djangotutor

Выпуск 0.0.1

Arsen Sardaryan

мар. 19, 2025

Contents:

Add your content using `reStructuredText` syntax. See the reStructuredText documentation for details.

Глава 1

djangotutor

class polls.models.Choice(*id*, *question*, *choice_text*, *votes*)

Базовые классы: Model

Параметры

- id (*BigAutoField*) – Primary key: ID
- choice_text (*CharField*) – Choice text
- votes (*IntegerField*) – Votes

Relationship fields:

Параметры

question (ForeignKey to *Question*) – Question (related name: choice)

exception DoesNotExist

Базовые классы: ObjectDoesNotExist

exception MultipleObjectsReturned

Базовые классы: MultipleObjectsReturned

_meta = <Options for Choice>

choice_text

Type: CharField

Choice text

id

Type: BigAutoField

Primary key: ID

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

question

Type: ForeignKey to *Question*

Question (related name: choice)

question_id

Internal field, use *question* instead.

votes

Type: IntegerField

Votes

class polls.models.Monitoring(*monitoring_id*, *cpu_usage*, *memory_usage*, *network_traffic*, *monitoring_time*, *virtual_machines*)

Базовые классы: Model

Параметры

- monitoring_id (*UUIDField*) – Primary key: Monitoring id
- cpu_usage (*FloatField*) – Cpu usage
- memory_usage (*FloatField*) – Memory usage
- network_traffic (*TimeField*) – Network traffic
- monitoring_time (*DateTimeField*) – Monitoring time

Relationship fields:

Параметры
virtual_machines (ForeignKey to *Virtuals*) – Virtual machines (related name: monitoring)

exception DoesNotExist

Базовые классы: ObjectDoesNotExist

exception MultipleObjectsReturned

Базовые классы: MultipleObjectsReturned

_meta = <Options for Monitoring>

cpu_usage

Type: FloatField

Cpu usage

get_next_by_monitoring_time(*, *field=<django.db.models.DateTimeField: monitoring_time>*, *is_next=True*, ***kwargs*)

Finds next instance based on *monitoring_time*. See get_next_by_FOO() for more information.

get_previous_by_monitoring_time(*, *field=<django.db.models.DateTimeField: monitoring_time>*, *is_next=False*, ***kwargs*)

Finds previous instance based on *monitoring_time*. See get_previous_by_FOO() for more information.

memory_usage

Type: FloatField

Memory usage

monitoring_id
: Type: UUIDField

  Primary key: Monitoring id

monitoring_time
: Type: DateTimeField

  Monitoring time

network_traffic
: Type: TimeField

  Network traffic

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

virtual_machines
: Type: ForeignKey to *Virtuals*

  Virtual machines (related name: monitoring)

virtual_machines_id
: Internal field, use *virtual_machines* instead.

class polls.models.Product(*id*, *name*, *price*)

Базовые классы: Model

Параметры

- id (*BigAutoField*) – Primary key: ID
- name (*CharField*) – Name
- price (*DecimalField*) – Price

exception DoesNotExist
: Базовые классы: ObjectDoesNotExist

exception MultipleObjectsReturned
: Базовые классы: MultipleObjectsReturned

__init__()

_meta = <Options for Product>

id
: Type: BigAutoField

  Primary key: ID

name
: Type: CharField

  Name

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

price
: Type: DecimalField

  Price

qux

Doc comment for instance attribute qux.

spam

Docstring for instance attribute spam.

class polls.models.Protocol(*values)

Базовые классы: TextChoices

RDP = 'rdp'

SSH = 'ssh'

VNC = 'vnc'

static _generate_next_value_(*name*, *start*, *count*, *last_values*)

Generate the next value when not given.

name: the name of the member start: the initial start value or None count: the number of existing members last_values: the list of values assigned

class polls.models.Question(*id*, *question_text*, *pub_date*)

Базовые классы: Model

Параметры

- id (*BigAutoField*) – Primary key: ID
- question_text (*CharField*) – Question text
- pub_date (*DateTimeField*) – Date published

Reverse relationships:

Параметры

choice (Reverse ForeignKey from *Choice*) – All choices of this question (related name of *question*)

exception DoesNotExist

Базовые классы: ObjectDoesNotExist

exception MultipleObjectsReturned

Базовые классы: MultipleObjectsReturned

_meta = <Options for Question>

choice_set

Type: Reverse ForeignKey from *Choice*

All choices of this question (related name of *question*)

get_next_by_pub_date(*, *field=<django.db.models.DateTimeField: pub_date>*, *is_next=True*, ***kwargs*)

Finds next instance based on *pub_date*. See get_next_by_FOO() for more information.

get_previous_by_pub_date(*, *field=<django.db.models.DateTimeField: pub_date>*, *is_next=False*, ***kwargs*)

Finds previous instance based on *pub_date*. See get_previous_by_FOO() for more information.

id

Type: BigAutoField

Primary key: ID

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

pub_date

Type: DateTimeField

Date published

question_text

Type: CharField

Question text

was_published_recently()

class polls.models.Virtuals(*id*, *hostname*, *protocol*, *user_id*, *address*, *port*, *user_vm*, *password_vm*, *ignore_cert*)

Базовые классы: Model

Параметры

- id (*UUIDField*) – Primary key: Id
- hostname (*CharField*) – Hostname
- protocol (*CharField*) – Protocol
- address (*CharField*) – Address
- port (*IntegerField*) – Port
- user_vm (*CharField*) – User vm
- password_vm (*CharField*) – Password vm
- ignore_cert (*BooleanField*) – Ignore cert

Relationship fields:

Параметры

user_id (ForeignKey to User) – User id (related name: virtuals)

Reverse relationships:

Параметры

monitoring (Reverse ForeignKey from *Monitoring*) – All monitorings of this virtuals (related name of *virtual_machines*)

exception DoesNotExist

Базовые классы: ObjectDoesNotExist

exception MultipleObjectsReturned

Базовые классы: MultipleObjectsReturned

_meta = <Options for Virtuals>

address

Type: CharField

Address

get_protocol_display(*, *field=<django.db.models.CharField: protocol>*)

Shows the label of the *protocol*. See get_FOO_display() for more information.

hostname

Type: CharField

Hostname

id

Type: UUIDField

Primary key: Id

ignore_cert

Type: BooleanField

Ignore cert

monitoring_set

Type: Reverse ForeignKey from *Monitoring*

All monitorings of this virtuals (related name of *virtual_machines*)

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

password_vm

Type: CharField

Password vm

port

Type: IntegerField

Port

protocol

Type: CharField

Protocol

Choices:

- rdp
- ssh
- vnc

user_id

Type: ForeignKey to User

User id (related name: virtuals)

user_id_id

Internal field, use *user_id* instead.

user_vm

Type: CharField

User vm

class polls.views.MonitoringDetailView(***kwargs*)

Базовые классы: APIView

delete(*request*, *monitoring_id*)

get(*request*, *monitoring_id*)

class polls.views.MonitoringListView(***kwargs*)

Базовые классы: APIView

get(*request*)

post(*request*)

class polls.views.VirtualsDetailView(***kwargs*)

Базовые классы: APIView

delete(*request*, *vm_id*)

get(*request*, *vm_id*)

class polls.views.VirtualsListView(***kwargs*)

Базовые классы: APIView

get(*request*)

post(*request*)

p

Символы

A

C

D

G

H

I

M

N

O

P

Q

R

S

U

V

W