```

```
class SnippetSerializer(serializers.Serializer):
    id = serializers.IntegerField(read_only=True)
    title = serializers.CharField(required=False, allow_blank=True,
max_length=100)
    code = serializers.CharField(style={'base_template': 'textarea.html'})
    linenos = serializers.BooleanField(required=False)
    language = serializers.ChoiceField(choices=LANGUAGE_CHOICES,
default='python')
    style = serializers.ChoiceField(choices=STYLE_CHOICES,
default='friendly')

    def create(self, validated_data):
        """
        Create and return a new `Snippet` instance, given the validated
data.
        """
        return Snippet.objects.create(**validated_data)

    def update(self, instance, validated_data):
        """
        Update and return an existing `Snippet` instance, given the
validated data.
        """
        instance.title = validated_data.get('title', instance.title)
        instance.code = validated_data.get('code', instance.code)
        instance.linenos = validated_data.get('linenos', instance.linenos)
        instance.language = validated_data.get('language',
instance.language)
        instance.style = validated_data.get('style', instance.style)
        instance.save()
        return instance
```

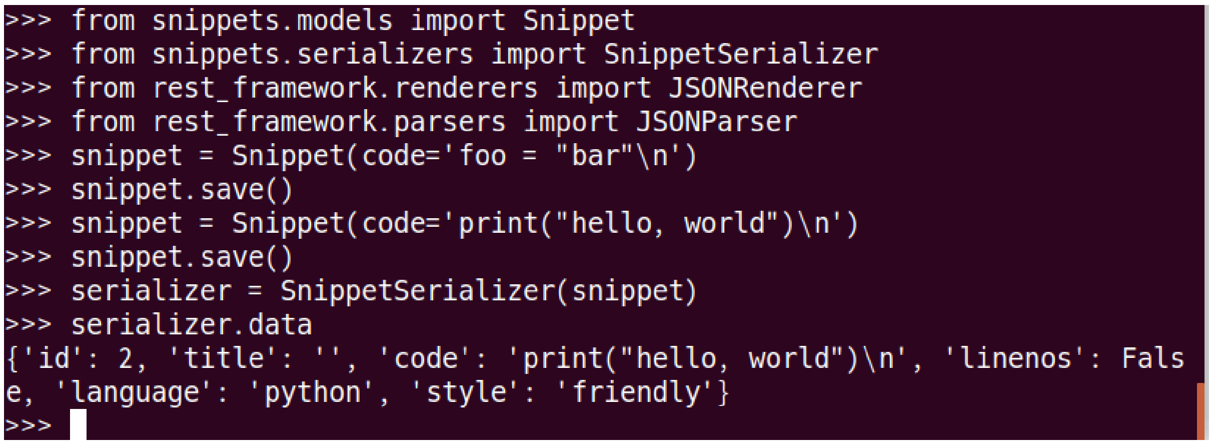
- serializer 클래스의 첫 번째 부분은 직렬화/역직렬화되는 필드를 정의합니다. create() 및 update() 메소드는 serializer.save()를 호출할 때 완전한 기능을 갖춘 인스턴스가 생성되거나 수정되는 방법을 정의합니다.

2. Serializer와 DeSerializer

```
python manage.py shell
```

```
from snippets.models import Snippet
from snippets.serializers import SnippetSerializer
from rest_framework.renderers import JSONRenderer
from rest_framework.parsers import JSONParser

snippet = Snippet(code='foo = "bar"\n')
snippet.save()

snippet = Snippet(code='print("hello, world")\n')
```

```
snippet.save()
```



```
>>> from snippets.models import Snippet
>>> from snippets.serializers import SnippetSerializer
>>> from rest_framework.renderers import JSONRenderer
>>> from rest_framework.parsers import JSONParser
>>> snippet = Snippet(code='foo = "bar"\n')
>>> snippet.save()
>>> snippet = Snippet(code='print("hello, world")\n')
>>> snippet.save()
>>> serializer = SnippetSerializer(snippet)
>>> serializer.data
{'id': 2, 'title': '', 'code': 'print("hello, world")\n', 'linenos': Fals
e, 'language': 'python', 'style': 'friendly'}
>>> 
```

```
>>> from snippets.models import Snippet
>>> from snippets.serializers import SnippetSerializer
>>> from rest_framework.renderers import JSONRenderer
>>> from rest_framework.parsers import JSONParser
>>> snippet = Snippet(code='foo = "bar"\n')
>>> snippet.save()
>>> snippet = Snippet(code='print("hello, world")\n')
>>> snippet.save()
>>> serializer = SnippetSerializer(snippet)
>>> serializer.data
{'id': 2, 'title': '', 'code': 'print("hello, world")\n', 'linenos': False, 'language': 'python', 'style':
'friendly'}
```

a. Serializer Class 는 db python 데이터를 client json 데이터로 변경해 주는 Serializer와 client json 데이터 db python 데이터로 변경해 주는 DeSerializer 로 구성됨
b. 여기서는 Serializer 기능 구현
c. python data : db에 저장되어 있는 데이터,
    i. serializer는 python data를 json data로 바꾸는 작업을 진행
    ii. 모델 인스턴스를 Python 기본 데이터 유형으로 변환.

We've now got a few snippet instances to play with. Let's take a look at serializing one of those instances.

```
serializer = SnippetSerializer(snippet)
serializer.data
# {'id': 2, 'title': '', 'code': 'print("hello, world")\n', 'linenos': False, 'language': 'pyt
```

    iii. python data : db에 저장되어 있는 데이터,

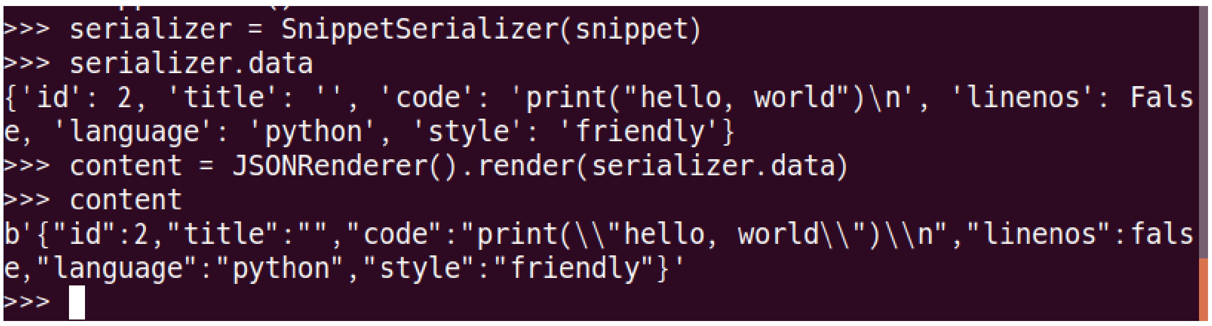
```
>>> serializer = SnippetSerializer(snippet)
>>> serializer.data
{'id': 2, 'title': '', 'code': 'print("hello, world")\n', 'linenos': False, 'language': 'python', 'style':
'friendly'}
```

iv. JSON 데이터 : client 로 보내지는 json 데이터
    1. serializer는 python data를 json data로 바꾸는 작업을 진행
    2. 직렬화 프로세스를 마무리하기 위해 python 데이터를 json으로 렌더링합니다.
    3. 아래와 같은 작업이 Serializer 에 구현되어 있음

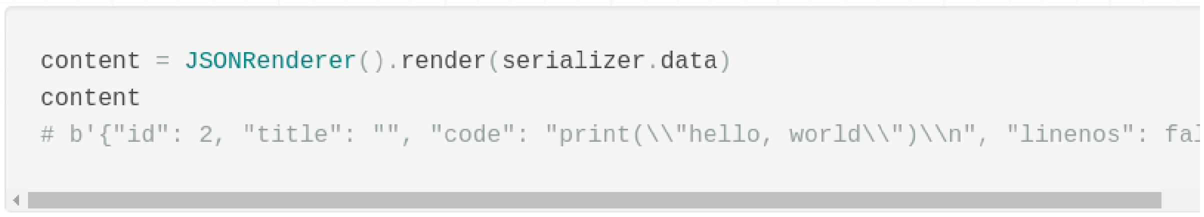
```
content = JSONRenderer().render(serializer.data)
content
# b'{"id": 2, "title": "", "code": "print(\\"hello, world\\")\\n", "linenos": false, "language":
"python", "style": "friendly"}
```

```
>>> content = JSONRenderer().render(serializer.data)
>>> content
b'{"id":2,"title":"","code":"print(\\"hello,
world\\")\\n","linenos":false,"language":"python","style":"friendly"}'
```

v. 아래 content는 json 데이터 타입이다
    1. client 에 도달하는 데이터 타입이다

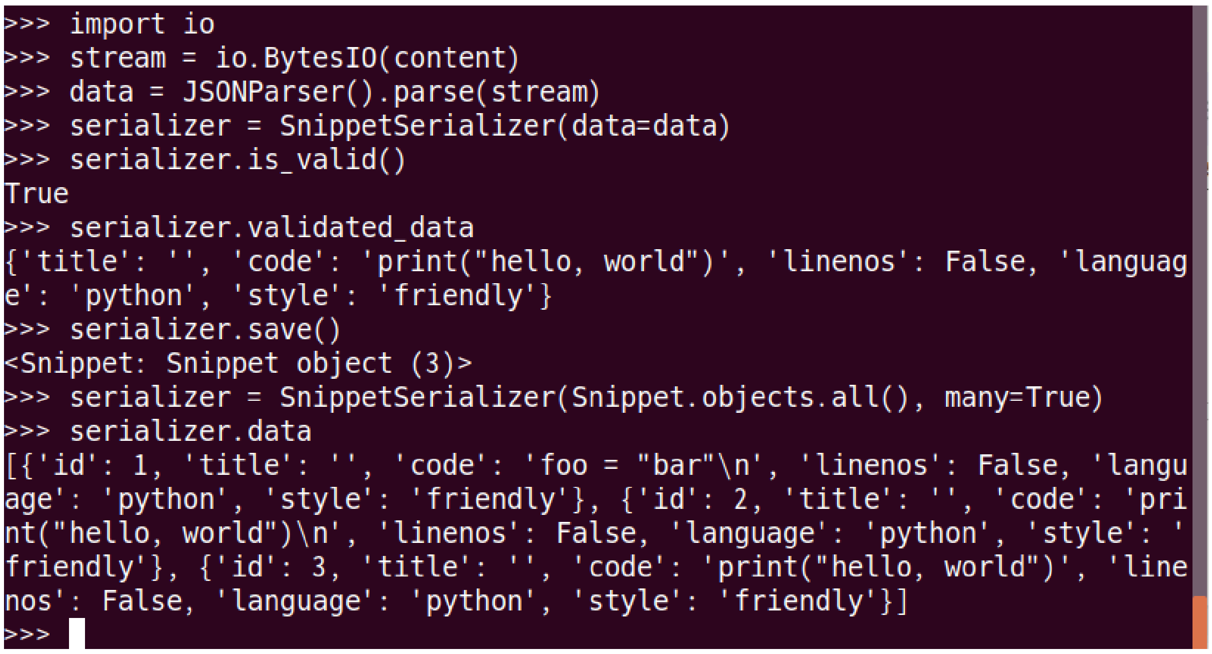


```
>>> serializer = SnippetSerializer(snippet)
>>> serializer.data
{'id': 2, 'title': '', 'code': 'print("hello, world")\n', 'linenos': Fals
e, 'language': 'python', 'style': 'friendly'}
>>> content = JSONRenderer().render(serializer.data)
>>> content
b'{"id":2,"title":"","code":"print(\\"hello, world\\")\\n","linenos":fals
e,"language":"python","style":"friendly"}'
>>> 
```



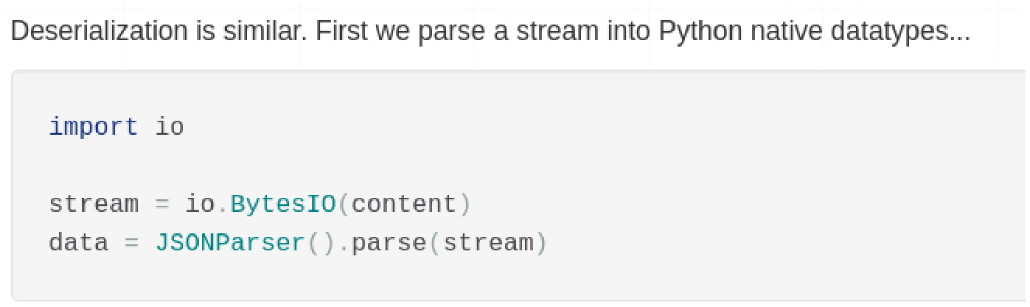
```
content = JSONRenderer().render(serializer.data)
content
# b'{"id": 2, "title": "", "code": "print(\\"hello, world\\")\\n", "linenos": fal
```

3. DeSerializer 구현



```
>>> import io
>>> stream = io.BytesIO(content)
>>> data = JSONParser().parse(stream)
>>> serializer = SnippetSerializer(data=data)
>>> serializer.is_valid()
True
>>> serializer.validated_data
{'title': '', 'code': 'print("hello, world")', 'linenos': False, 'languag
e': 'python', 'style': 'friendly'}
>>> serializer.save()
<Snippet: Snippet object (3)>
>>> serializer = SnippetSerializer(Snippet.objects.all(), many=True)
>>> serializer.data
[{'id': 1, 'title': '', 'code': 'foo = "bar"\n', 'linenos': False, 'langu
age': 'python', 'style': 'friendly'}, {'id': 2, 'title': '', 'code': 'pri
nt("hello, world")\n', 'linenos': False, 'language': 'python', 'style': '
friendly'}, {'id': 3, 'title': '', 'code': 'print("hello, world")', 'line
nos': False, 'language': 'python', 'style': 'friendly'}]
>>> 
```

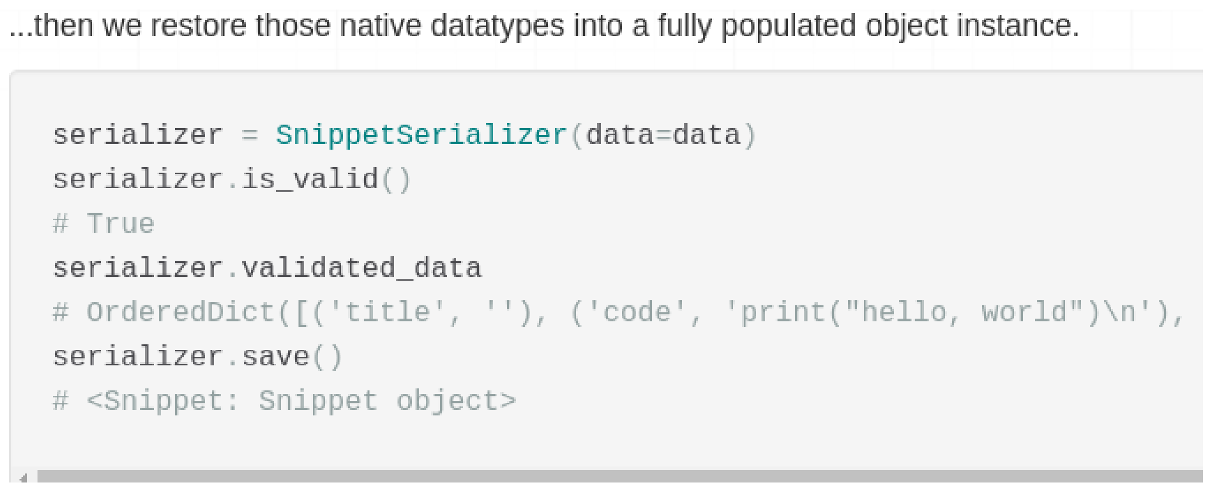
- 클라이언트의 content json 타입을 stream json으로 바꾸고, stream json 타입을 parse 하여 json 타입 data 를 만든다



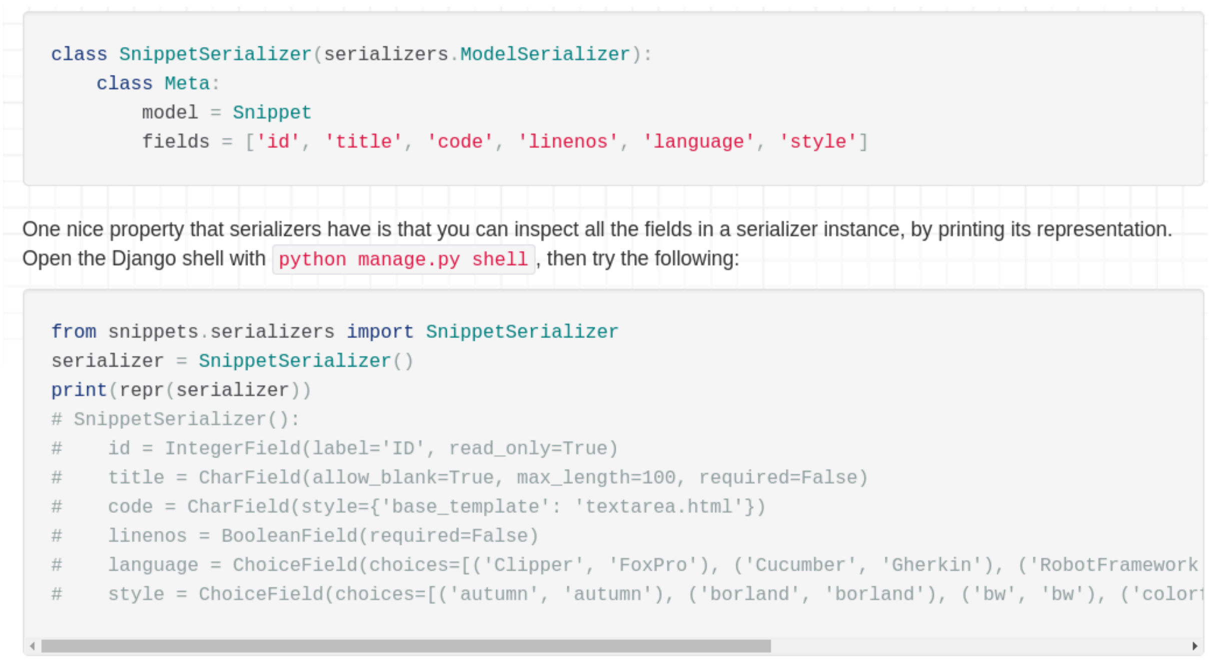
Deserialization is similar. First we parse a stream into Python native datatypes...

```
import io

stream = io.BytesIO(content)
data = JSONParser().parse(stream)
```

d. 그리고 json 데이터 타입을 SnippetSerializer를 통해 serializer 객체를 구현함.
   i. SnippetSerializer는 DeSerializer 구현 기능을 가지고 있다. client json 데이터를 db python 데이터로 변경해 주는 DeSerializer
   ii. serializer는 db python 타입이며, serializer.validated_data를 거쳐 즉, db python 타입이 확실한 지 확인하고, db python 타입인 serializer를 serializer.save()로 저장함


...then we restore those native datatypes into a fully populated object instance.

```
serializer = SnippetSerializer(data=data)
serializer.is_valid()
# True
serializer.validated_data
# OrderedDict([('title', ''), ('code', 'print("hello, world")\n'),
serializer.save()
# <Snippet: Snippet object>
```


```
class SnippetSerializer(serializers.ModelSerializer):
    class Meta:
        model = Snippet
        fields = ['id', 'title', 'code', 'linenos', 'language', 'style']
```

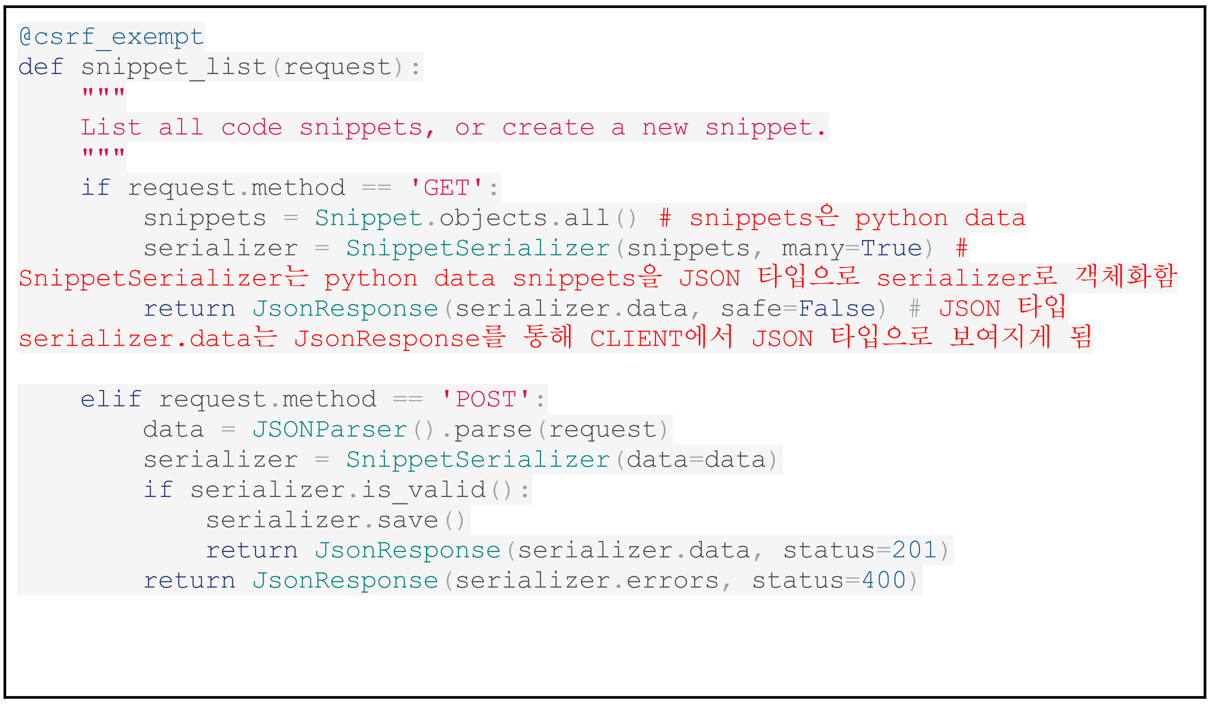
One nice property that serializers have is that you can inspect all the fields in a serializer instance, by printing its representation. Open the Django shell with `python manage.py shell`, then try the following:

```
from snippets.serializers import SnippetSerializer
serializer = SnippetSerializer()
print(repr(serializer))
# SnippetSerializer():
#    id = IntegerField(label='ID', read_only=True)
#    title = CharField(allow_blank=True, max_length=100, required=False)
#    code = CharField(style={'base_template': 'textarea.html'})
#    linenos = BooleanField(required=False)
#    language = ChoiceField(choices=[('Clipper', 'FoxPro'), ('Cucumber', 'Gherkin'), ('RobotFramework
#    style = ChoiceField(choices=[('autumn', 'autumn'), ('borland', 'borland'), ('bw', 'bw'), ('colorf
```

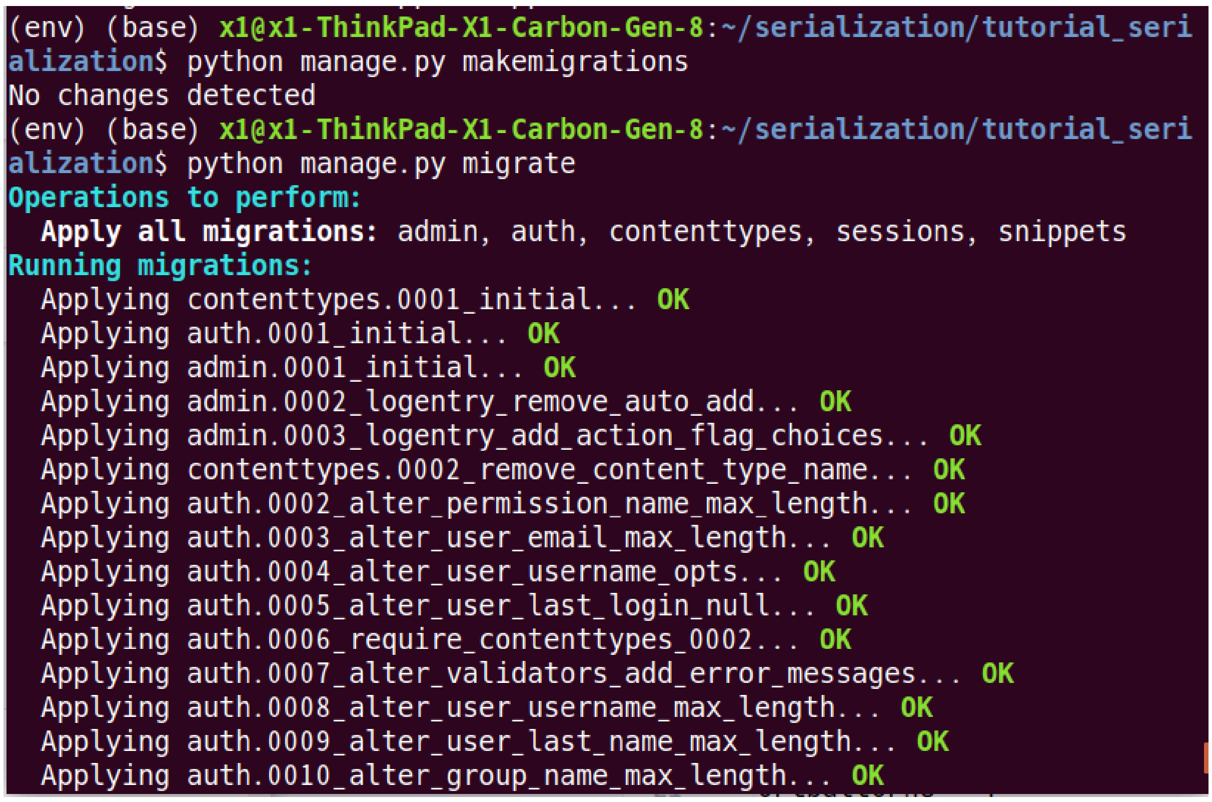
```
>>> print(repr(serializer))
SnippetSerializer():
   id = IntegerField(read_only=True)
   title = CharField(allow_blank=True, max_length=100, required=False)
   code = CharField(style={'base_template': 'textarea.html'})
   linenos = BooleanField(required=False)
   language = ChoiceField(choices=[('abap', 'ABAP'), ('abnf', 'ABNF'), ('actionscript',
'ActionScript'), ('actionscript3', 'ActionScript 3'), ('ada', 'Ada'), ('adl', 'ADL'), ('agda', 'Agda'),
('aheui', 'Aheui'), ('alloy', 'Alloy'), ('ambienttalk', 'AmbientTalk'), ('amdgpu', 'AMDGPU'),
('ampl', 'Ampl'), ('androidbp', 'Soong'), ('ansys', 'ANSYS parametric design language'),
('antlr', 'ANTLR'), ('antlr-actionscript', 'ANTLR With ActionScript Target'), ('antlr-cpp',
'ANTLR With CPP Target'), ('antlr-csharp', 'ANTLR With C# Target'), ('antlr-java', 'ANTLR
With Java Target'), ('antlr-objc', 'ANTLR With ObjectiveC Target'), ('antlr-perl', 'ANTLR With
Perl Target'), ('antlr-python', 'ANTLR With Python Target'), ('antlr-ruby', 'ANTLR With Ruby
Target'), ('apacheconf', 'ApacheConf'), ('apl', 'APL'), ('applescript', 'AppleScript'), ('arduino',
'Arduino'), ('arrow', 'Arrow'), ('arturo', 'Arturo'), ('asc', 'ASCII armored'), ('asn1', 'ASN.1'),
('aspectj', 'AspectJ'), ('aspx-cs', 'aspx-cs'), ('aspx-vb', 'aspx-vb'), ('asymptote', 'Asymptote'),
('augeas', 'Augeas'), ('autohotkey', 'autohotkey'), ('autoit', 'AutoIt'), ('awk', 'Awk'), ('bare',
'BARE'), ('basemake', 'Base Makefile'), ('bash', 'Bash'), ('batch', 'Batchfile'), ('bbcbasic',
'BBC Basic'), ('bbcode', 'BBCode'), ('bc', 'BC'), ('bdd', 'Bdd'), ('befunge', 'Befunge'), ('berry',
'Berry'), ('bibtex', 'BibTeX'), ('blitzbasic', 'BlitzBasic'), ('blitzmax', 'BlitzMax'), ('blueprint',
'Blueprint'), ('bnf', 'BNF'), ('boa', 'Boa'), ('boo', 'Boo'), ('boogie', 'Boogie'), ('bqn', 'BQN'),
('brainfuck', 'Brainfuck'), ('bst', 'BST'), ('bugs', 'BUGS'), ('c', 'C'), ('c-objdump', 'c-objdump'),
('ca65', 'ca65 assembler'), ('cadl', 'cADL'), ('camkes', 'CAmkES'), ('capdl', 'CapDL'),
('capnp', "Cap'n Proto"), ('carbon', 'Carbon'), ('cbmbas', 'CBM BASIC V2'), ('cddl', 'CDDL'),
('ceylon', 'Ceylon'), ('cfc', 'Coldfusion CFC'), ('cfengine3', 'CFEngine3'), ('cfm', 'Coldfusion
HTML'), ('cfs', 'cfstatement'), ('chaiscript', 'ChaiScript'), ('chapel', 'Chapel'), ('charmci',
'Charmci'), ('cheetah', 'Cheetah'), ('cirru', 'Cirru'), ('clay', 'Clay'), ('clean', 'Clean'), ('clojure',
'Clojure'), ('clojurescript', 'ClojureScript'), ('cmake', 'CMake'), ('cobol', 'COBOL'),
('cobolfree', 'COBOLFree'), ('coffeescript', 'CoffeeScript'), ('comal', 'COMAL-80'),
('common-lisp', 'Common Lisp'), ('componentpascal', 'Component Pascal'), ('console',
'Bash Session'), ('coq', 'Coq'), ('cplint', 'cplint'), ('cpp', 'C++'), ('cpp-objdump',
'cpp-objdump'), ('cpsa', 'CPSA'), ('cr', 'Crystal'), ('crmsh', 'Crmsh'), ('croc', 'Croc'), ('cryptol',
'Cryptol'), ('csharp', 'C#'), ('csound', 'Csound Orchestra'), ('csound-document', 'Csound
Document'), ('csound-score', 'Csound Score'), ('css', 'CSS'), ('css+django',
'CSS+Django/Jinja'), ('css+genshitext', 'CSS+Genshi Text'), ('css+lasso', 'CSS+Lasso'),
('css+mako', 'CSS+Mako'), ('css+mozpreproc', 'CSS+mozpreproc'), ('css+myghty',
'CSS+Myghty'), ('css+php', 'CSS+PHP'), ('css+ruby', 'CSS+Ruby'), ('css+smarty',
'CSS+Smarty'), ('css+ul4', 'CSS+UL4'), ('cuda', 'CUDA'), ('cypher', 'Cypher'), ('cython',
'Cython'), ('d', 'D'), ('d-objdump', 'd-objdump'), ('dart', 'Dart'), ('dasm16', 'DASM16'), ('dax',
'Dax'), ('debcontrol', 'Debian Control file'), ('debsources', 'Debian Sourcelist'), ('delphi',
'Delphi'), ('desktop', 'Desktop file'), ('devicetree', 'Devicetree'), ('dg', 'dg'), ('diff', 'Diff'),
('django', 'Django/Jinja'), ('docker', 'Docker'), ('doscon', 'MSDOS Session'), ('dpatch',
'Darcs Patch'), ('dtd', 'DTD'), ('duel', 'Duel'), ('dylan', 'Dylan'), ('dylan-console', 'Dylan
session'), ('dylan-lid', 'DylanLID'), ('earl-grey', 'Earl Grey'), ('easytrieve', 'Easytrieve'),
('ebnf', 'EBNF'), ('ec', 'eC'), ('ecl', 'ECL'), ('eiffel', 'Eiffel'), ('elixir', 'Elixir'), ('elm', 'Elm'), ('elpi',
'Elpi'), ('emacs-lisp', 'EmacsLisp'), ('email', 'E-mail'), ('erb', 'ERB'), ('erl', 'Erlang erl
session'), ('erlang', 'Erlang'), ('evoque', 'Evoque'), ('execline', 'execline'), ('extempore',
'xtlang'), ('ezhil', 'Ezhil'), ('factor', 'Factor'), ('fan', 'Fantom'), ('fancy', 'Fancy'), ('felix', 'Felix'),
('fennel', 'Fennel'), ('fift', 'Fift'), ('fish', 'Fish'), ('flatline', 'Flatline'), ('floscript', 'FloScript'),
('forth', 'Forth'), ('fortran', 'Fortran'), ('fortranfixed', 'FortranFixed'), ('foxpro', 'FoxPro'),
('freefem', 'Freefem'), ('fsharp', 'F#'), ('fstar', 'FStar'), ('func', 'FunC'), ('futhark', 'Futhark'),
('gap', 'GAP'), ('gap-console', 'GAP session'), ('gas', 'GAS'), ('gcode', 'g-code'), ('gdscript',
'GDScript'), ('genshi', 'Genshi'), ('genshitext', 'Genshi Text'), ('gherkin', 'Gherkin'), ('glsl',
'GLSL'), ('gnuplot', 'Gnuplot'), ('go', 'Go'), ('golo', 'Golo'), ('gooddata-cl', 'GoodData-CL'),
('gosu', 'Gosu'), ('graphql', 'GraphQL'), ('graphviz', 'Graphviz'), ('groff', 'Groff'), ('groovy',
'Groovy'), ('gsql', 'GSQL'), ('gst', 'Gosu Template'), ('haml', 'Haml'), ('handlebars',
'Handlebars'), ('haskell', 'Haskell'), ('haxe', 'Haxe'), ('haxeml', 'Hxml'), ('hexdump',
'Hexdump'), ('hlsl', 'HLSL'), ('hsail', 'HSAIL'), ('hspec', 'Hspec'), ('html', 'HTML'),
('html+cheetah', 'HTML+Cheetah'), ('html+django', 'HTML+Django/Jinja'), ('html+evoque',
'HTML+Evoque'), ('html+genshi', 'HTML+Genshi'), ('html+handlebars',
'HTML+Handlebars'), ('html+lasso', 'HTML+Lasso'), ('html+mako', 'HTML+Mako'),
('html+myghty', 'HTML+Myghty'), ('html+ng2', 'HTML + Angular2'), ('html+php',
'HTML+PHP'), ('html+smarty', 'HTML+Smarty'), ('html+twig', 'HTML+Twig'), ('html+ul4',
'HTML+UL4'), ('html+velocity', 'HTML+Velocity'), ('http', 'HTTP'), ('hybris', 'Hybris'),
('hylang', 'Hy'), ('i6t', 'Inform 6 template'), ('icon', 'Icon'), ('idl', 'IDL'), ('idris', 'Idris'), ('iex',
'Elixir iex session'), ('igor', 'Igor'), ('inform6', 'Inform 6'), ('inform7', 'Inform 7'), ('ini', 'INI'),
('io', 'Io'), ('ioke', 'Ioke'), ('irc', 'IRC logs'), ('isabelle', 'Isabelle'), ('j', 'J'), ('jags', 'JAGS'),
('janet', 'Janet'), ('jasmin', 'Jasmin'), ('java', 'Java'), ('javascript', 'JavaScript'),
```

('javascript+cheetah', 'JavaScript+Cheetah'), ('javascript+django',
'JavaScript+Django/Jinja'), ('javascript+lasso', 'JavaScript+Lasso'), ('javascript+mako',
'JavaScript+Mako'), ('javascript+mozpreproc', 'Javascript+mozpreproc'),
('javascript+myghty', 'JavaScript+Myghty'), ('javascript+php', 'JavaScript+PHP'),
('javascript+ruby', 'JavaScript+Ruby'), ('javascript+smarty', 'JavaScript+Smarty'), ('jcl',
'JCL'), ('jlcon', 'Julia console'), ('jmespath', 'JMESPath'), ('js+genshitext',
'JavaScript+Genshi Text'), ('js+ul4', 'Javascript+UL4'), ('jsgf', 'JSGF'), ('jslt', 'JSLT'), ('json',
'JSON'), ('jsonld', 'JSON-LD'), ('jsonnet', 'Jsonnet'), ('jsp', 'Java Server Page'), ('jsx', 'JSX'),
('julia', 'Julia'), ('juttle', 'Juttle'), ('k', 'K'), ('kal', 'Kal'), ('kconfig', 'Kconfig'), ('kmsg', 'Kernel
log'), ('koka', 'Koka'), ('kotlin', 'Kotlin'), ('kql', 'Kusto'), ('kuin', 'Kuin'), ('lasso', 'Lasso'),
('ldapconf', 'LDAP configuration file'), ('ldif', 'LDIF'), ('lean', 'Lean'), ('lean4', 'Lean4'), ('less',
'LessCss'), ('lighttpd', 'Lighttpd configuration file'), ('lilypond', 'LilyPond'), ('limbo', 'Limbo'),
('liquid', 'liquid'), ('literate-agda', 'Literate Agda'), ('literate-cryptol', 'Literate Cryptol'),
('literate-haskell', 'Literate Haskell'), ('literate-idris', 'Literate Idris'), ('livescript', 'LiveScript'),
('llvm', 'LLVM'), ('llvm-mir', 'LLVM-MIR'), ('llvm-mir-body', 'LLVM-MIR Body'), ('logos',
'Logos'), ('logtalk', 'Logtalk'), ('lsl', 'LSL'), ('lua', 'Lua'), ('luau', 'Luau'), ('macaulay2',
'Macaulay2'), ('make', 'Makefile'), ('mako', 'Mako'), ('maql', 'MAQL'), ('markdown',
'Markdown'), ('mask', 'Mask'), ('mason', 'Mason'), ('mathematica', 'Mathematica'), ('matlab',
'Matlab'), ('matlabsession', 'Matlab session'), ('maxima', 'Maxima'), ('mcfunction',
'MCFunction'), ('mcschema', 'MCSchema'), ('meson', 'Meson'), ('mime', 'MIME'), ('minid',
'MiniD'), ('miniscript', 'MiniScript'), ('mips', 'MIPS'), ('modelica', 'Modelica'), ('modula2',
'Modula-2'), ('mojo', 'Mojo'), ('monkey', 'Monkey'), ('monte', 'Monte'), ('moocode',
'MOOCode'), ('moonscript', 'MoonScript'), ('mosel', 'Mosel'), ('mozhashpreproc',
'mozhashpreproc'), ('mozpercentpreproc', 'mozpercentpreproc'), ('mql', 'MQL'), ('mscgen',
'Mscgen'), ('mupad', 'MuPAD'), ('mxml', 'MXML'), ('myghty', 'Myghty'), ('mysql', 'MySQL'),
('nasm', 'NASM'), ('ncl', 'NCL'), ('nemerle', 'Nemerle'), ('nesc', 'nesC'), ('nestedtext',
'NestedText'), ('newlisp', 'NewLisp'), ('newspeak', 'Newspeak'), ('ng2', 'Angular2'), ('nginx',
'Nginx configuration file'), ('nimrod', 'Nimrod'), ('nit', 'Nit'), ('nixos', 'Nix'), ('nodejsrepl',
'Node.js REPL console session'), ('notmuch', 'Notmuch'), ('nsis', 'NSIS'), ('numpy',
'NumPy'), ('nusmv', 'NuSMV'), ('objdump', 'objdump'), ('objdump-nasm', 'objdump-nasm'),
('objective-c', 'Objective-C'), ('objective-c++', 'Objective-C++'), ('objective-j', 'Objective-J'),
('ocaml', 'OCaml'), ('octave', 'Octave'), ('odin', 'ODIN'), ('omg-idl', 'OMG Interface Definition
Language'), ('ooc', 'Ooc'), ('opa', 'Opa'), ('openedge', 'OpenEdge ABL'), ('openscad',
'OpenSCAD'), ('org', 'Org Mode'), ('output', 'Text output'), ('pacmanconf', 'PacmanConf'),
('pan', 'Pan'), ('parasail', 'ParaSail'), ('pawn', 'Pawn'), ('peg', 'PEG'), ('perl', 'Perl'), ('perl6',
'Perl6'), ('phix', 'Phix'), ('php', 'PHP'), ('pig', 'Pig'), ('pike', 'Pike'), ('pkgconfig', 'PkgConfig'),
('plpgsql', 'PL/pgSQL'), ('pointless', 'Pointless'), ('pony', 'Pony'), ('portugol', 'Portugol'),
('postgres-explain', 'PostgreSQL EXPLAIN dialect'), ('postgresql', 'PostgreSQL SQL
dialect'), ('postscript', 'PostScript'), ('pot', 'Gettext Catalog'), ('pov', 'POVRay'), ('powershell',
'PowerShell'), ('praat', 'Praat'), ('procfile', 'Procfile'), ('prolog', 'Prolog'), ('promela',
'Promela'), ('promql', 'PromQL'), ('properties', 'Properties'), ('protobuf', 'Protocol Buffer'),
('prql', 'PRQL'), ('psql', 'PostgreSQL console (psql)'), ('psysh', 'PsySH console session for
PHP'), ('ptx', 'PTX'), ('pug', 'Pug'), ('puppet', 'Puppet'), ('pwsh-session', 'PowerShell
Session'), ('py+ul4', 'Python+UL4'), ('py2tb', 'Python 2.x Traceback'), ('pycon', 'Python
console session'), ('pypylog', 'PyPy Log'), ('pytb', 'Python Traceback'), ('python', 'Python'),
('python2', 'Python 2.x'), ('q', 'Q'), ('qbasic', 'QBasic'), ('qlik', 'Qlik'), ('qml', 'QML'), ('qvto',
'QVTO'), ('racket', 'Racket'), ('ragel', 'Ragel'), ('ragel-c', 'Ragel in C Host'), ('ragel-cpp',
'Ragel in CPP Host'), ('ragel-d', 'Ragel in D Host'), ('ragel-em', 'Embedded Ragel'),
('ragel-java', 'Ragel in Java Host'), ('ragel-objc', 'Ragel in Objective C Host'), ('ragel-ruby',
'Ragel in Ruby Host'), ('rbcon', 'Ruby irb session'), ('rconsole', 'RConsole'), ('rd', 'Rd'),
('reasonml', 'ReasonML'), ('rebol', 'REBOL'), ('red', 'Red'), ('redcode', 'Redcode'), ('registry',
'reg'), ('resourcebundle', 'ResourceBundle'), ('restructuredtext', 'reStructuredText'), ('rexx',
'Rexx'), ('rhtml', 'RHTML'), ('ride', 'Ride'), ('rita', 'Rita'), ('rng-compact', 'Relax-NG
Compact'), ('roboconf-graph', 'Roboconf Graph'), ('roboconf-instances', 'Roboconf

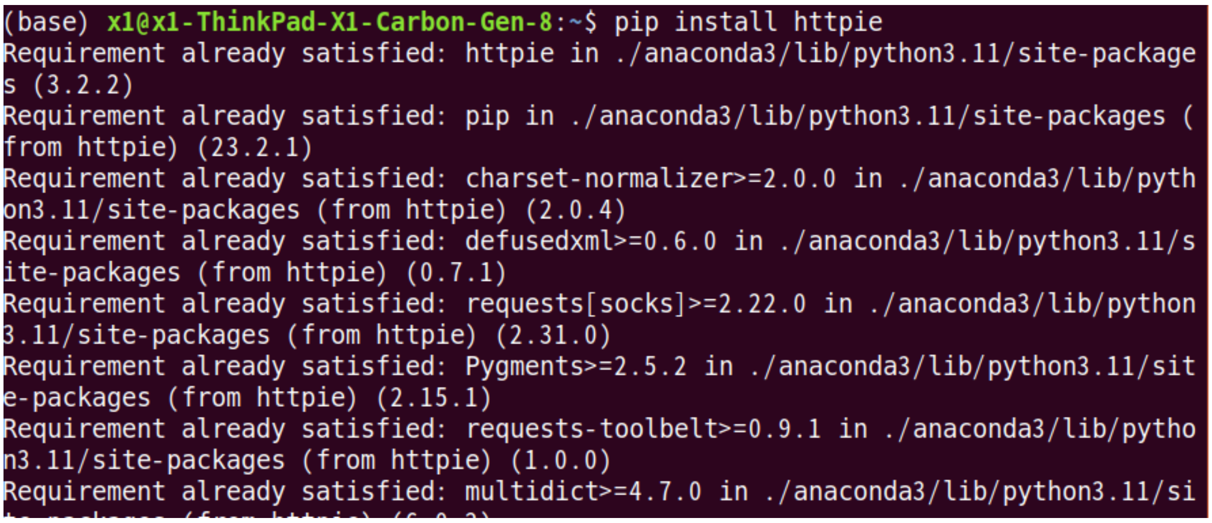
```
Instances'), ('robotframework', 'RobotFramework'), ('rql', 'RQL'), ('rsl', 'RSL'), ('ruby',
'Ruby'), ('rust', 'Rust'), ('sarl', 'SARL'), ('sas', 'SAS'), ('sass', 'Sass'), ('savi', 'Savi'), ('scala',
'Scala'), ('scaml', 'Scaml'), ('scdoc', 'scdoc'), ('scheme', 'Scheme'), ('scilab', 'Scilab'), ('scss',
'SCSS'), ('sed', 'Sed'), ('sgf', 'SmartGameFormat'), ('shen', 'Shen'), ('shexc', 'ShExC'),
('sieve', 'Sieve'), ('silver', 'Silver'), ('singularity', 'Singularity'), ('slash', 'Slash'), ('slim', 'Slim'),
('slurm', 'Slurm'), ('smali', 'Smali'), ('smalltalk', 'Smalltalk'), ('smarty', 'Smarty'), ('smithy',
'Smithy'), ('sml', 'Standard ML'), ('snbt', 'SNBT'), ('snobol', 'Snobol'), ('snowball',
'Snowball'), ('solidity', 'Solidity'), ('sophia', 'Sophia'), ('sp', 'SourcePawn'), ('sparql',
'SPARQL'), ('spec', 'RPMSpec'), ('spice', 'Spice'), ('splus', 'S'), ('sql', 'SQL'), ('sql+jinja',
'SQL+Jinja'), ('sqlite3', 'sqlite3con'), ('squidconf', 'SquidConf'), ('srcinfo', 'Srcinfo'), ('ssp',
'Scalate Server Page'), ('stan', 'Stan'), ('stata', 'Stata'), ('supercollider', 'SuperCollider'),
('swift', 'Swift'), ('swig', 'SWIG'), ('systemd', 'Systemd'), ('systemverilog', 'systemverilog'),
('tact', 'Tact'), ('tads3', 'TADS 3'), ('tal', 'Tal'), ('tap', 'TAP'), ('tasm', 'TASM'), ('tcl', 'Tcl'),
('tcsh', 'Tcsh'), ('tcshcon', 'Tcsh Session'), ('tea', 'Tea'), ('teal', 'teal'), ('teratermmacro', 'Tera
Term macro'), ('termcap', 'Termcap'), ('terminfo', 'Terminfo'), ('terraform', 'Terraform'), ('tex',
'TeX'), ('text', 'Text only'), ('thrift', 'Thrift'), ('ti', 'ThingsDB'), ('tid', 'tiddler'), ('tlb', 'Tl-b'), ('tls',
'TLS Presentation Language'), ('tnt', 'Typographic Number Theory'), ('todotxt', 'Todotxt'),
('toml', 'TOML'), ('trac-wiki', 'MoinMoin/Trac Wiki markup'), ('trafficscript', 'TrafficScript'),
('treetop', 'Treetop'), ('tsql', 'Transact-SQL'), ('turtle', 'Turtle'), ('twig', 'Twig'), ('typescript',
'TypeScript'), ('typoscript', 'TypoScript'), ('typoscriptcssdata', 'TypoScriptCssData'),
('typoscripthtmldata', 'TypoScriptHtmlData'), ('typst', 'Typst'), ('ucode', 'ucode'), ('ul4', 'UL4'),
('unicon', 'Unicon'), ('unixconfig', 'Unix/Linux config files'), ('urbiscript', 'UrbiScript'),
('urlencoded', 'urlencoded'), ('usd', 'USD'), ('vala', 'Vala'), ('vb.net', 'VB.net'), ('vbscript',
'VBScript'), ('vcl', 'VCL'), ('vclsnippets', 'VCLSnippets'), ('vctreestatus', 'VCTreeStatus'),
('velocity', 'Velocity'), ('verifpal', 'Verifpal'), ('verilog', 'verilog'), ('vgl', 'VGL'), ('vhdl', 'vhdl'),
('vim', 'VimL'), ('visualprolog', 'Visual Prolog'), ('visualprologgrammar', 'Visual Prolog
Grammar'), ('vyper', 'Vyper'), ('wast', 'WebAssembly'), ('wdiff', 'WDiff'), ('webidl', 'Web IDL'),
('wgsl', 'WebGPU Shading Language'), ('whiley', 'Whiley'), ('wikitext', 'Wikitext'), ('wowtoc',
'World of Warcraft TOC'), ('wren', 'Wren'), ('x10', 'X10'), ('xml', 'XML'), ('xml+cheetah',
'XML+Cheetah'), ('xml+django', 'XML+Django/Jinja'), ('xml+evoque', 'XML+Evoque'),
('xml+lasso', 'XML+Lasso'), ('xml+mako', 'XML+Mako'), ('xml+myghty', 'XML+Myghty'),
('xml+php', 'XML+PHP'), ('xml+ruby', 'XML+Ruby'), ('xml+smarty', 'XML+Smarty'),
('xml+ul4', 'XML+UL4'), ('xml+velocity', 'XML+Velocity'), ('xorg.conf', 'Xorg'), ('xpp', 'X++'),
('xquery', 'XQuery'), ('xslt', 'XSLT'), ('xtend', 'Xtend'), ('xul+mozpreproc',
'XUL+mozpreproc'), ('yaml', 'YAML'), ('yaml+jinja', 'YAML+Jinja'), ('yang', 'YANG'), ('yara',
'YARA'), ('zeek', 'Zeek'), ('zephir', 'Zephir'), ('zig', 'Zig'), ('zone', 'Zone')], default='python')
    style = ChoiceField(choices=[('abap', 'abap'), ('algol', 'algol'), ('algol_nu', 'algol_nu'),
('arduino', 'arduino'), ('autumn', 'autumn'), ('borland', 'borland'), ('bw', 'bw'), ('coffee',
'coffee'), ('colorful', 'colorful'), ('default', 'default'), ('dracula', 'dracula'), ('emacs', 'emacs'),
('friendly', 'friendly'), ('friendly_grayscale', 'friendly_grayscale'), ('fruity', 'fruity'),
('github-dark', 'github-dark'), ('gruvbox-dark', 'gruvbox-dark'), ('gruvbox-light',
'gruvbox-light'), ('igor', 'igor'), ('inkpot', 'inkpot'), ('lightbulb', 'lightbulb'), ('lilypond', 'lilypond'),
('lovelace', 'lovelace'), ('manni', 'manni'), ('material', 'material'), ('monokai', 'monokai'),
('murphy', 'murphy'), ('native', 'native'), ('nord', 'nord'), ('nord-darker', 'nord-darker'),
('one-dark', 'one-dark'), ('paraiso-dark', 'paraiso-dark'), ('paraiso-light', 'paraiso-light'),
('pastie', 'pastie'), ('perldoc', 'perldoc'), ('rainbow_dash', 'rainbow_dash'), ('rrt', 'rrt'), ('sas',
'sas'), ('solarized-dark', 'solarized-dark'), ('solarized-light', 'solarized-light'), ('staroffice',
'staroffice'), ('stata-dark', 'stata-dark'), ('stata-light', 'stata-light'), ('tango', 'tango'), ('trac',
'trac'), ('vim', 'vim'), ('vs', 'vs'), ('xcode', 'xcode'), ('zenburn', 'zenburn')], default='friendly')
```



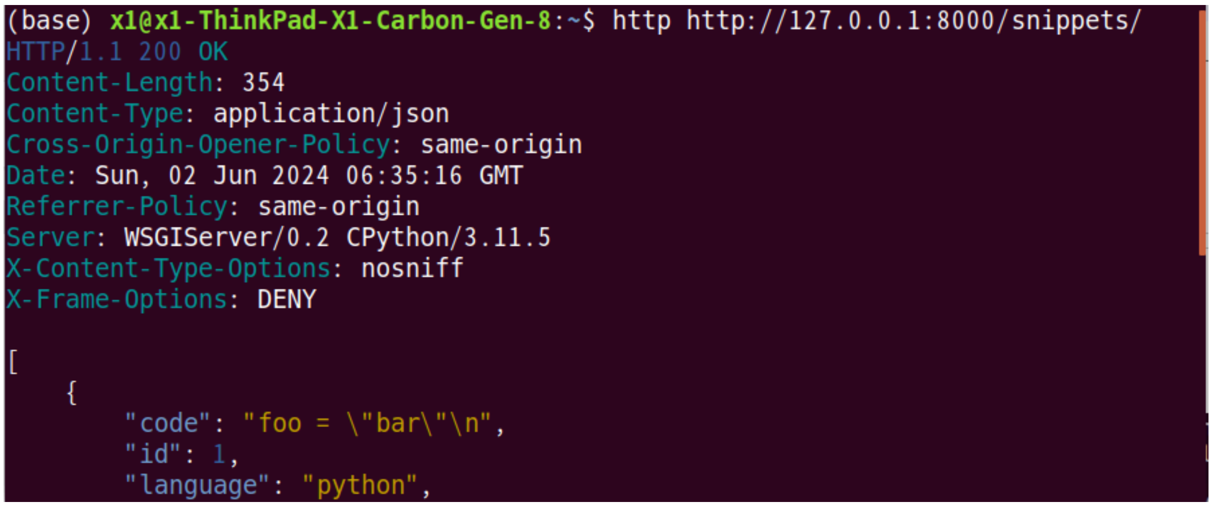
```
@csrf_exempt
def snippet_list(request):
    """
    List all code snippets, or create a new snippet.
    """
    if request.method == 'GET':
        snippets = Snippet.objects.all() # snippets은 python data
        serializer = SnippetSerializer(snippets, many=True)  #
SnippetSerializer는 python data snippets을 JSON 타입으로 serializer로 객체화함
        return JsonResponse(serializer.data, safe=False)  # JSON 타입
serializer.data는 JsonResponse를 통해 CLIENT에서 JSON 타입으로 보여지게 됨

    elif request.method == 'POST':
        data = JSONParser().parse(request)
        serializer = SnippetSerializer(data=data)
        if serializer.is_valid():
            serializer.save()
            return JsonResponse(serializer.data, status=201)
        return JsonResponse(serializer.errors, status=400)
```



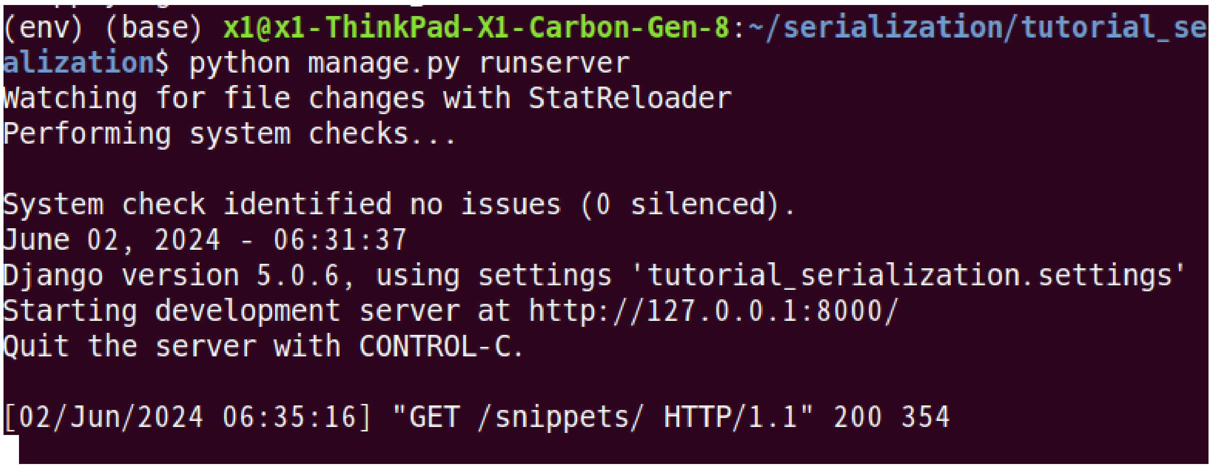
```
(env) (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/serialization/tutorial_seri
alization$ python manage.py makemigrations
No changes detected
(env) (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/serialization/tutorial_seri
alization$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions, snippets
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
```



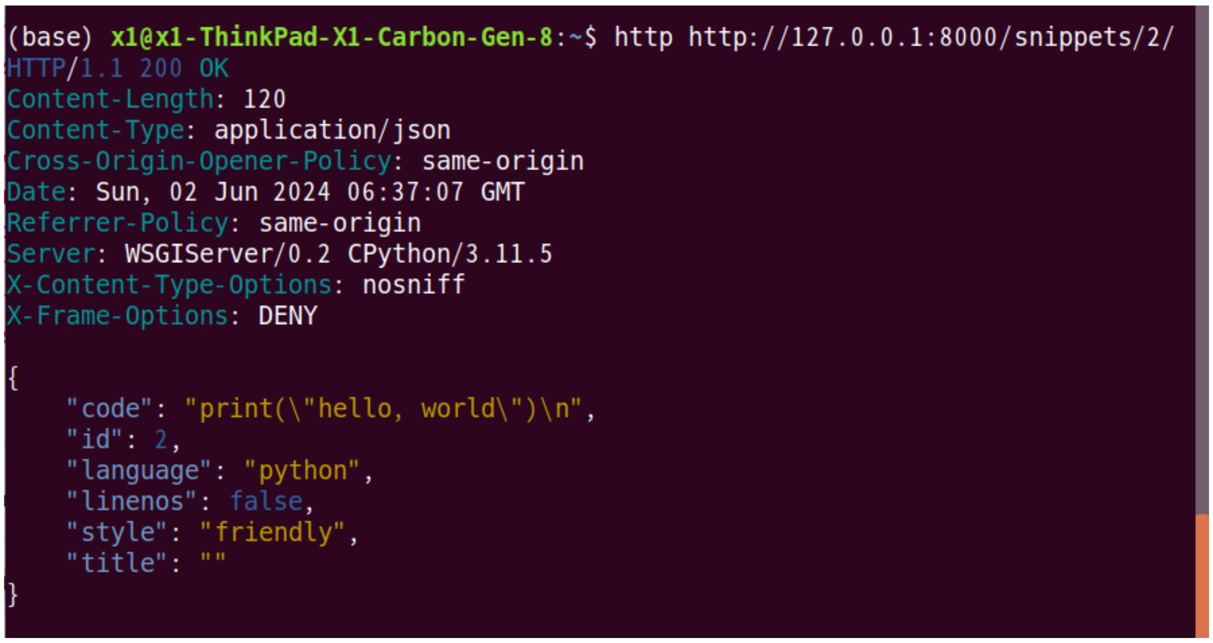
```
(base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~$ pip install httpie
Requirement already satisfied: httpie in ./anaconda3/lib/python3.11/site-package
s (3.2.2)
Requirement already satisfied: pip in ./anaconda3/lib/python3.11/site-packages (
from httpie) (23.2.1)
Requirement already satisfied: charset-normalizer>=2.0.0 in ./anaconda3/lib/pyth
on3.11/site-packages (from httpie) (2.0.4)
Requirement already satisfied: defusedxml>=0.6.0 in ./anaconda3/lib/python3.11/s
ite-packages (from httpie) (0.7.1)
Requirement already satisfied: requests[socks]>=2.22.0 in ./anaconda3/lib/python
3.11/site-packages (from httpie) (2.31.0)
Requirement already satisfied: Pygments>=2.5.2 in ./anaconda3/lib/python3.11/sit
e-packages (from httpie) (2.15.1)
Requirement already satisfied: requests-toolbelt>=0.9.1 in ./anaconda3/lib/pytho
n3.11/site-packages (from httpie) (1.0.0)
Requirement already satisfied: multidict>=4.7.0 in ./anaconda3/lib/python3.11/si
```



```
(base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~$ http http://127.0.0.1:8000/snippets/
HTTP/1.1 200 OK
Content-Length: 354
Content-Type: application/json
Cross-Origin-Opener-Policy: same-origin
Date: Sun, 02 Jun 2024 06:35:16 GMT
Referrer-Policy: same-origin
Server: WSGIServer/0.2 CPython/3.11.5
X-Content-Type-Options: nosniff
X-Frame-Options: DENY

[
    {
        "code": "foo = \"bar\"\n",
        "id": 1,
        "language": "python",
```



```
(env) (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/serialization/tutorial_se
alization$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
June 02, 2024 - 06:31:37
Django version 5.0.6, using settings 'tutorial_serialization.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.

[02/Jun/2024 06:35:16] "GET /snippets/ HTTP/1.1" 200 354
```



```
(base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~$ http http://127.0.0.1:8000/snippets/2/
HTTP/1.1 200 OK
Content-Length: 120
Content-Type: application/json
Cross-Origin-Opener-Policy: same-origin
Date: Sun, 02 Jun 2024 06:37:07 GMT
Referrer-Policy: same-origin
Server: WSGIServer/0.2 CPython/3.11.5
X-Content-Type-Options: nosniff
X-Frame-Options: DENY

{
    "code": "print(\"hello, world\")\n",
    "id": 2,
    "language": "python",
    "linenos": false,
    "style": "friendly",
    "title": ""
}
```

In another terminal window, we can test the server.

We can test our API using curl or httpie. Httpie is a user friendly http client that's written in Python. Let's install that.

You can install httpie using pip:

```
pip install httpie
```

Finally, we can get a list of all of the snippets:

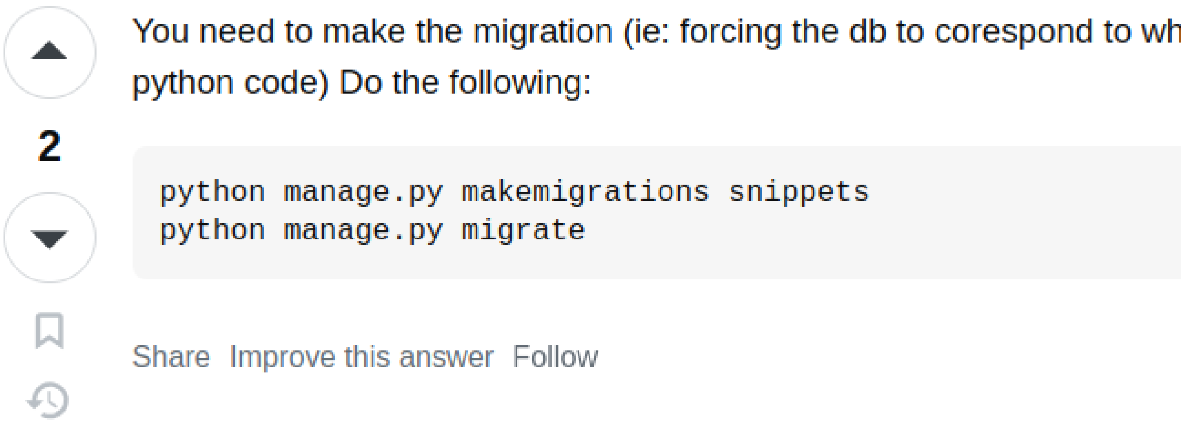
```
http http://127.0.0.1:8000/snippets/

HTTP/1.1 200 OK
...
[
  {
    "id": 1,
    "title": "",
    "code": "foo = \"bar\"\n",
    "linenos": false,
    "language": "python",
    "style": "friendly"
  },
  {
    "id": 2,
    "title": "",
    "code": "print(\"hello, world\")\n",
    "linenos": false,
    "language": "python",
    "style": "friendly"
  }
]
```

Or we can get a particular snippet by referencing its id:

```
http http://127.0.0.1:8000/snippets/2/

HTTP/1.1 200 OK
...
{
  "id": 2,
  "title": "",
  "code": "print(\"hello, world\")\n",
  "linenos": false,
  "language": "python",
  "style": "friendly"
}
```

1. 에러 1 처리

- https://stackoverflow.com/questions/39696148/django-no-such-table-snippets-snippet


You need to make the migration (ie: forcing the db to corespond to wh
python code) Do the following:

```
python manage.py makemigrations snippets
python manage.py migrate
```

```
 (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/tutorial3/tutorial$ python managey makemigrations snippets
Migrations for 'snippets':
  snippets/migrations/0001_initial.py
    - Create model Snippet


(env) (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/tutorial3/tutorial$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions, snippets
Running migrations:
  Applying snippets.0001_initial... OK
```

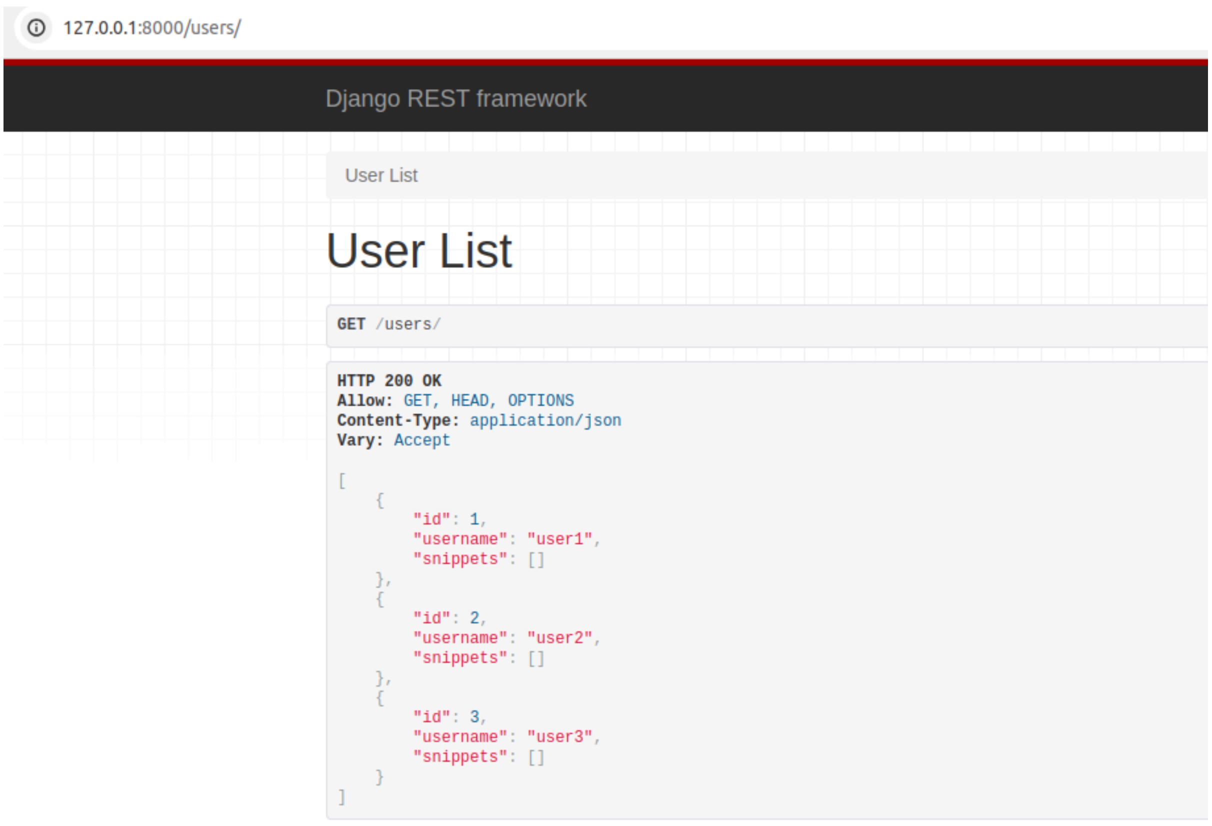
```
(env) (base) x1@x1-ThinkPad-X1-Carbon-Gen-8:~/tutorial3/tutorial$
```

- 4장 인증 에서는 python manage.py makemigrations snippets 명령어를 사용해야 snippets 이 업데이트 되면서 화면이 나타남

- 아래 snippets에 `owner`와 `highlighted` 필드가 추가됨

```
class Snippet(models.Model):
    owner = models.ForeignKey('auth.User',
related_name='snippets', on_delete=models.CASCADE)
    highlighted = models.TextField()
```

- http://127.0.0.1:8000/users/
- user 화면이 나타남


127.0.0.1:8000/users/

Django REST framework

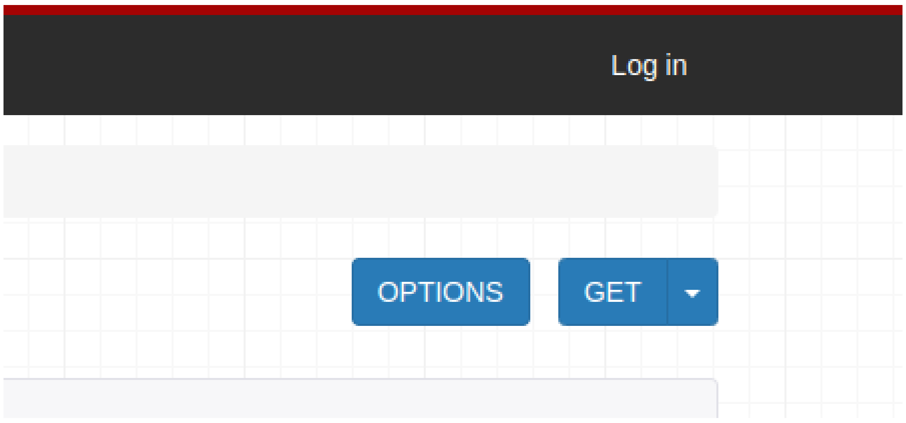
User List

User List

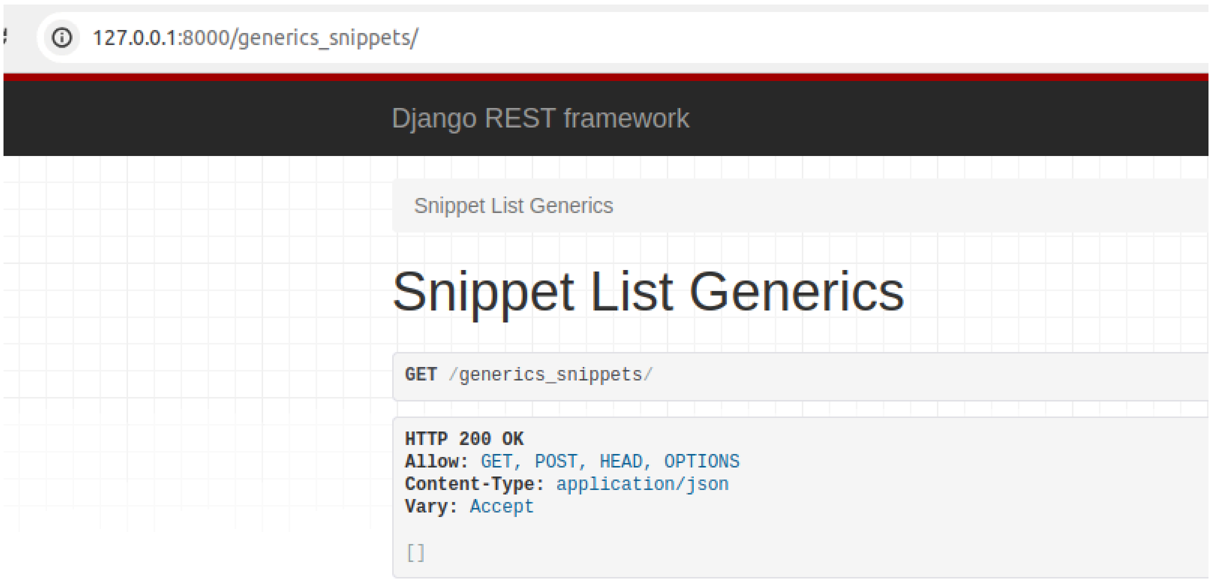
```
GET /users/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 1,
        "username": "user1",
        "snippets": []
    },
    {
        "id": 2,
        "username": "user2",
        "snippets": []
    },
    {
        "id": 3,
        "username": "user3",
        "snippets": []
    }
]
```

- 로그인이 나타남
    - 로그인이 나타나야 user1으로 로그인하고 새로운 user1이 작성한 snippets들을 작성할 수 있음

- 

- http://127.0.0.1:8000/generics_snippets/ 화면이 나타남



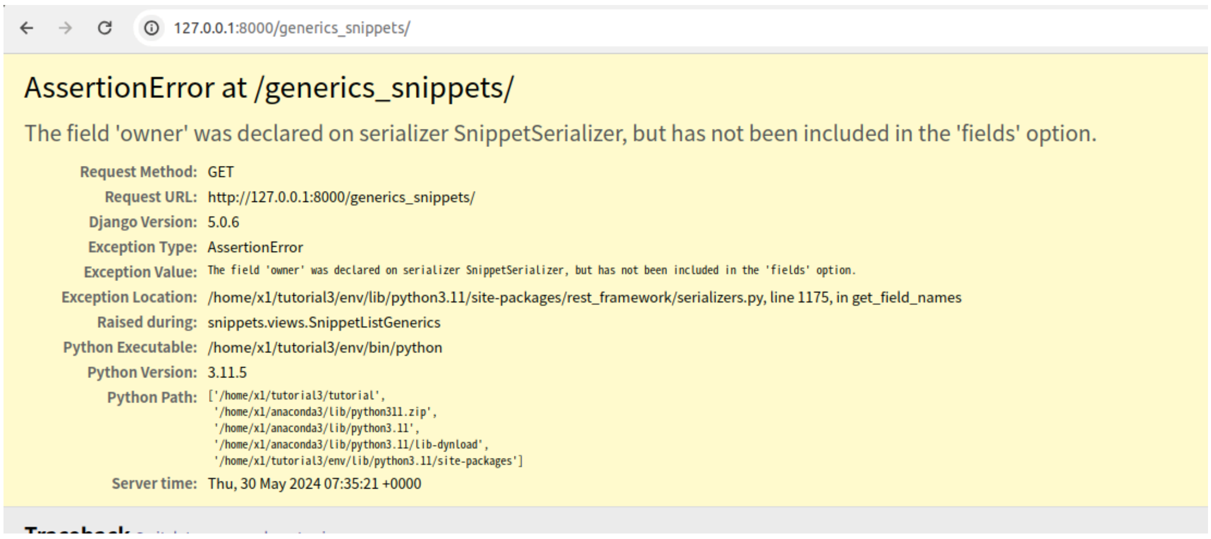
Snippet List Generics

```
GET /generics_snippets/
```

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[]
```

2. 에러 2 처리

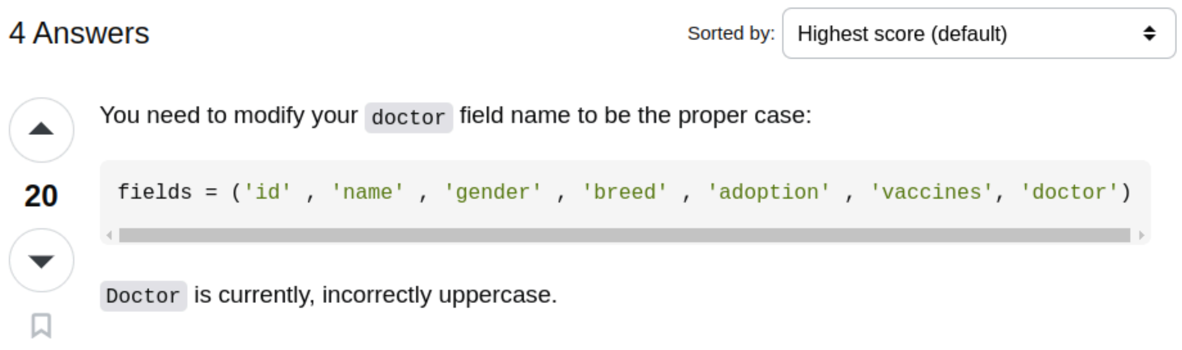
- 로그인 하고 generics_에 접속하면 아래 화면 이 나타남
    - The field 'owner' was declared on serializer SnippetSerializer, but has not been included in the 'fields' option.
    - owner가 있는 데 owner가 없음
    - 'owner' 필드가 serializer SnippetSerializer에서 선언되었지만 'fields' 옵션에 포함되지 않았습니다.
    -



127.0.0.1:8000/generics_snippets/

```
AssertionError at /generics_snippets/

The field 'owner' was declared on serializer SnippetSerializer, but has not been included in the 'fields' option.

     Request Method: GET
        Request URL: http://127.0.0.1:8000/generics_snippets/
     Django Version: 5.0.6
     Exception Type: AssertionError
    Exception Value: The field 'owner' was declared on serializer SnippetSerializer, but has not been included in the 'fields' option.
 Exception Location: /home/x1/tutorial3/env/lib/python3.11/site-packages/rest_framework/serializers.py, line 1175, in get_field_names
      Raised during: snippets.views.SnippetListGenerics
  Python Executable: /home/x1/tutorial3/env/bin/python
     Python Version: 3.11.5
        Python Path: ['/home/x1/tutorial3/tutorial',
                      '/home/x1/anaconda3/lib/python311.zip',
                      '/home/x1/anaconda3/lib/python3.11',
                      '/home/x1/anaconda3/lib/python3.11/lib-dynload',
                      '/home/x1/tutorial3/env/lib/python3.11/site-packages']
        Server time: Thu, 30 May 2024 07:35:21 +0000
```

- model에 새롭게 추가된 owner 필드를 serializer의 필드에도 새롭게 추가해야 함 https://stackoverflow.com/questions/36336145/assertionerror-the-field-was-declared-on-serializer-but-has-not-been-i



4 Answers

Sorted by: Highest score (default)

20

You need to modify your `doctor` field name to be the proper case:

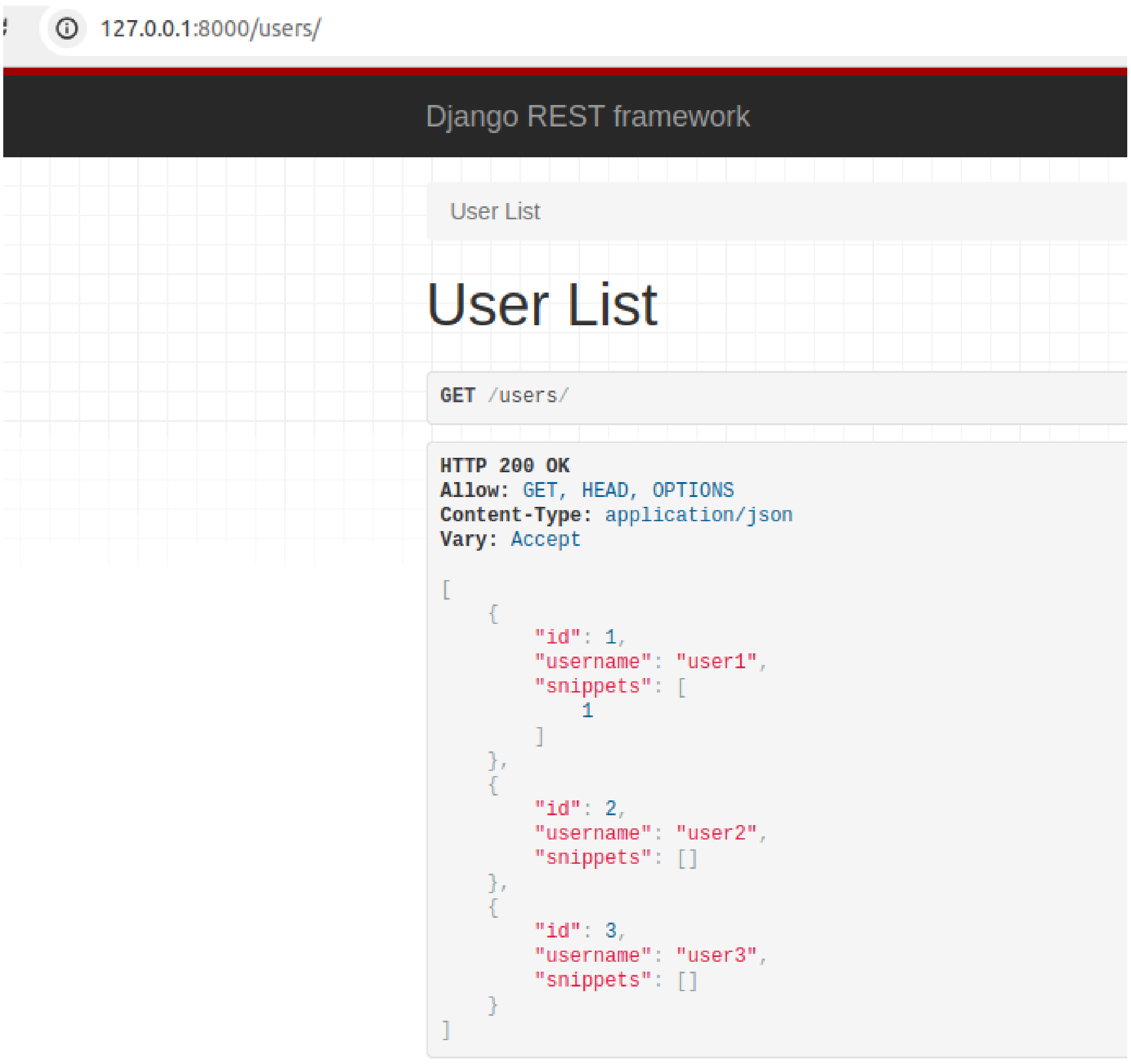
```
fields = ('id' , 'name' , 'gender' , 'breed' , 'adoption' , 'vaccines', 'doctor')
```

`Doctor` is currently, incorrectly uppercase.

```
class SnippetSerializer(serializers.ModelSerializer):
   owner = serializers.ReadOnlyField(source='owner.username')

   class Meta:
       model = Snippet
       fields = ['id', 'title', 'code', 'linenos', 'language',
'style', 'owner'] # 'owner' 필드 추가함 (에러 2 해결)
```

- http://127.0.0.1:8000/users/ 화면에서 user1으로 로그인하고, 아래 http://127.0.0.1:8000/generics_snippets/ 에서 `snippets를 새롭게 하나 작성한 후` http://127.0.0.1:8000/users/ 화면에 가면 `snippets 이 한 개 추가된 것을 확인할 수 있음`
-

```
{
        "id": 1,
        "username": "user1",
        "snippets": [
            1
        ]
```



127.0.0.1:8000/users/

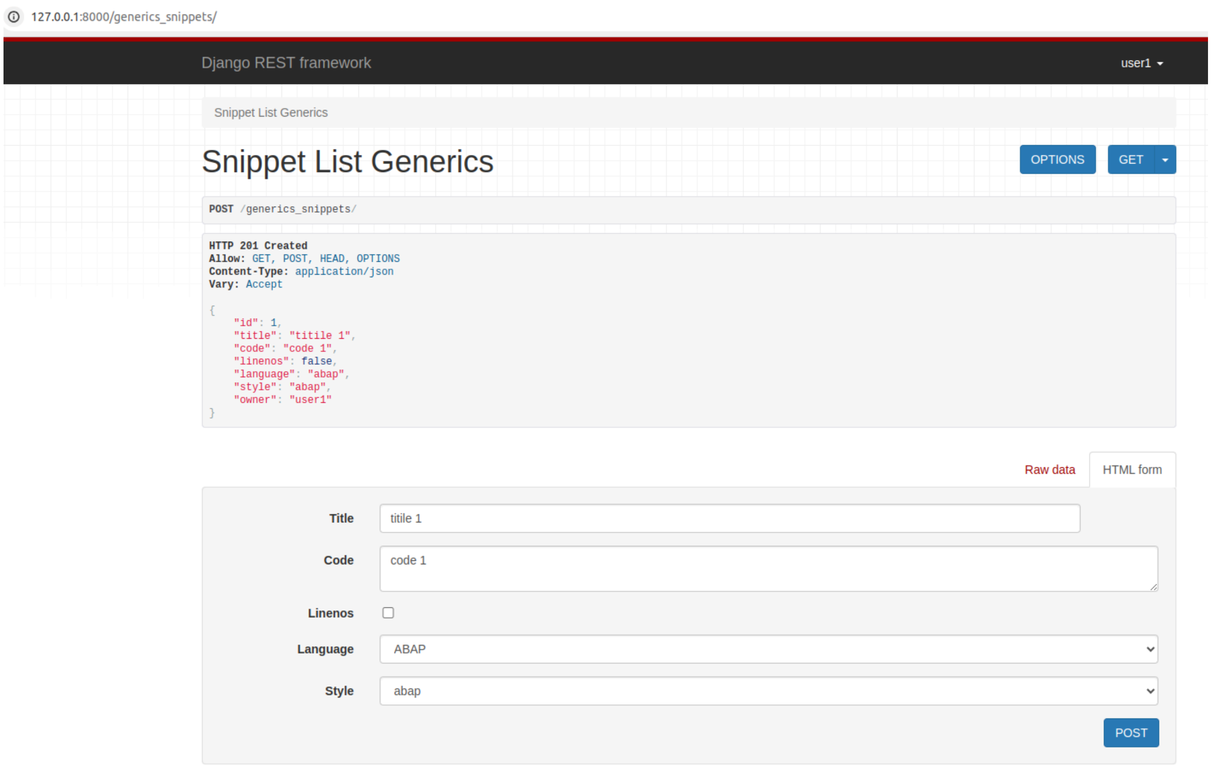
Django REST framework

User List

User List

```
GET /users/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 1,
        "username": "user1",
        "snippets": [
            1
        ]
    },
    {
        "id": 2,
        "username": "user2",
        "snippets": []
    },
    {
        "id": 3,
        "username": "user3",
        "snippets": []
    }
]
```



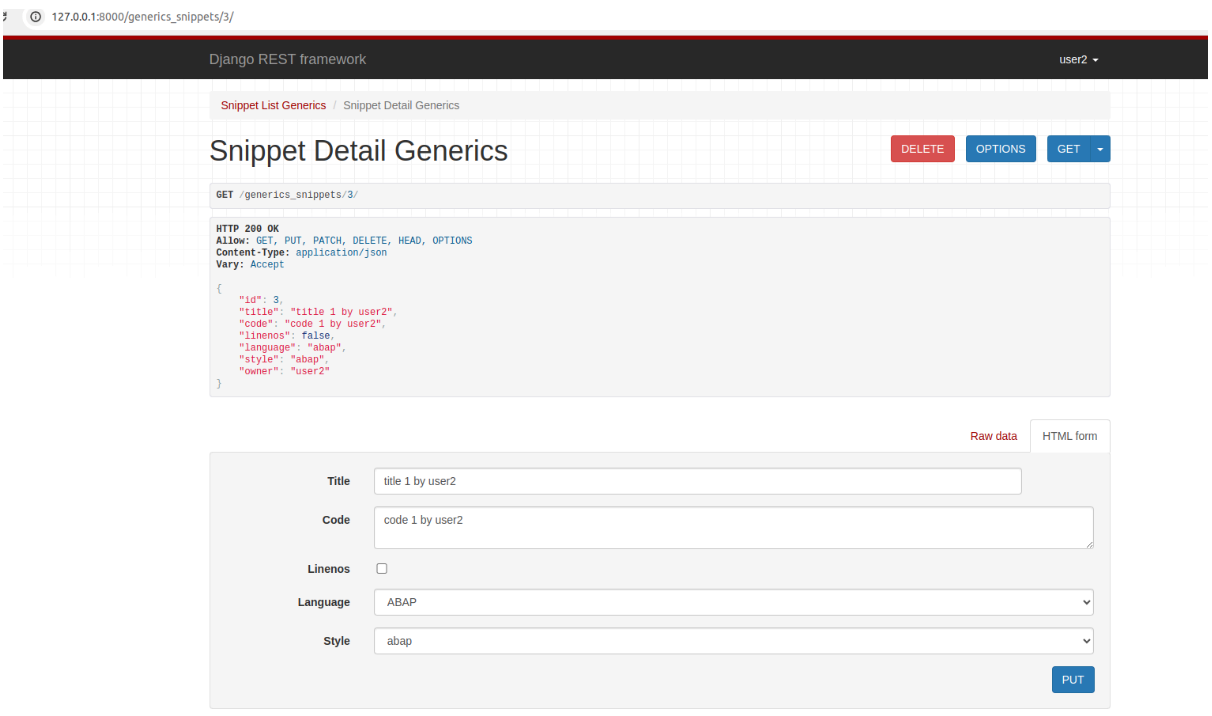
Snippet List Generics

```
POST /generics_snippets/
```

```
HTTP 201 Created
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 1,
    "title": "titile 1",
    "code": "code 1",
    "linenos": false,
    "language": "abap",
    "style": "abap",
    "owner": "user1"
}
```

Title	titile 1
Code	code 1
Linenos	☐
Language	ABAP
Style	abap

- user2로 로그인했을 때, user2가 작성한 snippet (id=3번이다) 에 대해서는 수정하고 삭제할 수 있는 권한이 부여된다.


127.0.0.1:8000/generics_snippets/3/
Django REST framework
user2
Snippet List Generics / Snippet Detail Generics

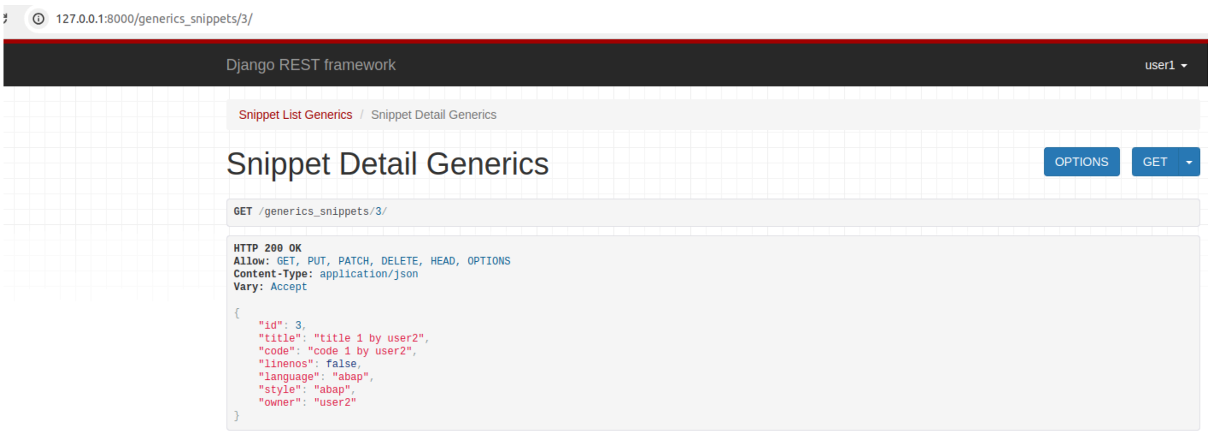
Snippet Detail Generics

```
GET /generics_snippets/3/
```

```
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 3,
    "title": "title 1 by user2",
    "code": "code 1 by user2",
    "linenos": false,
    "language": "abap",
    "style": "abap",
    "owner": "user2"
}
```

Title	title 1 by user2
Code	code 1 by user2
Linenos	☐
Language	ABAP
Style	abap

- user1로 로그인했을 때, user3번이 작성한 snippet id=3 은 수정할 수 없도록 되어 있다.


127.0.0.1:8000/generics_snippets/3/
Django REST framework
user1
Snippet List Generics / Snippet Detail Generics
Snippet Detail Generics
OPTIONS
GET

```
GET /generics_snippets/3/
```

```
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 3,
    "title": "title 1 by user2",
    "code": "code 1 by user2",
    "linenos": false,
    "language": "abap",
    "style": "abap",
    "owner": "user2"
}
```