

# Axion documentation

version

**Bobby**

May 11, 2025



# Contents

<b>AxionDocs documentation</b>	<b>1</b>
axion_smarty-main	1
App package	1
Submodules	1
App.admin module	1
App.apps module	1
App.constants module	1
App.models module	1
App.serializers module	1
App.signals module	2
App.tests module	2
App.urls module	2
App.views module	2
Module contents	2
AuthUser package	2
Subpackages	2
AuthUser.migrations package	2
Submodules	2
AuthUser.migrations.0001_initial module	2
AuthUser.migrations.0002_client_first_name_client_last_name module	3
Module contents	3
Submodules	3
AuthUser.admin module	3
AuthUser.apps module	3
AuthUser.forms module	3
AuthUser.models module	4
AuthUser.serializers module	8
AuthUser.tests module	8
AuthUser.urls module	9
AuthUser.views module	9
Module contents	9
Device package	9
Subpackages	9
Device.migrations package	9
Submodules	9
Device.migrations.0001_initial module	9
Device.migrations.0002_alter_devicemodel_device_gen module	10
Module contents	10
Submodules	10
Device.admin module	10
Device.apps module	10
Device.forms module	10
Device.models module	10
Device.serializers module	13

Device.tests module	13
Device.urls module	14
Device.views module	14
Module contents	14
Entity package	14
Subpackages	14
Entity.migrations package	14
Submodules	14
Entity.migrations.0001_initial module	14
Module contents	14
Submodules	14
Entity.admin module	14
Entity.apps module	14
Entity.models module	14
Entity.serializers module	18
Entity.tests module	18
Entity.urls module	19
Entity.views module	19
Module contents	20
Main package	20
Submodules	20
Main.asgi module	20
Main.settings module	20
Main.urls module	20
Main.wsgi module	20
Module contents	21
MicrosoftAuth package	21
Submodules	21
MicrosoftAuth.admin module	21
MicrosoftAuth.apps module	21
MicrosoftAuth.authentication module	21
MicrosoftAuth.functions module	21
MicrosoftAuth.models module	21
MicrosoftAuth.tests module	21
MicrosoftAuth.urls module	21
MicrosoftAuth.views module	21
Module contents	22
Order package	22
Subpackages	22
Order.migrations package	22
Submodules	22
Order.migrations.0001_initial module	22
Order.migrations.0002_order_client_name module	22
Order.migrations.0003_rename_client_name_order_client_personal_name module	23
Order.migrations.0004_alter_maintenanceprotocol_location module	23
Order.migrations.0005_alter_maintenanceprotocol_status_and_more module	23

Module contents	23
Submodules	23
Order.admin module	23
Order.apps module	23
Order.forms module	23
Order.models module	24
Order.serializers module	27
Order.tests module	28
Order.urls module	28
Order.views module	29
Module contents	29
manage module	29
<b>Index</b>	<b>31</b>
<b>Python Module Index</b>	<b>39</b>



# AxionDocs documentation

Add your content using `reStructuredText` syntax. See the [reStructuredText](#) documentation for details.

## axion\_smarty-main

### App package

#### Submodules

#### App.admin module

Admin configuration for registering models to the Django admin interface.

This module registers all the key models used in the application so that they can be managed through the Django admin panel.

#### App.apps module

App configuration for the 'App' Django application.

This module sets up the application configuration and ensures that signals are imported when the app is ready.

**class** `App.apps.AppConfig` (`app_name`, `app_module`)

Bases: `AppConfig`

Configuration class for the 'App' application.

**default\_auto\_field**

The default field type for auto-generated primary keys.

**Type:** `str`

**name**

The full Python path to the application.

**Type:** `str`

**default\_auto\_field** = `'django.db.models.BigAutoField'`

**name** = `'App'`

**ready()**

Hook for application initialization.

This method is called when the application is fully loaded. It imports the *signals* module to register model signals.

#### App.constants module

#### App.models module

#### App.serializers module

Custom serializer base class for selective field serialization.

This module defines a base serializer that allows dynamically including only specified fields via the serializer's context.

**class** `App.serializers.ParentSerializer` (`*args`, `**kwargs`)

Bases: `ModelSerializer`

A base serializer that allows dynamic inclusion of fields.

This class enables child serializers to specify which fields to include by passing a *fields* list in the serializer context.

#### Behavior:

- If *fields* are passed in the context, only those fields will be retained.
- If no *fields* are passed, all fields defined in the Meta class will be included.

#### Example

```
`python serializer = MyChildSerializer(instance, context={'fields': ['id', 'name']})`
```

This approach is useful when building flexible APIs or reusable serializers.

## App.signals module

Signal handlers for the 'App' Django application.

This module handles model signals related to many-to-many relationships in Orders. Specifically, it listens for when devices are added to an Order and creates the corresponding MaintenanceProtocol entries based on the device model's specifications.

`App.signals.create_protocols_for_order` (sender, instance, action, pk\_set, \*\*kwargs)

Signal handler that creates MaintenanceProtocol entries when devices are added to an Order.

Triggered after devices are added to the many-to-many *device* field of an *Order*.

#### Parameters:

- **sender** (*Model*) – The intermediate model for the many-to-many relationship.
- **instance** (*Order*) – The Order instance to which devices are being added.
- **action** (*str*) – The type of change. We only respond to 'post\_add'.
- **pk\_set** (*set*) – Primary keys of the devices that were added.
- **\*\*kwargs** – Additional keyword arguments provided by the signal system.

#### Behavior:

- For each newly added device, dynamically imports the corresponding device model class.
- Retrieves the device-specific *fields* definition.
- Ensures a *MaintenanceProtocol* is created or fetched for the order/device combination.

## App.tests module

## App.urls module

## App.views module

## Module contents

## AuthUser package

## Subpackages

## AuthUser.migrations package

## Submodules

## AuthUser.migrations.0001\_initial module



```

class AuthUser.migrations.0001_initial.Migration (name, app_label)
    Bases: Migration

    dependencies = [('Entity', '0001_initial'), ('auth', '0012_alter_user_first_name_max_length')]

    initial = True

    operations = [<CreateModel name='Client', fields=[('id', <django.db.models.fields.BigAutoField>), ('role',
<django.db.models.fields.CharField>), ('entity', <django.db.models.fields.related.ForeignKey>)],
options={'verbose_name': 'Cliente', 'verbose_name_plural': 'Clientes'}>, <CreateModel name='Engineer',
fields=[('id', <django.db.models.fields.BigAutoField>), ('password', <django.db.models.fields.CharField>),
('last_login', <django.db.models.fields.DateTimeField>), ('is_superuser', <django.db.models.fields.BooleanField>),
('username', <django.db.models.fields.CharField>), ('first_name', <django.db.models.fields.CharField>),
('last_name', <django.db.models.fields.CharField>), ('email', <django.db.models.fields.EmailField>), ('is_staff',
<django.db.models.fields.BooleanField>), ('is_active', <django.db.models.fields.BooleanField>), ('date_joined',
<django.db.models.fields.DateTimeField>), ('role', <django.db.models.fields.CharField>), ('signature',
<django.db.models.fields.TextField>), ('entity', <django.db.models.fields.related.ForeignKey>), ('groups',
<django.db.models.fields.related.ManyToManyField>), ('user_permissions',
<django.db.models.fields.related.ManyToManyField>)], options={'verbose_name': 'Ingeniero',
'verbose_name_plural': 'Ingenieros'}, managers=[('objects', <django.contrib.auth.models.UserManager object>)]]>]

```

### AuthUser.migrations.0002\_client\_first\_name\_client\_last\_name module

```

class AuthUser.migrations.0002_client_first_name_client_last_name.Migration (name,
app_label)
    Bases: Migration

    dependencies = [('AuthUser', '0001_initial')]

    operations = [
        <AddField model_name='client', name='first_name',
field=<django.db.models.fields.CharField>>,
        <AddField model_name='client', name='last_name',
field=<django.db.models.fields.CharField>>]

```

### Module contents

### Submodules

### AuthUser.admin module

### AuthUser.apps module

```

class AuthUser.apps.UserConfig (app_name, app_module)
    Bases: AppConfig

    default_auto_field = 'django.db.models.BigAutoField'

    name = 'AuthUser'

```

### AuthUser.forms module

Forms for the AuthUser app.

Includes customizations for the Django authentication form.

```

class AuthUser.forms.CustomLoginForm (request=None, *args, **kwargs)
    Bases: AuthenticationForm

    Custom login form that applies Tailwind CSS classes for styling.

```

Inherits from Django's built-in *AuthenticationForm* and overrides the default widget attributes for the username and password fields.

```
base_fields = {'password': <django.forms.fields.CharField object>, 'username': <django.forms.fields.CharField object>}
```

```
declared_fields = {'password': <django.forms.fields.CharField object>, 'username': <django.forms.fields.CharField object>}
```

**property** `media`

Return all media required to render the widgets on this form.

## AuthUser.models module

Models for the AuthUser app.

Includes custom user models for Engineers and Clients who are associated with entities.

```
class AuthUser.models.Client(*args, **kwargs)
```

Bases: **EnterpriseUser**

Model representing a client user.

**first\_name**

Client's first name.

**Type:** str

**last\_name**

Client's last name.

**Type:** str

**exception** `DoesNotExist`

Bases: `ObjectDoesNotExist`

**exception** `MultipleObjectsReturned`

Bases: `MultipleObjectsReturned`

**entity**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via `ForwardOneToOneDescriptor` subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Child.parent` is a `ForwardManyToOneDescriptor` instance.

**entity\_id**

**first\_name**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

```
get_role_display(*, field=<django.db.models.fields.CharField: role>)
```

**id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**last\_name**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**objects** = <django.db.models.manager.Manager object>

#### **order\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Parent.children is a ReverseManyToOneDescriptor instance.

Most of the implementation is delegated to a dynamically defined manager class built by create\_forward\_many\_to\_many\_manager() defined below.

#### **role**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**class** AuthUser.models.**Engineer** (\*args, \*\*kwargs)

Bases: **EnterpriseUser**, **AbstractUser**

Model representing an engineer, extending the base Django user and EnterpriseUser.

#### **signature**

The engineer's signature (used in reports).

**Type:** str

#### **groups**

Groups the engineer belongs to.

**Type:** ManyToMany

#### **user\_permissions**

Permissions specific to the engineer.

**Type:** ManyToMany

#### **exception** DoesNotExist

Bases: **ObjectDoesNotExist**

#### **exception** MultipleObjectsReturned

Bases: **MultipleObjectsReturned**

#### **date\_joined**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

#### **email**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

#### **entity**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Child.parent is a ForwardManyToOneDescriptor instance.

#### **entity\_id**

#### **first\_name**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

```
get_next_by_date_joined (*, field=<django.db.models.fields.DateTimeField:
date_joined>, is_next=True, **kwargs)
```

```
get_previous_by_date_joined (*, field=<django.db.models.fields.DateTimeField:
date_joined>, is_next=False, **kwargs)
```

```
get_role_display(* ,field=<django.db.models.fields.CharField: role>)
```

### **groups**

Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation.  
In the example:

```
class Pizza(Model):
    toppings = ManyToManyField(Topping, related_name='pizzas')
```

`Pizza.toppings` and `Topping.pizzas` are `ManyToManyDescriptor` instances.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

### **id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

### **is\_active**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

### **is\_staff**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

### **is\_superuser**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

### **last\_login**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

### **last\_name**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

### **order\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.  
In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

### **password**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

### **role**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

#### signature

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

#### user\_permissions

Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation. In the example:

```
class Pizza(Model):
    toppings = ManyToManyField(Topping, related_name='pizzas')
```

`Pizza.toppings` and `Topping.pizzas` are `ManyToManyDescriptor` instances.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

#### username

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

```
class AuthUser.models.EnterpriseUser(*args, **kwargs)
```

Bases: `Model`

Abstract base model for users belonging to an entity (company or institution).

#### entity

The entity (organization) the user is associated with.

**Type:** `Entity`

#### role

The user's role within the entity, chosen from predefined choices.

**Type:** `str`

```
class Meta
```

Bases: `object`

`abstract = False`

```
ROLE_CHOICES = (('MAN', 'Gerente'), ('ENG', 'Ingeniero/a'), ('REP', 'Representante de ventas'), ('ADM', 'Administrativo'), ('BOS', 'Jefe/a de unidad'), ('SUP', 'Supervisor/a'), ('DOC', 'Doctor/a'), ('NUR', 'Enfermero/a'), ('TEN', 'Tens'))
```

#### entity

Accessor to the related object on the forward side of a many-to-one or one-to-one (via `ForwardOneToOneDescriptor` subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Child.parent` is a `ForwardManyToOneDescriptor` instance.

#### entity\_id

```
get_role_display(*, field=<django.db.models.fields.CharField: role>)
```

#### role

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

## AuthUser.serializers module

Serializers for the AuthUser app.

Includes serializers for the Engineer model.

**class** AuthUser.serializers.**EngineerSerializer** (\*args, \*\*kwargs)

Bases: **ModelSerializer**

Serializer for the Engineer model.

This serializer converts Engineer model instances into JSON format and vice versa.

**Fields:**

id (int): The unique identifier for the engineer. first\_name (str): The first name of the engineer. last\_name (str): The last name of the engineer. email (str): The email address of the engineer. role (str): The role of the engineer within the entity. signature (str): The engineer's signature (used in reports).

**class** Meta

Bases: **object**

**fields** = ['id', 'first\_name', 'last\_name', 'email', 'role', 'signature']

**model**

alias of **Engineer**

## AuthUser.tests module

Test cases for the AuthUser app models.

Includes tests for the Engineer and Client models, ensuring proper creation, relationships, and functionality of their fields.

**class** AuthUser.tests.**ClientModelTestCase** (methodName='runTest')

Bases: **TestCase**

Test case for the Client model.

This test case covers the creation and validation of a Client instance, ensuring that the client's fields, such as role and entity relationship, are correctly set.

**setUp ()**

Set up the necessary data for the tests.

Creates an entity and a client instance for use in the test cases.

**test\_client\_creation ()**

Test the creation of a client.

Verifies that the client instance has been created with the correct role and entity.

**class** AuthUser.tests.**EngineerModelTestCase** (methodName='runTest')

Bases: **TestCase**

Test case for the Engineer model.

This test case covers the creation and validation of an Engineer instance, including its relationships and attributes such as username, role, signature, and the entity it belongs to.

**setUp ()**

Set up the necessary data for the tests.

Creates an entity and an engineer instance for use in the test cases.

**test\_engineer\_creation ()**

Test the creation of an engineer.

Verifies that the engineer instance has been created correctly with the expected attributes.

**test\_engineer\_groups ()**

Test adding groups to an engineer.

Verifies that the engineer can have multiple groups and that groups can be correctly added to the engineer's group set.

## AuthUser.urls module

URL routing for the AuthUser app.

Defines the API endpoints related to the Engineer model, registering the EngineerViewSet with the DefaultRouter and including the generated URLs.

## AuthUser.views module

Views for the AuthUser app.

Defines a viewset for managing Engineer instances via the Django REST framework.

```
class AuthUser.views.EngineerViewSet (**kwargs)
    Bases: ModelViewSet
    ViewSet for handling Engineer model operations.
    This ViewSet provides list, create, retrieve, update, and destroy actions for the Engineer model. It uses a partial
    update by default to simplify PATCH support.
```

```
basename = None
```

```
description = None
```

```
detail = None
```

```
name = None
```

```
permission_classes = [<class 'rest_framework.permissions.IsAuthenticated'>]
```

```
queryset
```

```
serializer_class
    alias of EngineerSerializer
```

```
suffix = None
```

```
update (request, *args, **kwargs)
    Override the default update behavior to allow partial updates by default.
    This means clients don't need to specify all required fields in a PUT request.
```

## Module contents

## Device package

## Subpackages

## Device.migrations package

## Submodules

## Device.migrations.0001\_initial module

```
class Device.migrations.0001_initial.Migration (name, app_label)
    Bases: Migration
```

```
dependencies = [('Entity', '0001_initial')]
```

```
initial = True
```

```
operations = [<CreateModel name='DeviceModel', fields=[('id', <django.db.models.fields.BigAutoField>),
('device_type', <django.db.models.fields.CharField>), ('device_gen', <django.db.models.fields.CharField>),
('part_number', <django.db.models.fields.CharField>)], options={'verbose_name': 'Modelo', 'verbose_name_plural':
'Modelos'}>, <CreateModel name='Device', fields=[('id', <django.db.models.fields.BigAutoField>), ('serial_number',
<django.db.models.fields.CharField>), ('client', <django.db.models.fields.related.ForeignKey>), ('contract',
<django.db.models.fields.related.ForeignKey>), ('device_model', <django.db.models.fields.related.ForeignKey>)],
options={'verbose_name': 'Equipo', 'verbose_name_plural': 'Equipos'}>]
```

## Device.migrations.0002\_alter\_devicemodel\_device\_gen module

```
class Device.migrations.0002_alter_devicemodel_device_gen.Migration(name, app_label)
Bases: Migration
```

```
dependencies = [('Device', '0001_initial')]
```

```
operations = [(<AlterField model_name='devicemodel', name='device_gen',
field=<django.db.models.fields.CharField>)]
```

## Module contents

## Submodules

## Device.admin module

## Device.apps module

```
class Device.apps.DeviceConfig(app_name, app_module)
Bases: AppConfig
```

```
default_auto_field = 'django.db.models.BigAutoField'
```

```
name = 'Device'
```

## Device.forms module

## Device.models module

```
class Device.models.Device(*args, **kwargs)
```

Bases: **Model**

Stores a specific device instance tied to a client and contract.

### Fields:

client: Entity that owns the device. contract: Related contract if applicable. device\_model: Link to the device's generic model info. serial\_number: Unique serial number for identification.

```
exception DoesNotExist
```

Bases: **ObjectDoesNotExist**

```
exception MultipleObjectsReturned
```

Bases: **MultipleObjectsReturned**

```
client
```



Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Child.parent is a ForwardManyToOneDescriptor instance.

#### **client\_id**

#### **contract**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Child.parent is a ForwardManyToOneDescriptor instance.

#### **contract\_id**

#### **device\_model**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Child.parent is a ForwardManyToOneDescriptor instance.

#### **device\_model\_id**

#### **id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

#### **included\_devices**

Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation.

In the example:

```
class Pizza(Model):
    toppings = ManyToManyField(Topping, related_name='pizzas')
```

Pizza.toppings and Topping.pizzas are ManyToManyDescriptor instances.

Most of the implementation is delegated to a dynamically defined manager class built by create\_forward\_many\_to\_many\_manager() defined below.

#### **maintenanceprotocol\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Parent.children is a ReverseManyToOneDescriptor instance.

Most of the implementation is delegated to a dynamically defined manager class built by create\_forward\_many\_to\_many\_manager() defined below.

**objects** = <django.db.models.manager.Manager object>

#### **serial\_number**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**class** `Device.models.DeviceModel` (\*args, \*\*kwargs)

Bases: `Model`

Stores generic information about a device model.

**Fields:**

`device_type`: Category of the device (e.g., processor, endoscope). `device_gen`: Device generation or product series. `part_number`: Manufacturer part number or model identifier.

**exception** `DoesNotExist`

Bases: `ObjectDoesNotExist`

`GENERATION_CHOICES` = `[('OPT', 'Optera (170)'), ('EX3', 'Exera III (190)'), ('EX2', 'Exera II (180)'), ('EX1', 'Exera I (160)'), ('LUC', 'Evis Lucera (270)'), ('LUE', 'Evis Lucera Elite (290)'), ('EV3', 'Evis Exera III (190)'), ('EV2', 'Evis Exera II (180)'), ('VSR', 'Visera (150/160)'), ('VSP', 'Visera Pro'), ('VSE', 'Visera Elite'), ('VS2', 'Visera Elite II'), ('X1P', 'X1 Series'), ('OIP', 'Procesador de imágenes'), ('CV1', 'Serie CV-100'), ('CV2', 'Serie CV-200'), ('CV3', 'Serie CV-300'), ('CV4', 'Serie CV-400'), ('OES', 'Serie OES'), ('PRO', 'Serie PRO'), ('ULT', 'Serie ULTRA'), ('FIB', 'Fibroscopio de fibra óptica'), ('NOG', 'Sin generacion')]`

**exception** `MultipleObjectsReturned`

Bases: `MultipleObjectsReturned`

`TYPE_CHOICES` = `[('END', 'Endoscopio flexible'), ('VPR', 'Procesador de video'), ('LIS', 'Fuente de luz'), ('PMP', 'Bomba de irrigación'), ('INF', 'Insuflador de CO2'), ('MON', 'Monitor de grado médico'), ('WST', 'Estación de trabajo'), ('TEL', 'Telescopio'), ('ECG', 'Unidad de electrocirugía'), ('UDI', 'Sistema de documentación')]`

**device\_gen**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**device\_type**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

`get_device_gen_display` (\*, field=<django.db.models.fields.CharField: device\_gen>)

`get_device_type_display` (\*, field=<django.db.models.fields.CharField: device\_type>)

**id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**model**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

`objects` = <django.db.models.manager.Manager object>

**part\_number**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

## Device.serializers module

**class** Device.serializers.**DeviceModelSerializer** (\*args, \*\*kwargs)

Bases: **ParentSerializer**

Serializer for the DeviceModel model. Only includes minimal fields (ID and part number).

**class** **Meta**

Bases: **object**

**fields** = ['id', 'part\_number']

**model**

alias of **DeviceModel**

**class** Device.serializers.**DeviceSerializer** (\*args, \*\*kwargs)

Bases: **ModelSerializer**

Read-only serializer for Device model. Includes human-readable client and device model names.

**class** **Meta**

Bases: **object**

**fields** = ['id', 'client\_name', 'device\_model\_name', 'serial\_number']

**model**

alias of **Device**

## Device.tests module

**class** Device.tests.**DeviceModelTestCase** (methodName='runTest')

Bases: **TestCase**

**test\_device\_model\_creation()**

Test creation of a DeviceModel instance and its string representation.

**class** Device.tests.**DeviceTestCase** (methodName='runTest')

Bases: **TestCase**

**setUp()**

Create common Entity, Contract, and DeviceModel for device creation test.

**test\_device\_creation()**

Test creation of a Device and its relationships and string output.

**Device.urls module****Device.views module****Module contents****Entity package****Subpackages****Entity.migrations package****Submodules****Entity.migrations.0001\_initial module**

```

class Entity.migrations.0001_initial.Migration (name, app_label)
    Bases: Migration

    dependencies = []

    initial = True

    operations = [<CreateModel name='Entity', fields=[('id', <django.db.models.fields.BigAutoField>), ('name',
<django.db.models.fields.CharField>), ('address', <django.db.models.fields.CharField>)],
options={'verbose_name': 'Entidad', 'verbose_name_plural': 'Entidades'}>, <CreateModel name='Contract',
fields=[('id', <django.db.models.fields.BigAutoField>), ('number', <django.db.models.fields.PositiveIntegerField>),
('contract_type', <django.db.models.fields.CharField>), ('start_date', <django.db.models.fields.DateField>),
('end_date', <django.db.models.fields.DateField>), ('entity', <django.db.models.fields.related.ForeignKey>)],
options={'verbose_name': 'Contrato', 'verbose_name_plural': 'Contratos'}>, <CreateModel name='Area',
fields=[('id', <django.db.models.fields.BigAutoField>), ('name', <django.db.models.fields.CharField>), ('entity',
<django.db.models.fields.related.ForeignKey>)], options={'verbose_name': 'Area', 'verbose_name_plural':
'Areas'}>]

```

**Module contents****Submodules****Entity.admin module****Entity.apps module**

```

class Entity.apps.EntityConfig (app_name, app_module)
    Bases: AppConfig

    default_auto_field = 'django.db.models.BigAutoField'

    name = 'Entity'

```

**Entity.models module**

```

class Entity.models.Area (*args, **kwargs)
    Bases: Model

```

Entity area entity: Entity name: Area name

**exception** DoesNotExist

Bases: ObjectDoesNotExist

**exception** MultipleObjectsReturned

Bases: MultipleObjectsReturned

**entity**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Child.parent is a ForwardManyToOneDescriptor instance.

**entity\_id**

**id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**name**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**objects** = <django.db.models.manager.Manager object>

**class** Entity.models.Contract (\*args, \*\*kwargs)

Bases: Model

Entity's contract information entity: Entity information number: Contract number contract\_type: Contract type (Warranty/Post sales) start\_date: Start date end\_date: End date conditions\*: File to check detailed information

CONTRACT\_CHOICES = [('GT', 'Garantia'), ('PV', 'Post Venta')]

**exception** DoesNotExist

Bases: ObjectDoesNotExist

**exception** MultipleObjectsReturned

Bases: MultipleObjectsReturned

**contract\_type**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**device\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Parent.children is a ReverseManyToOneDescriptor instance.

Most of the implementation is delegated to a dynamically defined manager class built by create\_forward\_many\_to\_many\_manager() defined below.

**end\_date**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**entity**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Child.parent is a ForwardManyToOneDescriptor instance.

**entity\_id**

```
get_contract_type_display (*, field=<django.db.models.fields.CharField:
contract_type>)
```

```
get_next_by_end_date (*, field=<django.db.models.fields.DateField: end_date>,
is_next=True, **kwargs)
```

```
get_next_by_start_date (*, field=<django.db.models.fields.DateField: start_date>,
is_next=True, **kwargs)
```

```
get_previous_by_end_date (*, field=<django.db.models.fields.DateField: end_date>,
is_next=False, **kwargs)
```

```
get_previous_by_start_date (*, field=<django.db.models.fields.DateField: start_date>,
is_next=False, **kwargs)
```

**id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**number**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**objects** = <django.db.models.manager.Manager object>

**order\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Parent.children is a ReverseManyToOneDescriptor instance.

Most of the implementation is delegated to a dynamically defined manager class built by create\_forward\_many\_to\_many\_manager() defined below.

**start\_date**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

```
class Entity.models.Entity(*args, **kwargs)
```

Bases: Model

Hospital/Clinic/Institute, stores its information name: Entity's name address: Entity's address

```
exception DoesNotExist
```

Bases: ObjectDoesNotExist

```
exception MultipleObjectsReturned
```

Bases: MultipleObjectsReturned

**address**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**area\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

**client\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

**contract\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

**device\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

**engineer\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

**id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**name**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

`objects = <django.db.models.manager.Manager object>`

#### `order_set`

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

## Entity.serializers module

**class** `Entity.serializers.AreaSerializer(*args, **kwargs)`

Bases: `ModelSerializer`

Serializer for the Area model.

Embeds the full *EntitySerializer* for read-only display of related entity information.

**class** `Meta`

Bases: `object`

`fields = '__all__'`

`model`

alias of `Area`

**class** `Entity.serializers.ContractSerializer(*args, **kwargs)`

Bases: `ModelSerializer`

Serializer for the Contract model.

Includes basic contract fields along with the related entity's name as *client\_name*.

**class** `Meta`

Bases: `object`

`fields = ['id', 'number', 'client_name', 'contract_type', 'start_date', 'end_date']`

`model`

alias of `Contract`

**class** `Entity.serializers.EntitySerializer(*args, **kwargs)`

Bases: `ModelSerializer`

Serializer for the Entity model.

Provides basic fields such as name and address of a hospital, clinic, or institute.

**class** `Meta`

Bases: `object`

`fields = ['id', 'name', 'address']`

`model`

alias of `Entity`

## Entity.tests module



```
class Entity.tests.AreaModelTest (methodName='runTest')
```

Bases: **TestCase**

Test case for the Area model.

```
test_area_creation ()
```

Test that an Area instance is correctly created and linked to an Entity, and the string representation matches its name.

```
class Entity.tests.ContractModelTest (methodName='runTest')
```

Bases: **TestCase**

Test case for the Contract model.

```
test_contract_creation ()
```

Test that a Contract instance is created with valid data, is linked to an Entity, and its string representation is correct.

```
class Entity.tests.EntityModelTest (methodName='runTest')
```

Bases: **TestCase**

Test case for the Entity model.

```
test_entity_creation ()
```

Test that an Entity instance is created correctly and string representation matches its name.

## Entity.urls module

## Entity.views module

```
class Entity.views.AreaViewSet (**kwargs)
```

Bases: **ModelViewSet**

Provides CRUD operations for Area instances.

```
basename = None
```

```
description = None
```

```
detail = None
```

```
name = None
```

```
queryset
```

```
serializer_class
```

alias of **AreaSerializer**

```
suffix = None
```

```
class Entity.views.ContractViewSet (**kwargs)
```

Bases: **ModelViewSet**

Provides CRUD operations for Contract instances. Uses `select_related` to optimize queries involving the related Entity.

```
basename = None
```

```
description = None
```

```
detail = None
```

```
name = None
```

**queryset**

**serializer\_class**

alias of `ContractSerializer`

**suffix** = *None*

**class** `Entity.views.EntityViewSet` (\*\*kwargs)

Bases: `ModelViewSet`

Provides CRUD operations for Entity instances.

**basename** = *None*

**description** = *None*

**detail** = *None*

**name** = *None*

**queryset**

**serializer\_class**

alias of `EntitySerializer`

**suffix** = *None*

## Module contents

## Main package

## Submodules

## Main.asgi module

ASGI config for Main project.

It exposes the ASGI callable as a module-level variable named `application`.

For more information on this file, see <https://docs.djangoproject.com/en/5.1/howto/deployment/asgi/>

## Main.settings module

```
Main.settings.INSTALLED_APPS = ['django.contrib.admin', 'django.contrib.auth', 'django.contrib.contenttypes',
'django.contrib.sessions', 'django.contrib.messages', 'django.contrib.staticfiles', 'django.contrib.admindocs',
'rest_framework', 'corsheaders', 'oauth2_provider', 'App', 'AuthUser', 'Entity', 'Device', 'Order', 'MicrosoftAuth']
```

```
AUTHENTICATION_BACKENDS = [
    'django.contrib.auth.backends.ModelBackend',
]
```

## Main.urls module

## Main.wsgi module

WSGI config for Main project.

It exposes the WSGI callable as a module-level variable named `application`.

For more information on this file, see <https://docs.djangoproject.com/en/5.1/howto/deployment/wsgi/>

## Module contents

## MicrosoftAuth package

## Submodules

### MicrosoftAuth.admin module

### MicrosoftAuth.apps module

```
class MicrosoftAuth.apps.AuthConfig (app_name, app_module)
    Bases: AppConfig

    default_auto_field = 'django.db.models.BigAutoField'

    name = 'MicrosoftAuth'
```

### MicrosoftAuth.authentication module

```
class MicrosoftAuth.authentication.microsoftOAuth2Authentication
    Bases: BaseAuthentication

    authenticate (request)
        Authenticate the request and return a two-tuple of (user, token).
```

### MicrosoftAuth.functions module

```
MicrosoftAuth.functions.base64url_decode (base64url)
MicrosoftAuth.functions.get_microsoft_public_keys ()
MicrosoftAuth.functions.jwk_to_pem (jwk)
MicrosoftAuth.functions.validate_microsoft_token (token)
```

### MicrosoftAuth.models module

### MicrosoftAuth.tests module

### MicrosoftAuth.urls module

### MicrosoftAuth.views module

```
MicrosoftAuth.views.error (request)
MicrosoftAuth.views.microsoft_callback (request)
MicrosoftAuth.views.microsoft_login (request: HttpRequest) → HttpResponseRedirect
    Initiates the Microsoft OAuth2 login flow based on the application type. Selects between ConfidentialClientApplication or PublicClientApplication depending on the app_type parameter.
```

**Parameters:**

- **request** (*HttpRequest*) – The Django HTTP request object.
- **app\_type** (*str*) – The type of the application. Use 'server' for server-side apps (ConfidentialClientApplication) or 'mobile' for client-side apps (PublicClientApplication). Default is 'server'.

**Returns:** A JSON response containing the authorization URL.  
**Return type:** JsonResponse  
**Raises:** **KeyError** – If required environment variables are missing.

#### Notes

- Ensure `MICROSOFT_CLIENT_ID`, `MICROSOFT_CLIENT_SECRET`, and `MICROSOFT_TENANT` are set in the environment.
- `server` app type requires `ConfidentialClientApplication` (with client secret).
- `mobile` app type uses `PublicClientApplication` (no client secret).
- The `auth_flow` stored in the session should not contain sensitive information.
- Consider storing the redirect URI in an environment variable instead of hardcoding it.

```
MicrosoftAuth.views.microsoft_logout (request: HttpRequest, app_type: str = 'server')
```

```
MicrosoftAuth.views.set_client ()
```

## Module contents

### Order package

#### Subpackages

#### Order.migrations package

#### Submodules

#### Order.migrations.0001\_initial module

```
class Order.migrations.0001_initial.Migration (name, app_label)
```

Bases: Migration

```
dependencies = [('AuthUser', '0001_initial'), ('Device', '0001_initial'), ('Entity', '0001_initial')]
```

```
initial = True
```

```
operations = [<CreateModel name='Order', fields=[('id', <django.db.models.fields.BigAutoField>),
('created_at', <django.db.models.fields.DateField>), ('client_sign', <django.db.models.fields.TextField>), ('status',
<django.db.models.fields.CharField>), ('client', <django.db.models.fields.related.ForeignKey>), ('client_auth_user',
<django.db.models.fields.related.ForeignKey>), ('contract', <django.db.models.fields.related.ForeignKey>),
('device', <django.db.models.fields.related.ManyToManyField>), ('engineer',
<django.db.models.fields.related.ForeignKey>)], options={'verbose_name': 'Orden de trabajo',
'verbosename_plural': 'Ordenes de trabajo'}>, <CreateModel name='MaintenanceProtocol', fields=[('id',
<django.db.models.fields.BigAutoField>), ('status', <django.db.models.fields.CharField>), ('location',
<django.db.models.fields.CharField>), ('fields', <django.db.models.fields.json.JSONField>), ('device',
<django.db.models.fields.related.ForeignKey>), ('order', <django.db.models.fields.related.ForeignKey>)],
options={'verbose_name': 'Protocolo', 'verbose_name_plural': 'Protocolos'}>]
```

#### Order.migrations.0002\_order\_client\_name module

```
class Order.migrations.0002_order_client_name.Migration (name, app_label)
```

Bases: Migration

```
dependencies = [('Order', '0001_initial')]
```

```
operations = [<AddField model_name='order', name='client_name',
field=<django.db.models.fields.CharField>>]
```

### Order.migrations.0003\_rename\_client\_name\_order\_client\_personal\_name module

```
class Order.migrations.0003_rename_client_name_order_client_personal_name.Migration
(name, app_label)
Bases: Migration

dependencies = [('Order', '0002_order_client_name')]

operations = [<RenameField model_name='order', old_name='client_name',
new_name='client_personal_name'>]
```

### Order.migrations.0004\_alter\_maintenanceprotocol\_location module

```
class Order.migrations.0004_alter_maintenanceprotocol_location.Migration (name,
app_label)
Bases: Migration

dependencies = [('Order', '0003_rename_client_name_order_client_personal_name')]

operations = [<AlterField model_name='maintenanceprotocol', name='location',
field=<django.db.models.fields.CharField>>]
```

### Order.migrations.0005\_alter\_maintenanceprotocol\_status\_and\_more module

```
class Order.migrations.0005_alter_maintenanceprotocol_status_and_more.Migration (name,
app_label)
Bases: Migration

dependencies = [('Order', '0004_alter_maintenanceprotocol_location')]

operations = [<AlterField model_name='maintenanceprotocol', name='status',
field=<django.db.models.fields.CharField>>, <AlterField model_name='order', name='client_personal_name',
field=<django.db.models.fields.CharField>>]
```

### Module contents

### Submodules

### Order.admin module

### Order.apps module

```
class Order.apps.OrderConfig (app_name, app_module)
Bases: AppConfig

default_auto_field = 'django.db.models.BigAutoField'

name = 'Order'
```

### Order.forms module

Forms for the Order app.

Includes `ModelForm` definitions for creating and updating `Order` instances.

```
class Order.forms.OrderForm (data=None, files=None, auto_id='id_%s', prefix=None,
initial=None, error_class=<class 'django.forms.utils.ErrorList'>, label_suffix=None,
empty_permitted=False, instance=None, use_required_attribute=None, renderer=None)
```

Bases: `ModelForm`

Form for creating and updating `Order` instances. Applies consistent styling to each field using Tailwind CSS classes.

```
class Meta
```

Bases: `object`

```
fields = ['client', 'contract', 'device', 'status', 'engineer']
```

```
model
```

alias of `Order`

```
widgets = {'client': <django.forms.widgets.Select object>, 'contract': <django.forms.widgets.Select object>,
'device': <django.forms.widgets.SelectMultiple object>, 'engineer': <django.forms.widgets.Select object>,
'status': <django.forms.widgets.Select object>}
```

```
base_fields = {'client': <django.forms.models.ModelChoiceField object>, 'contract':
<django.forms.models.ModelChoiceField object>, 'device': <django.forms.models.ModelMultipleChoiceField
object>, 'engineer': <django.forms.models.ModelChoiceField object>, 'status':
<django.forms.fields.TypedChoiceField object>}
```

```
declared_fields = {}
```

```
property media
```

Return all media required to render the widgets on this form.

## Order.models module

Models for managing maintenance orders and protocols.

Includes: - `Order`: Represents a work order for one or more devices. - `MaintenanceProtocol`: Stores the maintenance report for a device in an order.

```
class Order.models.MaintenanceProtocol (*args, **kwargs)
```

Bases: `Model`

Stores the results of a maintenance protocol for a specific device within an order.

**Fields:**

`device`: The device being serviced. `order`: The order under which this protocol is executed. `status`: Final operational status of the device after maintenance. `location`: Physical location of the device during maintenance. `fields`: Flexible data for recording custom protocol fields (e.g., measurements, checklists).

```
exception DoesNotExist
```

Bases: `ObjectDoesNotExist`

```
MAINTENANCE_STATUS = [('SR', 'Sin revisar'), ('EPEN', 'Equipo pendiente'), ('OPSO', 'Equipo operativo sin
observaciones'), ('OPCO', 'Equipo operativo con observaciones'), ('SSTT', 'Equipo se encuentra en servicio
técnico'), ('ERBJ', 'Equipo reportado de baja'), ('ENDB', 'Equipo no disponible')]
```

```
exception MultipleObjectsReturned
```

Bases: `MultipleObjectsReturned`

```
device
```

Accessor to the related object on the forward side of a many-to-one or one-to-one (via `ForwardOneToOneDescriptor` subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Child.parent is a ForwardManyToOneDescriptor instance.

**device\_id**

**fields**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**get\_status\_display**(\*, field=<django.db.models.fields.CharField: status>)

**id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**location**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**objects** = <django.db.models.manager.Manager object>

**order**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Child.parent is a ForwardManyToOneDescriptor instance.

**order\_id**

**status**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**class** Order.models.**Order** (\*args, \*\*kwargs)

Bases: **Model**

Represents a work order assigned to an engineer for one or more devices.

**Fields:**

engineer: The engineer responsible for the order. client: The institution or client receiving the service. contract: Optional contract linked to the order. client\_AuthUser: The authenticated client user, if applicable. device: Devices included in the order. created\_at: Timestamp of order creation. client\_personal\_name: Name of the client representative. client\_sign: Signature of the client representative. status: Current status of the order (Scheduled, Completed, Delayed).

**exception** DoesNotExist

Bases: **ObjectDoesNotExist**

**exception** MultipleObjectsReturned

Bases: **MultipleObjectsReturned**

**ORDER\_STATUS** = [('PR', 'Programada'), ('CT', 'Completada'), ('AT', 'Atradasa')]

**client**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Child.parent` is a `ForwardManyToOneDescriptor` instance.

#### **client\_AuthUser**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via `ForwardOneToOneDescriptor` subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Child.parent` is a `ForwardManyToOneDescriptor` instance.

#### **client\_AuthUser\_id**

#### **client\_id**

#### **client\_personal\_name**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

#### **client\_sign**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

#### **contract**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via `ForwardOneToOneDescriptor` subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Child.parent` is a `ForwardManyToOneDescriptor` instance.

#### **contract\_id**

#### **created\_at**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

#### **device**

Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation.

In the example:

```
class Pizza(Model):
    toppings = ManyToManyField(Topping, related_name='pizzas')
```

`Pizza.toppings` and `Topping.pizzas` are `ManyToManyDescriptor` instances.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

#### **engineer**

Accessor to the related object on the forward side of a many-to-one or one-to-one (via `ForwardOneToOneDescriptor` subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```



`Child.parent` is a `ForwardManyToOneDescriptor` instance.

**engineer\_id**

**get\_next\_by\_created\_at** (\*, field=<django.db.models.fields.DateField: created\_at>, is\_next=True, \*\*kwargs)

**get\_previous\_by\_created\_at** (\*, field=<django.db.models.fields.DateField: created\_at>, is\_next=False, \*\*kwargs)

**get\_status\_display** (\*, field=<django.db.models.fields.CharField: status>)

**id**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

**maintenanceprotocol\_set**

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

**objects** = <django.db.models.manager.Manager object>

**status**

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

## Order.serializers module

Serializers for Orders and Maintenance Protocols.

Includes: - `MaintenanceListSerializer`: Summary view of maintenance protocol. - `MaintenanceSerializer`: Detailed view of maintenance protocol. - `OrderListSerializer`: Lightweight serializer for listing orders. - `OrderSerializer`: Full order details with nested maintenance protocols.

**class** `Order.serializers.MaintenanceListSerializer` (\*args, \*\*kwargs)

Bases: `ParentSerializer`

A lightweight serializer for listing maintenance protocols. Includes basic device and order info.

**class** `Meta`

Bases: `object`

**fields** = ['id', 'order', 'device\_type', 'device\_model', 'device\_serial', 'device\_contract', 'location', 'status']

**model**

alias of `MaintenanceProtocol`

**class** `Order.serializers.MaintenanceSerializer` (\*args, \*\*kwargs)

Bases: `ParentSerializer`

A detailed serializer for individual maintenance protocols. Includes editable custom fields.

**class** `Meta`

Bases: `object`

**fields** = ['id', 'order', 'device\_type', 'device\_model', 'device\_serial', 'device\_contract', 'location', 'status', 'fields']

```

model
    alias of MaintenanceProtocol

```

```

class Order.serializers.OrderListSerializer(*args, **kwargs)
    Bases: ModelSerializer
    A simplified serializer used for listing orders.

```

```

class Meta
    Bases: object

```

```

fields = ['id', 'client_name', 'client_address', 'created_at', 'status', 'client_sign', 'client_personal_name']

```

```

model
    alias of Order

```

```

class Order.serializers.OrderSerializer(*args, **kwargs)
    Bases: ModelSerializer
    A detailed serializer for a single order. Includes related maintenance protocols.

```

```

class Meta
    Bases: object

```

```

fields = ['id', 'client_name', 'client_address', 'created_at', 'status', 'maintenance_protocols', 'client_sign',
'client_personal_name']

```

```

model
    alias of Order

```

```

get_maintenance_protocols(obj)
    Returns all maintenance protocols associated with this order. Uses a simplified serializer but includes all fields.

```

## Order.tests module

Unit tests for the Order and MaintenanceProtocol models.

```

class Order.tests.OrderModelTest(methodName='runTest')
    Bases: TestCase
    Test case for Order and MaintenanceProtocol model behaviors.

```

```

setUp()
    Create related models for testing: - Entity (client) - Engineer - Client AuthUser - Contract - DeviceModel - Device
    - Order (with M2M device) - MaintenanceProtocol

```

```

test_order_str_representation()
    Test the __str__ method of the Order model.

```

```

test_protocol_str_representation()
    Test the __str__ method of the MaintenanceProtocol model.

```

## Order.urls module

URL routing configuration for the 'Order' domain.

This module defines API routes for working with maintenance orders and protocols, including viewsets for read/write operations, as well as utility endpoints for querying devices, contracts, and related protocols.

### Routes:

- 'order/' → DRF ViewSet for Order model (CRUD operations)
- 'protocol/' → DRF ViewSet for MaintenanceProtocol model
- 'orders/all/' → List all orders

- 'orders/filter/' → Filter orders by query parameters
- 'orders/<order\_id>/' → Retrieve an order by ID
- 'orders/create/' → Create a new order
- 'orders/<order\_pk>/update/' → Update an existing order
- 'orders/<order\_pk>/delete/' → Delete an order
- 'orders/<order\_pk>/protocols/' → Get all protocols linked to an order
- 'devices/by-client/' → Retrieve all devices linked to a client
- 'contracts/by-client/' → Retrieve all contracts linked to a client

## Order.views module

## Module contents

## manage module

Django's command-line utility for administrative tasks.

```
manage.main()
```

Run administrative tasks.



# Index

## A

[abstract](#) ([AuthUser.models.EnterpriseUser.Meta](#) attribute)

[address](#) ([Entity.models.Entity](#) attribute)

### App

[module](#)

### App.admin

[module](#)

### App.apps

[module](#)

### App.constants

[module](#)

### App.models

[module](#)

### App.serializers

[module](#)

### App.signals

[module](#)

### App.tests

[module](#)

[AppConfig](#) (class in [App.apps](#))

[Area](#) (class in [Entity.models](#))

[Area.DoesNotExist](#)

[Area.MultipleObjectsReturned](#)

[area\\_set](#) ([Entity.models.Entity](#) attribute)

[AreaModelTest](#) (class in [Entity.tests](#))

[AreaSerializer](#) (class in [Entity.serializers](#))

[AreaSerializer.Meta](#) (class in [Entity.serializers](#))

[AreaViewSet](#) (class in [Entity.views](#))

[AuthConfig](#) (class in [MicrosoftAuth.apps](#))

[authenticate\(\)](#) ([MicrosoftAuth.authentication.microsoftO](#)  
[auth2Authentication](#) method)

### AuthUser

[module](#)

### AuthUser.admin

[module](#)

### AuthUser.apps

[module](#)

### AuthUser.forms

[module](#)

### AuthUser.migrations

[module](#)

### AuthUser.migrations.0001\_initial

[module](#)

### AuthUser.migrations.0002\_client\_first\_name\_client\_last\_name

[module](#)

### AuthUser.models

[module](#)

### AuthUser.serializers

[module](#)

### AuthUser.tests

[module](#)

### AuthUser.urls

[module](#)

### AuthUser.views

[module](#)

## B

[base64url\\_decode\(\)](#) (in [MicrosoftAuth.functions](#) [module](#))

[base\\_fields](#) ([AuthUser.forms.CustomLoginForm](#) attribute)

([Order.forms.OrderForm](#) attribute)

[basename](#) ([AuthUser.views.EngineerViewSet](#) attribute)

([Entity.viewsAreaViewSet](#) attribute)

([Entity.views.ContractViewSet](#) attribute)

([Entity.views.EntityViewSet](#) attribute)

## C

[Client](#) (class in [AuthUser.models](#))

[client](#) ([Device.models.Device](#) attribute)

([Order.models.Order](#) attribute)

[Client.DoesNotExist](#)

[Client.MultipleObjectsReturned](#)

[client\\_AuthUser](#) ([Order.models.Order](#) attribute)

[client\\_AuthUser\\_id](#) ([Order.models.Order](#) attribute)

[client\\_id](#) ([Device.models.Device](#) attribute)

([Order.models.Order](#) attribute)

[client\\_personal\\_name](#) ([Order.models.Order](#) attribute)

[client\\_set](#) ([Entity.models.Entity](#) attribute)

[client\\_sign](#) ([Order.models.Order](#) attribute)

[ClientModelTestCase](#) (class in [AuthUser.tests](#))

[Contract](#) (class in [Entity.models](#))

[contract](#) ([Device.models.Device](#) attribute)

([Order.models.Order](#) attribute)

[Contract.DoesNotExist](#)

[Contract.MultipleObjectsReturned](#)

CONTRACT\_CHOICES (Entity.models.Contract attribute)  
contract\_id (Device.models.Device attribute)  
(Order.models.Order attribute)  
contract\_set (Entity.models.Entity attribute)  
contract\_type (Entity.models.Contract attribute)  
ContractModelTest (class in Entity.tests)  
ContractSerializer (class in Entity.serializers)  
ContractSerializer.Meta (class in Entity.serializers)  
ContractViewSet (class in Entity.views)  
create\_protocols\_for\_order() (in module App.signals)  
created\_at (Order.models.Order attribute)  
CustomLoginForm (class in AuthUser.forms)

## D

date\_joined (AuthUser.models.Engineer attribute)  
declared\_fields (AuthUser.forms.CustomLoginForm attribute)  
(Order.forms.OrderForm attribute)  
default\_auto\_field (App.apps.AppConfig attribute) [1]  
(AuthUser.apps.UserConfig attribute)  
(Device.apps.DeviceConfig attribute)  
(Entity.apps.EntityConfig attribute)  
(MicrosoftAuth.apps.AuthConfig attribute)  
(Order.apps.OrderConfig attribute)  
dependencies  
(AuthUser.migrations.0001\_initial.Migration attribute)  
(AuthUser.migrations.0002\_client\_first\_name\_client\_last\_name.Migration attribute)  
(Device.migrations.0001\_initial.Migration attribute)  
(Device.migrations.0002\_alter\_devicemodel\_device\_gen.Migration attribute)  
(Entity.migrations.0001\_initial.Migration attribute)  
(Order.migrations.0001\_initial.Migration attribute)  
(Order.migrations.0002\_order\_client\_name.Migration attribute)  
(Order.migrations.0003\_rename\_client\_name\_order\_client\_personal\_name.Migration attribute)  
(Order.migrations.0004\_alter\_maintenanceprotocol\_location.Migration attribute)  
(Order.migrations.0005\_alter\_maintenanceprotocol\_status\_and\_more.Migration attribute)  
description (AuthUser.views.EngineerViewSet attribute)  
(Entity.views.AreaViewSet attribute)  
(Entity.views.ContractViewSet attribute)

(Entity.views.EntityViewSet attribute)  
detail (AuthUser.views.EngineerViewSet attribute)  
(Entity.views.AreaViewSet attribute)  
(Entity.views.ContractViewSet attribute)  
(Entity.views.EntityViewSet attribute)  
**Device**  
module  
Device (class in Device.models)  
device (Order.models.MaintenanceProtocol attribute)  
(Order.models.Order attribute)  
**Device.admin**  
module  
**Device.apps**  
module  
Device.DoesNotExist  
**Device.forms**  
module  
**Device.migrations**  
module  
**Device.migrations.0001\_initial**  
module  
**Device.migrations.0002\_alter\_devicemodel\_device\_gen**  
module  
**Device.models**  
module  
Device.MultipleObjectsReturned  
**Device.serializers**  
module  
**Device.tests**  
module  
**Device.urls**  
module  
**Device.views**  
module  
device\_gen (Device.models.DeviceModel attribute)  
device\_id (Order.models.MaintenanceProtocol attribute)  
device\_model (Device.models.Device attribute)  
device\_model\_id (Device.models.Device attribute)  
device\_set (Entity.models.Contract attribute)  
(Entity.models.Entity attribute)  
device\_type (Device.models.DeviceModel attribute)  
DeviceConfig (class in Device.apps)  
DeviceModel (class in Device.models)  
DeviceModel.DoesNotExist

DeviceModel.MultipleObjectsReturned  
DeviceModelSerializer (class in Device.serializers)  
DeviceModelSerializer.Meta (class in Device.serializers)  
DeviceModelTestCase (class in Device.tests)  
DeviceSerializer (class in Device.serializers)  
DeviceSerializer.Meta (class in Device.serializers)  
DeviceTestCase (class in Device.tests)

## E

email (AuthUser.models.Engineer attribute)  
end\_date (Entity.models.Contract attribute)  
Engineer (class in AuthUser.models)  
engineer (Order.models.Order attribute)  
Engineer.DoesNotExist  
Engineer.MultipleObjectsReturned  
engineer\_id (Order.models.Order attribute)  
engineer\_set (Entity.models.Entity attribute)  
EngineerModelTestCase (class in AuthUser.tests)  
EngineerSerializer (class in AuthUser.serializers)  
EngineerSerializer.Meta (class in AuthUser.serializers)  
EngineerViewSet (class in AuthUser.views)  
EnterpriseUser (class in AuthUser.models)  
EnterpriseUser.Meta (class in AuthUser.models)

### Entity

module  
entity (AuthUser.models.Client attribute)  
(AuthUser.models.Engineer attribute)  
(AuthUser.models.EnterpriseUser attribute) [1]  
Entity (class in Entity.models)  
entity (Entity.models.Area attribute)  
(Entity.models.Contract attribute)

### Entity.admin

module

### Entity.apps

module

Entity.DoesNotExist

### Entity.migrations

module

### Entity.migrations.0001\_initial

module

### Entity.models

module

Entity.MultipleObjectsReturned

### Entity.serializers

module

### Entity.tests

module

### Entity.urls

module

### Entity.views

module

entity\_id (AuthUser.models.Client attribute)  
(AuthUser.models.Engineer attribute)  
(AuthUser.models.EnterpriseUser attribute)  
(Entity.models.Area attribute)  
(Entity.models.Contract attribute)  
EntityConfig (class in Entity.apps)  
EntityModelTest (class in Entity.tests)  
EntitySerializer (class in Entity.serializers)  
EntitySerializer.Meta (class in Entity.serializers)  
EntityViewSet (class in Entity.views)  
error() (in module MicrosoftAuth.views)

## F

fields (AuthUser.serializers.EngineerSerializer.Meta attribute)  
(Device.serializers.DeviceModelSerializer.Meta attribute)  
(Device.serializers.DeviceSerializer.Meta attribute)  
(Entity.serializers.AreaSerializer.Meta attribute)  
(Entity.serializers.ContractSerializer.Meta attribute)  
(Entity.serializers.EntitySerializer.Meta attribute)  
(Order.forms.OrderForm.Meta attribute)  
(Order.models.MaintenanceProtocol attribute)  
(Order.serializers.MaintenanceListSerializer.Meta attribute)  
(Order.serializers.MaintenanceSerializer.Meta attribute)  
(Order.serializers.OrderListSerializer.Meta attribute)  
(Order.serializers.OrderSerializer.Meta attribute)  
first\_name (AuthUser.models.Client attribute) [1]  
(AuthUser.models.Engineer attribute)

## G

### GENERATION\_CHOICES

(Device.models.DeviceModel attribute)

get\_contract\_type\_display() (Entity.models.Contract method)

`get_device_gen_display()` (Device.models.DeviceModel method)

`get_device_type_display()` (Device.models.DeviceModel method)

`get_maintenance_protocols()` (Order.serializers.OrderSerializer method)

`get_microsoft_public_keys()` (in module MicrosoftAuth.functions)

`get_next_by_created_at()` (Order.models.Order method)

`get_next_by_date_joined()` (AuthUser.models.Engineer method)

`get_next_by_end_date()` (Entity.models.Contract method)

`get_next_by_start_date()` (Entity.models.Contract method)

`get_previous_by_created_at()` (Order.models.Order method)

`get_previous_by_date_joined()` (AuthUser.models.Engineer method)

`get_previous_by_end_date()` (Entity.models.Contract method)

`get_previous_by_start_date()` (Entity.models.Contract method)

`get_role_display()` (AuthUser.models.Client method)

(AuthUser.models.Engineer method)

(AuthUser.models.EnterpriseUser method)

`get_status_display()` (Order.models.MaintenanceProtocol method)

(Order.models.Order method)

`groups` (AuthUser.models.Engineer attribute) [1]

## I

`id` (AuthUser.models.Client attribute)

(AuthUser.models.Engineer attribute)

(Device.models.Device attribute)

(Device.models.DeviceModel attribute)

(Entity.models.Area attribute)

(Entity.models.Contract attribute)

(Entity.models.Entity attribute)

(Order.models.MaintenanceProtocol attribute)

(Order.models.Order attribute)

`included_devices` (Device.models.Device attribute)

`initial` (AuthUser.migrations.0001\_initial.Migration attribute)

(Device.migrations.0001\_initial.Migration attribute)

(Entity.migrations.0001\_initial.Migration attribute)

(Order.migrations.0001\_initial.Migration attribute)

`INSTALLED_APPS` (in module Main.settings)

`is_active` (AuthUser.models.Engineer attribute)

`is_staff` (AuthUser.models.Engineer attribute)

`is_superuser` (AuthUser.models.Engineer attribute)

## J

`jwk_to_pem()` (in module MicrosoftAuth.functions)

## L

`last_login` (AuthUser.models.Engineer attribute)

`last_name` (AuthUser.models.Client attribute) [1]

(AuthUser.models.Engineer attribute)

`location` (Order.models.MaintenanceProtocol attribute)

## M

### Main

module

`main()` (in module manage)

### Main.asgi

module

### Main.settings

module

### Main.wsgi

module

### MAINTENANCE\_STATUS

(Order.models.MaintenanceProtocol attribute)

`MaintenanceListSerializer` (class in Order.serializers)

`MaintenanceListSerializer.Meta` (class in Order.serializers)

`MaintenanceProtocol` (class in Order.models)

`MaintenanceProtocol.DoesNotExist`

`MaintenanceProtocol.MultipleObjectsReturned`

`maintenanceprotocol_set` (Device.models.Device attribute)

(Order.models.Order attribute)

`MaintenanceSerializer` (class in Order.serializers)

`MaintenanceSerializer.Meta` (class in Order.serializers)

### manage

module

`media` (AuthUser.forms.CustomLoginForm property)

(Order.forms.OrderForm property)

`microsoft_callback()` (in module MicrosoftAuth.views)

`microsoft_login()` (in module MicrosoftAuth.views)

`microsoft_logout()` (in module MicrosoftAuth.views)



<b>MicrosoftAuth</b>	(Order.serializers.MaintenanceSerializer.Meta attribute)
module	
<b>MicrosoftAuth.admin</b>	(Order.serializers.OrderListSerializer.Meta attribute)
module	
<b>MicrosoftAuth.apps</b>	(Order.serializers.OrderSerializer.Meta attribute)
module	
<b>MicrosoftAuth.authentication</b>	<b>module</b>
module	App
<b>MicrosoftAuth.functions</b>	App.admin
module	App.apps
<b>MicrosoftAuth.models</b>	App.constants
module	App.models
<b>MicrosoftAuth.tests</b>	App.serializers
module	App.signals
<b>MicrosoftAuth.urls</b>	App.tests
module	AuthUser
<b>MicrosoftAuth.views</b>	AuthUser.admin
module	AuthUser.apps
microsoftOAuth2Authentication (class in MicrosoftAuth.authentication)	AuthUser.forms
Migration (class in AuthUser.migrations.0001_initial)	AuthUser.migrations
(class in AuthUser.migrations.0002_client_first_name_client_last_name)	AuthUser.migrations.0001_initial
(class in Device.migrations.0001_initial)	AuthUser.migrations.0002_client_first_name_client_last_name
(class in Device.migrations.0002_alter_devicemodel_device_gen)	AuthUser.models
(class in Entity.migrations.0001_initial)	AuthUser.serializers
(class in Order.migrations.0001_initial)	AuthUser.tests
(class in Order.migrations.0002_order_client_name)	AuthUser.urls
(class in Order.migrations.0003_rename_client_name_order_client_personal_name)	AuthUser.views
(class in Order.migrations.0004_alter_maintenanceprotocol_location)	Device
(class in Order.migrations.0005_alter_maintenanceprotocol_status_and_more)	Device.admin
model (AuthUser.serializers.EngineerSerializer.Meta attribute)	Device.apps
(Device.models.DeviceModel attribute)	Device.forms
(Device.serializers.DeviceModelSerializer.Meta attribute)	Device.migrations
(Device.serializers.DeviceSerializer.Meta attribute)	Device.migrations.0001_initial
(Entity.serializers.AreaSerializer.Meta attribute)	Device.migrations.0002_alter_devicemodel_device_gen
(Entity.serializers.ContractSerializer.Meta attribute)	Device.models
(Entity.serializers.EntitySerializer.Meta attribute)	Device.serializers
(Order.forms.OrderForm.Meta attribute)	Device.tests
(Order.serializers.MaintenanceListSerializer.Meta attribute)	Device.urls
	Device.views
	Entity
	Entity.admin
	Entity.apps

Entity.migrations  
Entity.migrations.0001\_initial  
Entity.models  
Entity.serializers  
Entity.tests  
Entity.urls  
Entity.views  
Main  
Main.asgi  
Main.settings  
Main.wsgi  
manage  
MicrosoftAuth  
MicrosoftAuth.admin  
MicrosoftAuth.apps  
MicrosoftAuth.authentication  
MicrosoftAuth.functions  
MicrosoftAuth.models  
MicrosoftAuth.tests  
MicrosoftAuth.urls  
MicrosoftAuth.views  
Order  
Order.admin  
Order.apps  
Order.forms  
Order.migrations  
Order.migrations.0001\_initial  
Order.migrations.0002\_order\_client\_name  
Order.migrations.0003\_rename\_client\_name\_order\_client\_personal\_name  
Order.migrations.0004\_alter\_maintenanceprotocol\_location  
Order.migrations.0005\_alter\_maintenanceprotocol\_status\_and\_more  
Order.models  
Order.serializers  
Order.tests  
Order.urls  
Order.views

## N

name (App.apps.AppConfig attribute) [1]  
(AuthUser.apps.UserConfig attribute)

(AuthUser.views.EngineerViewSet attribute)  
(Device.apps.DeviceConfig attribute)  
(Entity.apps.EntityConfig attribute)  
(Entity.models.Area attribute)  
(Entity.models.Entity attribute)  
(Entity.viewsAreaViewSet attribute)  
(Entity.views.ContractViewSet attribute)  
(Entity.views.EntityViewSet attribute)  
(MicrosoftAuth.apps.AuthConfig attribute)  
(Order.apps.OrderConfig attribute)  
number (Entity.models.Contract attribute)

## O

objects (AuthUser.models.Client attribute)  
(Device.models.Device attribute)  
(Device.models.DeviceModel attribute)  
(Entity.models.Area attribute)  
(Entity.models.Contract attribute)  
(Entity.models.Entity attribute)  
(Order.models.MaintenanceProtocol attribute)  
(Order.models.Order attribute)  
operations (AuthUser.migrations.0001\_initial.Migration attribute)  
(AuthUser.migrations.0002\_client\_first\_name\_client\_last\_name.Migration attribute)  
(Device.migrations.0001\_initial.Migration attribute)  
(Device.migrations.0002\_alter\_devicemodel\_device\_gen.Migration attribute)  
(Entity.migrations.0001\_initial.Migration attribute)  
(Order.migrations.0001\_initial.Migration attribute)  
(Order.migrations.0002\_order\_client\_name.Migration attribute)  
(Order.migrations.0003\_rename\_client\_name\_order\_client\_personal\_name.Migration attribute)  
(Order.migrations.0004\_alter\_maintenanceprotocol\_location.Migration attribute)  
(Order.migrations.0005\_alter\_maintenanceprotocol\_status\_and\_more.Migration attribute)

### Order

module

Order (class in Order.models)

order (Order.models.MaintenanceProtocol attribute)

### Order.admin

module

### Order.apps

[module](#)

[Order.DoesNotExist](#)

**Order.forms**

[module](#)

**Order.migrations**

[module](#)

**Order.migrations.0001\_initial**

[module](#)

**Order.migrations.0002\_order\_client\_name**

[module](#)

**Order.migrations.0003\_rename\_client\_name\_order\_client\_personal\_name**

[module](#)

**Order.migrations.0004\_alter\_maintenanceprotocol\_location**

[module](#)

**Order.migrations.0005\_alter\_maintenanceprotocol\_status\_and\_more**

[module](#)

**Order.models**

[module](#)

[Order.MultipleObjectsReturned](#)

**Order.serializers**

[module](#)

**Order.tests**

[module](#)

**Order.urls**

[module](#)

**Order.views**

[module](#)

[order\\_id](#) ([Order.models.MaintenanceProtocol](#) attribute)

[order\\_set](#) ([AuthUser.models.Client](#) attribute)

[\(AuthUser.models.Engineer](#) attribute)

[\(Entity.models.Contract](#) attribute)

[\(Entity.models.Entity](#) attribute)

[ORDER\\_STATUS](#) ([Order.models.Order](#) attribute)

[OrderConfig](#) (class in [Order.apps](#))

[OrderForm](#) (class in [Order.forms](#))

[OrderForm.Meta](#) (class in [Order.forms](#))

[OrderListSerializer](#) (class in [Order.serializers](#))

[OrderListSerializer.Meta](#) (class in [Order.serializers](#))

[OrderModelTest](#) (class in [Order.tests](#))

[OrderSerializer](#) (class in [Order.serializers](#))

[OrderSerializer.Meta](#) (class in [Order.serializers](#))

## P

[ParentSerializer](#) (class in [App.serializers](#))

[part\\_number](#) ([Device.models.DeviceModel](#) attribute)

[password](#) ([AuthUser.models.Engineer](#) attribute)

[permission\\_classes](#) ([AuthUser.views.EngineerViewSet](#) attribute)

## Q

[queryset](#) ([AuthUser.views.EngineerViewSet](#) attribute)

[\(Entity.views.AreaViewSet](#) attribute)

[\(Entity.views.ContractViewSet](#) attribute)

[\(Entity.views.EntityViewSet](#) attribute)

## R

[ready\(\)](#) ([App.apps.AppConfig](#) method)

[role](#) ([AuthUser.models.Client](#) attribute)

[\(AuthUser.models.Engineer](#) attribute)

[\(AuthUser.models.EnterpriseUser](#) attribute) [1]

[ROLE\\_CHOICES](#) ([AuthUser.models.EnterpriseUser](#) attribute)

## S

[serial\\_number](#) ([Device.models.Device](#) attribute)

[serializer\\_class](#) ([AuthUser.views.EngineerViewSet](#) attribute)

[\(Entity.views.AreaViewSet](#) attribute)

[\(Entity.views.ContractViewSet](#) attribute)

[\(Entity.views.EntityViewSet](#) attribute)

[set\\_client\(\)](#) (in module [MicrosoftAuth.views](#))

[setUp\(\)](#) ([AuthUser.tests.ClientModelTestCase](#) method)

[\(AuthUser.tests.EngineerModelTestCase](#) method)

[\(Device.tests.DeviceTestCase](#) method)

[\(Order.tests.OrderModelTest](#) method)

[signature](#) ([AuthUser.models.Engineer](#) attribute) [1]

[start\\_date](#) ([Entity.models.Contract](#) attribute)

[status](#) ([Order.models.MaintenanceProtocol](#) attribute)

[\(Order.models.Order](#) attribute)

[suffix](#) ([AuthUser.views.EngineerViewSet](#) attribute)

[\(Entity.views.AreaViewSet](#) attribute)

[\(Entity.views.ContractViewSet](#) attribute)

[\(Entity.views.EntityViewSet](#) attribute)

## T

`test_area_creation()` (Entity.tests.AreaModelTest method)

`test_client_creation()`  
(AuthUser.tests.ClientModelTestCase method)

`test_contract_creation()`  
(Entity.tests.ContractModelTest method)

`test_device_creation()` (Device.tests.DeviceTestCase method)

`test_device_model_creation()`  
(Device.tests.DeviceModelTestCase method)

`test_engineer_creation()`  
(AuthUser.tests.EngineerModelTestCase method)

`test_engineer_groups()`  
(AuthUser.tests.EngineerModelTestCase method)

`test_entity_creation()` (Entity.tests.EntityModelTest method)

`test_order_str_representation()`  
(Order.tests.OrderModelTest method)

`test_protocol_str_representation()`  
(Order.tests.OrderModelTest method)

`TYPE_CHOICES` (Device.models.DeviceModel attribute)

## U

`update()` (AuthUser.views.EngineerViewSet method)

`user_permissions` (AuthUser.models.Engineer attribute)  
[1]

`UserConfig` (class in AuthUser.apps)

`username` (AuthUser.models.Engineer attribute)

## V

`validate_microsoft_token()` (in module  
MicrosoftAuth.functions)

## W

`widgets` (Order.forms.OrderForm.Meta attribute)

# Python Module Index

## a

- App
- App.admin
- App.apps
- App.constants
- App.models
- App.serializers
- App.signals
- App.tests
- AuthUser
- AuthUser.admin
- AuthUser.apps
- AuthUser.forms
- AuthUser.migrations
- AuthUser.migrations.0001\_initial
- AuthUser.migrations.0002\_client\_first\_name\_client\_last\_name
- AuthUser.models
- AuthUser.serializers
- AuthUser.tests
- AuthUser.urls
- AuthUser.views

## d

- Device
- Device.admin
- Device.apps
- Device.forms
- Device.migrations
- Device.migrations.0001\_initial
- Device.migrations.0002\_alter\_devicemodel\_device\_generated\_name
- Device.models
- Device.serializers
- Device.tests
- Device.urls
- Device.views

## e

- Entity
- Entity.admin

- Entity.apps
- Entity.migrations
- Entity.migrations.0001\_initial
- Entity.models
- Entity.serializers
- Entity.tests
- Entity.urls
- Entity.views

## m

- Main
- Main.asgi
- Main.settings
- Main.wsgi
- manage
- MicrosoftAuth
- MicrosoftAuth.admin
- MicrosoftAuth.apps
- MicrosoftAuth.authentication
- MicrosoftAuth.functions
- MicrosoftAuth.models
- MicrosoftAuth.tests
- MicrosoftAuth.urls
- MicrosoftAuth.views

## o

- Order
- Order.admin
- Order.apps
- Order.forms
- Order.migrations
- Order.migrations.0001\_initial
- Order.migrations.0002\_order\_client\_name
- Order.migrations.0003\_rename\_client\_name\_order\_client\_personal\_name
- Order.migrations.0004\_alter\_maintenanceprotocol\_location
- Order.migrations.0005\_alter\_maintenanceprotocol\_status\_and\_more
- Order.models
- Order.serializers
- Order.tests
- Order.urls

