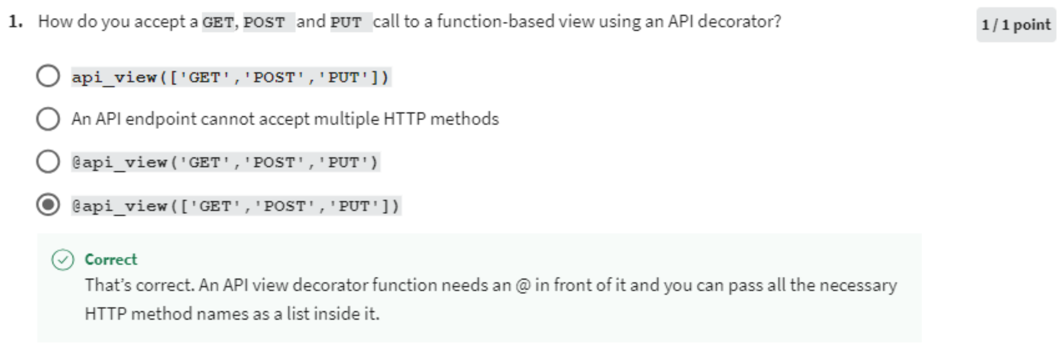


1. How do you accept a `GET`, `POST` and `PUT` call to a function-based view using an API decorator?

1 / 1 point

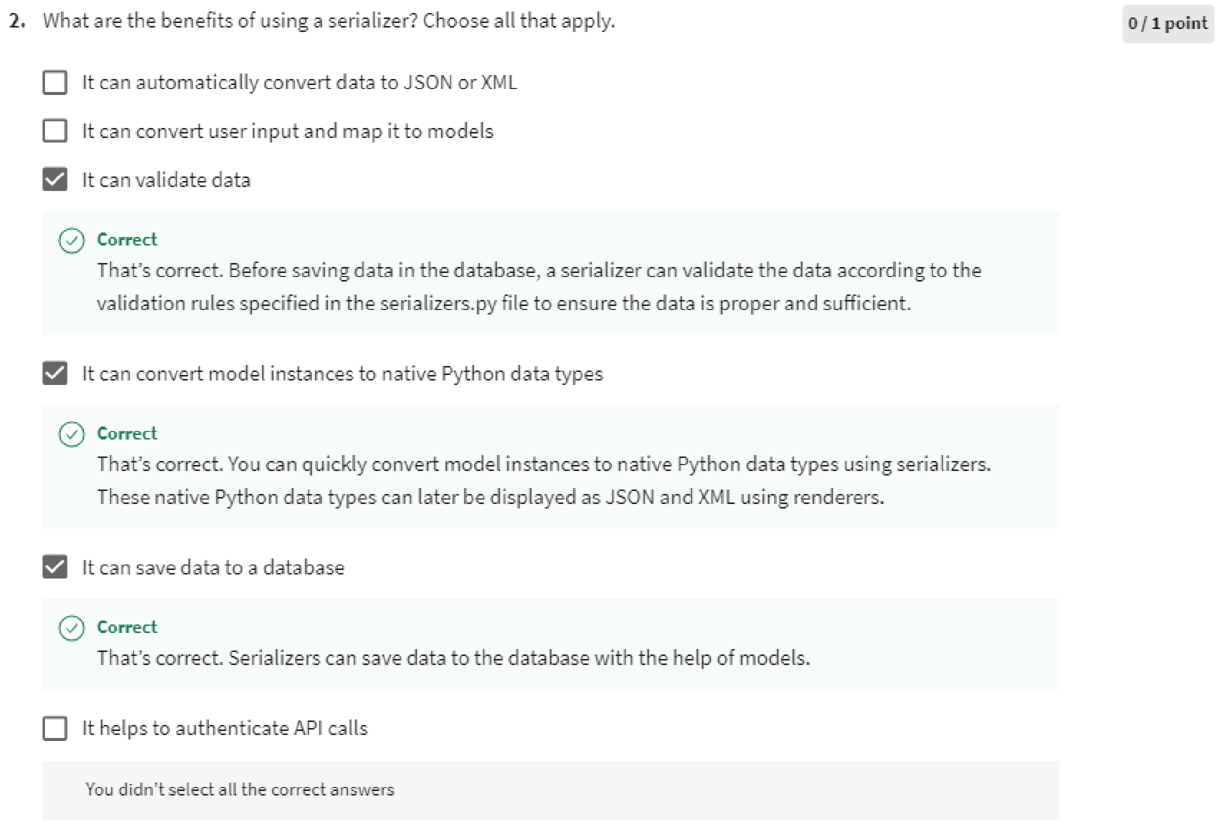
○ `api_view(['GET','POST','PUT'])`

○ An API endpoint cannot accept multiple HTTP methods

○ `@api_view('GET','POST','PUT')`

● `@api_view(['GET','POST','PUT'])`

Correct
That's correct. An API view decorator function needs an @ in front of it and you can pass all the necessary HTTP method names as a list inside it.



2. What are the benefits of using a serializer? Choose all that apply.

0 / 1 point

☐ It can automatically convert data to JSON or XML

☐ It can convert user input and map it to models

☑ It can validate data

Correct
That's correct. Before saving data in the database, a serializer can validate the data according to the validation rules specified in the serializers.py file to ensure the data is proper and sufficient.

☑ It can convert model instances to native Python data types

Correct
That's correct. You can quickly convert model instances to native Python data types using serializers. These native Python data types can later be displayed as JSON and XML using renderers.

☑ It can save data to a database

Correct
That's correct. Serializers can save data to the database with the help of models.

☐ It helps to authenticate API calls

You didn't select all the correct answers

3. Which of the following are valid serializer classes in DRF? Choose all that apply.

1 / 1 point

☑ `HyperLinkModelSerializer`

✓ **Correct**
That’s correct. You can use this serializer class to quickly create hyperlinked relationships between related models and display them as hyperlinks.

☐ `RelationshipSerializer`

☐ `PrimaryKeySerializer`

☑ `ModelSerializer`

✓ **Correct**
That’s correct. Model serializers are used to quickly serialize models and their relationships.

☑ `Serializer`

✓ **Correct**
That’s correct. This is the base Serializer class in DRF which can be used to serialize model instances and standalone objects.

4. You can access the data attribute of a serializer class at any time.

1 / 1 point

○ True

● False

✓ **Correct**
That’s correct. The data attribute of a serializer class can only be accessed after the validation is done in the serializer.

5. Which of the following renderers comes with DRF by default? Choose all that apply.

1 / 1 point

☐ XML Renderer

☐ YAML Renderer

☑ HTML Renderer

> **Correct**
> That's correct. DRF comes with a few HTML renderers to help you render static and dynamic HTML content.

☑ JSON Renderer

> **Correct**
> That's correct. The JSON renderer comes as a built-in package in the Django REST Framework.

6. Which of the following statement is true about DRF?

1 / 1 point

● DRF is built for API development

○ Learning DRF is tough

○ DRF doesn't work with different database engines

○ DRF is a standalone framework

> **Correct**
> That's correct. Though you can use DRF to create standard HTML content, DRF is specifically built for developers to create API projects very quickly. It comes with all the necessary classes and modules like ViewSets, generic views, serializers, authentication and permissions classes and many more which API developers require frequently in their projects. DRF also has excellent documentation and a huge community of developers so getting help or support is easier.

7. Which of the following panels are available in the DDT or Django debug toolbar? Choose all that apply.

1 / 1 point

☐ Network

☐ Throttle

☑ SQL

> **Correct**
> That’s correct. This panel displays all the SQL queries executed for the current request.

☑ Headers

> **Correct**
> That’s correct. The headers panel lists all the headers for the current request and response.

☑ Profiling

> **Correct**
> That’s correct. This panel displays the full call stack for the current request.

8. To serialize a queryset that returns more than one item, which of the following arguments is necessary for the serializer class?

1 / 1 point

○ `related=True`

○ `multiple_items=True`

● `many=True`

> **Correct**
> That’s correct. You need to pass many=True to the serializer class when it’s dealing with a queryset that returns more than one item.

9. Which of the following statements are true about using renderers? Choose all that apply.

1 / 1 point

☑ Renderers need an `Accept` header to work properly

> **Correct**
> That's correct. Based on these Accept headers DRF invokes the appropriate renderer to display the output properly.

☑ If no `Accept` header is present, DRF uses JSON renderer by default.

> **Correct**
> That's correct. If there is no accept header present, DRF displays the output in JSON using the built-in JSONRenderer class.

□ You cannot use multiple renderers in a project

□ You cannot forcefully use a single renderer

☑ Renderers can automatically convert the output

> **Correct**
> That's correct. When you load these renderer classes in the settings.py file, they will work automatically based on the Accept header that an API client sent. You don't need to write extra code for that.

10. How do you display related model fields as hyperlinks? Choose all that apply.

1 / 1 point

□ A ModelSerializer can do it automatically

□ Using `RelationshipSerializer`

☑ Using `HyperlinkedRelatedField`

> **Correct**
> That's correct. A HyperlinkedRelatedField serializer field can display related models as hyperlinks.

☑ Using `HyperlinkedModelSerializer`

> **Correct**
> That's correct. There is a special serializer class called HyperlinkedModelSerializer which can also display related models as a hyperlink.