# **Camping Le Maine Blanc**

Release 1.0

**Nathalie Darnaudat** 

## **CONTENTS:**

1 Modules et Apps				
		Core		
		Tests Core		
	1.3	Reservations	14	
		Tests Reservations		
		Bookings		
	1.6	Tests Bookings	40	
Рy	thon I	Module Index	43	
In	dex		45	

Welcome to the Camping Le Maine Blanc site documentation

This documentation covers the Python modules, views, forms, admin, and tests of the project.

You can add your content using the reStructuredText syntax. For more details, see the documentation: reStructuredText

CONTENTS: 1

2 CONTENTS:

**CHAPTER** 

ONE

## **MODULES ET APPS**

## **1.1 Core**

Modèles

## class core.models.CampingInfo(\*args, \*\*kwargs)

Bases: TranslatableModel

Stores general camping rules and schedules.

Fields include reception hours, arrival and departure times, and gate opening hours.

#### **Security:**

• No direct user input, so XSS or SQL injection risk is minimal.

#### translations

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.

#### exception DoesNotExist

Bases: ObjectDoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

#### arrivals\_end\_high

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## arrivals\_end\_low

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## arrivals\_start\_high

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### departure\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### id

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

#### portal\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## portal\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### welcome\_afternoon\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### welcome\_afternoon\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### welcome\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### welcome\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### class core.models.SwimmingPoolInfo(\*args, \*\*kwargs)

Bases: TranslatableModel

Stores swimming pool schedule information.

## **Security:**

• No user input processed.

#### translations

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.

## exception DoesNotExist

Bases: ObjectDoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

id

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

#### pool\_opening\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### pool\_opening\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### class core.models.FoodInfo(\*args, \*\*kwargs)

Bases: TranslatableModel

Stores information about food services such as food trucks, pizza days, bread reservations, and bar opening hours.

#### **Translations:**

- Automatically translates burger and pizza days using DeepL API.
- Logs translation errors without interrupting save process.

#### **Security:**

- No user input is directly processed here, reducing XSS risk.
- DeepL API errors are caught and logged.

#### translations

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.

```
save(*args, **kwargs)
```

Overrides save to automatically translate certain fields into multiple languages using DeepL API. Errors are logged but do not interrupt saving.

#### **Security:**

• Only controlled default values are translated; no user input is processed.

1.1. Core 5

#### exception DoesNotExist

Bases: ObjectDoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

#### bar\_hours\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### bar\_hours\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### bread\_hours\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### bread\_hours\_reservations

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### bread\_hours\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### burger\_food\_days

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### burger\_food\_hours\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## burger\_food\_hours\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### id

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

## pizza\_food\_days

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## class core.models.LaundryInfo(\*args, \*\*kwargs)

Bases: TranslatableModel

Stores information about laundry services and pricing.

#### **Security:**

• Only fixed numeric values; no user input.

#### translations

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.

#### exception DoesNotExist

Bases: ObjectDoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

## dryer\_price

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

id

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

## washing\_machine\_price

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

Bases: TranslatedFieldsModel

#### exception DoesNotExist

Bases: TranslationDoesNotExist, DoesNotExist, DoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

## arrivals\_end\_high

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

#### arrivals\_end\_low

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

## arrivals\_start\_high

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

1.1. Core 7

#### departure\_end

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

get\_language\_code\_display(\*, field=<parler.utils.compat.HideChoicesCharField: language\_code>)

#### id

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

#### language\_code

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

#### master

The mandatory Foreign key field to the shared model.

#### master\_id

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

#### portal\_end

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

#### portal\_start

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

#### welcome\_afternoon\_end

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

#### welcome\_afternoon\_start

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

## welcome\_end

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

#### welcome\_start

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

bread\_hours\_reservations, bread\_hours\_start, bread\_hours\_end, bar\_hours\_start, bar\_hours\_end, master)

Bases: TranslatedFieldsModel

#### exception DoesNotExist

Bases: TranslationDoesNotExist, DoesNotExist, DoesNotExist

#### exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

#### bar\_hours\_end

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

#### bar\_hours\_start

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

#### bread\_hours\_end

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

## bread\_hours\_reservations

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

## bread\_hours\_start

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

## burger\_food\_days

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

#### burger\_food\_hours\_end

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

## burger\_food\_hours\_start

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

get\_language\_code\_display(\*, field=<parler.utils.compat.HideChoicesCharField: language\_code>)

#### id

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

## language\_code

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

#### master

The mandatory Foreign key field to the shared model.

## master\_id

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

#### pizza\_food\_days

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

Bases: TranslatedFieldsModel

1.1. Core 9

#### exception DoesNotExist

Bases: TranslationDoesNotExist, DoesNotExist, DoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

#### dryer\_price

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

get\_language\_code\_display(\*, field=<parler.utils.compat.HideChoicesCharField: language\_code>)

id

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

#### language\_code

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

#### master

The mandatory Foreign key field to the shared model.

#### master\_id

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

## washing\_machine\_price

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

Bases: TranslatedFieldsModel

#### exception DoesNotExist

Bases: TranslationDoesNotExist, DoesNotExist, DoesNotExist

#### exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

get\_language\_code\_display(\*, field=<parler.utils.compat.HideChoicesCharField: language\_code>)

id

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

#### language\_code

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

#### master

The mandatory Foreign key field to the shared model.

#### master\_id

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

## pool\_opening\_end

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

#### pool\_opening\_start

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

• Vues

## core.views.home\_view(request)

Render the homepage.

#### **Template:**

core/home.html

#### **Context:**

languages (list of tuples): list of supported languages and their flag filenames.

#### **Security:**

• No user input processed; safe from XSS or injection.

#### core.views.about\_view(request)

Render the about page.

#### **Security:**

• No user input processed.

#### core.views.infos\_view(request)

Render the information page with pricing, supplements, seasons, mobile homes, camping info, capacity, and other prices.

#### **Features:**

- · Groups normal and worker prices
- · Handles supplements and visitor prices
- · Dynamically sets mobile home descriptions based on current language

## **Security:**

- · Only retrieves data from the database
- · No user input is processed
- · Safe against XSS and injection

## core.views.services\_view(request)

Render the Services page including swimming pool, food, and laundry info.

#### **Security:**

- Only reads database objects
- · No user input processed

## core.views.accommodations\_view(request)

Render the Accommodations page.

#### **Security:**

• No user input processed

1.1. Core 11

```
core.views.activities_view(request)
     Render the Activities page.
     Security:
            • No user input processed
core.views.legal_view(request)
     Render the Legal page.
     Security:
            · Static content, safe
core.views.privacy_view(request)
     Render the Privacy Policy page.
     Security:
            · Static content, safe
core.views.not_found_view(request, exception=None)
     Render the 404 Not Found page.
     Security:
            • Static content
            • Exception handling passed safely
core.views.robots_txt(request)
     Serve the robots.txt file for search engines.
          Returns
              Plain text response with crawling rules for bots.
   • Admin
class core.apps.CoreConfig(app_name, app_module)
     Bases: AppConfig
     Configuration for the 'core' app, which manages general campsite information.
     default_auto_field = 'django.db.models.BigAutoField'
     name = 'core'
     verbose_name = 'Informations diverses'
   · Templates Tags
   • Sitemaps
class core.sitemaps.MultilingualStaticSitemap
     Bases: Sitemap
     changefreq = 'monthly'
     priority = 0.8
     pages = ['home', 'about', 'infos', 'services', 'accommodations', 'activities',
     'reservation_request', 'legal', 'privacy-policy']
```

```
languages = ['fr', 'en', 'es', 'de', 'nl']
     items()
          Returns a list of tuples (language, name_url) Example: [('fr', 'home'), ('en', 'home'), ...]
     location(item)
          Constructs the final URL with the language code
     alternates(item)
          Adds <xhtml:link> tags to indicate translated versions
1.2 Tests Core
   • Modèles
core.tests.test_models.test_campinginfo_fixture(campinginfo fr)
     Verify that the CampingInfo fixture is correctly created and times are set.
core.tests.test_models.test_swimmingpoolinfo_fixture(swimmingpoolinfo_fr)
     Verify that the SwimmingPoolInfo fixture is correctly created and opening times are correct.
core.tests.test_models.test_foodinfo_fixture(foodinfo_fr)
     Verify that the FoodInfo fixture is correctly created and food hours are set.
core.tests.test_models.test_laundryinfo_fixture(laundryinfo_fr)
     Verify that the LaundryInfo fixture is correctly created and prices are correct.
   • Vues
core.tests.test_views.test_home_view(client)
     Home page should load successfully and use the correct template.
core.tests.test_views.test_about_view(client)
     About page should load successfully and use the correct template.
core.tests.test_views.test_accommodations_view(client)
     Accommodations page should load successfully and use the correct template.
core.tests.test_views.test_activities_view(client)
     Activities page should load successfully and use the correct template.
core.tests.test_views.test_legal_view(client)
     Legal page should load successfully and use the correct template.
core.tests.test_views.test_privacy_view(client)
     Privacy policy page should load successfully and use the correct template.
core.tests.test_views.test_not_found_view(client)
     Not found page should return 404 and use the correct template.
core.tests.test_views.test_infos_view(client, campinginfo_fr, mobilehome_fr)
     Infos view should load successfully and provide camping info and mobilehome data.
core.tests.test_views.test_services_view(client, swimmingpoolinfo fr, foodinfo fr, laundryinfo fr)
     Services view should load successfully and provide swimming, food, and laundry info.
```

1.2. Tests Core

## 1.3 Reservations

• Vues

reservations.views.reservation\_request\_view(request)

## Handles reservation requests from users:

- GET: display empty reservation form
- POST: validate form, translate message, send email to admin, show success message

## **Security measures:**

- Form validation via ReservationRequestForm
- Escape user-provided message to prevent XSS
- Deepl API errors are caught and logged, original message preserved
- Email sending is wrapped, fail\_silently=False
- No sensitive data (API key) is exposed to templates
- · Formulaires

Bases: Form

Form to handle reservation requests from customers.

#### Fields:

- Customer info: name, first name, address, postal code, city, phone, email
- Reservation dates: start\_date, end\_date
- Accommodation details: accommodation\_type, tent dimensions, vehicle\_length
- Guests: adults, children\_over\_8, children\_under\_8, pets
- Electricity info: electricity, cable\_length
- Message: optional free text

## **Security:**

- Fields are validated and escaped to prevent XSS
- Conditional fields validated in clean()

```
ACCOMMODATION_CHOICES = [('', 'Choisissez un type'), ('tent', 'Tente'), ('car_tent', 'Voiture tente'), ('caravan', 'Caravane'), ('fourgon', 'Fourgon aménagé'), ('van', 'Van'), ('camping_car', 'Camping_car'), ('mobil-home', 'Mobil-home')]

ELECTRICITY_CHOICES = [('yes', 'Avec électricité'), ('no', 'Sans électricité')]
```

#### clean()

#### **Custom validation:**

- Validate dates (arrival < departure, not in past)
- Validate conditional fields (tent dimensions, vehicle length, cable if electricity)
- Escape free-text message to prevent XSS

```
base_fields = {'accommodation_type': <django.forms.fields.ChoiceField object>,
     'address': <django.forms.fields.CharField object>, 'adults':
     <django.forms.fields.ChoiceField object>, 'cable_length':
     <django.forms.fields.DecimalField object>, 'children_over_8':
    <django.forms.fields.ChoiceField object>, 'children_under_8':
    <django.forms.fields.ChoiceField object>, 'city': <django.forms.fields.CharField</pre>
    object>, 'electricity': <django.forms.fields.ChoiceField object>, 'email':
    <django.forms.fields.EmailField object>, 'end_date': <django.forms.fields.DateField</pre>
    object>, 'first_name': <django.forms.fields.CharField object>, 'message':
    <django.forms.fields.CharField object>, 'name': <django.forms.fields.CharField</pre>
    object>, 'pets': <django.forms.fields.ChoiceField object>, 'phone':
    <django.forms.fields.CharField object>, 'postal_code':
    <django.forms.fields.CharField object>, 'start_date':
    <django.forms.fields.DateField object>, 'tent_length':
    <django.forms.fields.DecimalField object>, 'tent_width':
    <django.forms.fields.DecimalField object>, 'vehicle_length':
    <django.forms.fields.DecimalField object>}
    declared_fields = {'accommodation_type': <django.forms.fields.ChoiceField object>,
     'address': <django.forms.fields.CharField object>, 'adults':
    <django.forms.fields.ChoiceField object>, 'cable_length':
    <django.forms.fields.DecimalField object>, 'children_over_8':
    <django.forms.fields.ChoiceField object>, 'children_under_8':
<django.forms.fields.ChoiceField object>, 'city': <django.forms.fields.CharField</pre>
    object>, 'electricity': <django.forms.fields.ChoiceField object>, 'email':
    <django.forms.fields.EmailField object>, 'end_date': <django.forms.fields.DateField</pre>
    object>, 'first_name': <django.forms.fields.CharField object>, 'message':
    <django.forms.fields.CharField object>, 'name': <django.forms.fields.CharField</pre>
    object>, 'pets': <diango.forms.fields.ChoiceField object>, 'phone':
    <django.forms.fields.CharField object>, 'postal_code':
    <django.forms.fields.CharField object>, 'start_date':
    <django.forms.fields.DateField object>, 'tent_length':
    <django.forms.fields.DecimalField object>, 'tent_width':
    <django.forms.fields.DecimalField object>, 'vehicle_length':
    <django.forms.fields.DecimalField object>}
    property media
         Return all media required to render the widgets on this form.

    Configuration

class reservations.apps.ReservationsConfig(app_name, app_module)
    Bases: AppConfig
```

1.3. Reservations 15

Configuration for the 'reservations' app, which handles booking requests and forms.

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

```
name = 'reservations'
```

## 1.4 Tests Reservations

Vues

reservations.tests.test\_views.valid\_reservation\_data()

Valid POST data for the reservation form

reservations.tests.test\_views.test\_get\_reservation\_request\_view(client)

GET request to reservation\_request returns the form and correct template.

 ${\tt reservations.tests.test\_views.test\_post\_valid\_reservation\_with\_datetime\_and\_label({\it mock\_translate}, {\it mock\_translate},$ 

mock\_send,
client,
valid\_reservation\_data)

POST valid reservation sends email, shows success, and sets submission\_datetime.

reservations.tests.test\_views.test\_post\_invalid\_reservation(client)

POST request with missing required fields returns form errors

Formulaires

reservations.tests.test\_forms.valid\_form\_data()

Valid data for the booking form

reservations.tests.test\_forms.test\_form\_valid(valid form data)

Form with all valid data should be valid.

reservations.tests.test\_forms.test\_form\_missing\_required\_field(valid\_form\_data)

Form missing 'name' field should be invalid and report error.

reservations.tests.test\_forms.test\_form\_dates\_invalid(valid\_form\_data)

Start date in the past or end date not after start should be invalid.

reservations.tests.test\_forms.test\_form\_tent\_requires\_dimensions(valid\_form\_data)

Tent accommodation requires both length and width to be filled.

reservations.tests.test\_forms.test\_form\_vehicle\_requires\_length(valid\_form\_data)

Vehicle-type accommodation requires vehicle\_length to be filled.

 $reservations. test\_form\_cable\_requires\_if\_electricity\_yes (\textit{valid\_form\_data})$ 

If electricity is 'yes', cable\_length must be filled.

## 1.5 Bookings

• Modèles

class bookings.models.SupplementPrice(\*args, \*\*kwargs)

Bases: Model

Stores additional pricing options for reservations.

#### Fields:

- extra\_adult\_price: price per extra adult
- child\_over\_8\_price: price per child over 8 years

- child\_under\_8\_price: price per child under 8 years
- pet\_price: price per pet
- extra\_vehicle\_price: price per additional vehicle
- extra\_tent\_price: price per additional tent
- visitor\_price\_without\_swimming\_pool: visitor price without pool
- visitor\_price\_with\_swimming\_pool: visitor price with pool

## **Security:**

- · Only stores numeric data
- · No user input processed directly

#### extra\_adult\_price

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

#### child\_over\_8\_price

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

#### child\_under\_8\_price

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

## pet\_price

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

#### extra\_vehicle\_price

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

#### extra\_tent\_price

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

## visitor\_price\_without\_swimming\_pool

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

#### visitor\_price\_with\_swimming\_pool

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

## exception DoesNotExist

Bases: ObjectDoesNotExist

## ${\tt exception~MultipleObjectsReturned}$

Bases: MultipleObjectsReturned

## id

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>

#### prices

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.

## class bookings.models.Price(\*args, \*\*kwargs)

Bases: Model

Stores the base pricing for different types of accommodations and seasons.

#### Fields:

- booking\_type: main type (tent, caravan, camping\_car)
- · season: low/mid/high
- is worker: boolean flag for worker rates
- price\_1\_person\_with\_electricity, price\_2\_persons\_with\_electricity
- price\_1\_person\_without\_electricity, price\_2\_persons\_without\_electricity
- supplements: related SupplementPrice

#### save()

auto-assigns included\_people and supplements

#### - clean()

validates camping-car pricing rules

#### **Security:**

- Only numeric and enum fields
- Validation ensures business rules are respected

```
SEASON_CHOICES = [('low', 'Basse Saison'), ('mid', 'Moyenne Saison'), ('high',
'Haute Saison')]
```

```
TYPE_CHOICES = [('tent', 'Tente / Voiture Tente'), ('caravan', 'Caravane / Fourgon /
Van'), ('camping_car', 'Camping-car'), ('other', 'Emplacement ouvrier weekend')]
```

#### booking\_type

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

## season

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

#### is\_worker

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

#### price\_1\_person\_with\_electricity

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

#### price\_2\_persons\_with\_electricity

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

## price\_1\_person\_without\_electricity

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

## price\_2\_persons\_without\_electricity

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

## included\_people

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

## worker\_week\_price

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

## weekend\_price\_without\_electricity

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

## weekend\_price\_with\_electricity

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

#### supplements

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

In the example:

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

Child.parent is a ForwardManyToOneDescriptor instance.

## save(\*args, \*\*kwargs)

Automatically assigns included\_people based on booking\_type and ensures a SupplementPrice is associated if missing.

## clean()

Validate business rules before saving. For camping\_car, ensures that '1 person' price fields are empty. Raises ValidationError on violation.

## exception DoesNotExist

Bases: ObjectDoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

get\_booking\_type\_display(\*, field=<django.db.models.fields.CharField: booking\_type>)

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

id

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>

## supplements\_id

```
class bookings.models.OtherPrice(*args, **kwargs)
```

Bases: TranslatableModel

Stores miscellaneous pricing info such as tourist tax.

#### Fields:

- · current\_year: current calendar year
- tourist tax date: date when tourist tax applies
- price\_tourist\_tax: price per night/person

## **Security:**

- Read-only for user input
- · Only numeric and date fields

#### translations

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.

#### exception DoesNotExist

Bases: ObjectDoesNotExist

#### exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

## current\_year

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

id

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

#### price\_tourist\_tax

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### tourist\_tax\_date

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## class bookings.models.Capacity(\*args, \*\*kwargs)

Bases: Model

Stores maximum capacity for each booking type.

#### Fields:

- booking\_type: type of accommodation
- max\_places: maximum allowed placements
- number\_locations: total locations
- number\_mobile\_homes: number of mobile homes

## **Security:**

- · Only numeric data
- Used for internal validation (check\_capacity)

## booking\_type

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

## max\_places

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

#### number\_locations

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

## number\_mobile\_homes

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

#### exception DoesNotExist

Bases: ObjectDoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

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

## id

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 bookings.models.Booking(\*args, \*\*kwargs)

Bases: Model

Stores client booking information.

#### Fields:

- Personal info: last\_name, first\_name, address, postal\_code, city, phone, email
- Reservation info: start\_date, end\_date, booking\_type, booking\_subtype, electricity
- Counts: adults, children\_over\_8, children\_under\_8, pets
- Extras: extra\_vehicle, extra\_tent, deposit\_paid
- Timestamps: created\_at, updated\_at

#### - get\_season()

determines season based on start\_date

#### - calculate\_total\_price()

calculates total cost including supplements

## - calculate\_deposit()

calculates 15% deposit

#### - save()

auto-assigns main type, included\_people, supplements

## - check\_capacity()

checks if capacity is available

## - clean()

validates business rules and capacity

#### **Security:**

- Input validation via clean() ensures business rules are respected
- Numeric and enum fields only; safe from injection
- check\_capacity prevents overbooking
- All calculations server-side

```
TYPE_CHOICES = [('tent', 'Tente / Voiture Tente'), ('caravan', 'Caravane / Fourgon /
Van'), ('camping_car', 'Camping-car'), ('other', 'Emplacement ouvrier weekend')]

SUBTYPE_CHOICES = [('tent', 'Tente'), ('car_tent', 'Voiture Tente'), ('caravan',
'Caravane'), ('fourgon', 'Fourgon'), ('van', 'Van'), ('camping_car', 'Camping-car')]

ELECTRICITY_CHOICES = [('yes', 'Avec électricité'), ('no', 'Sans électricité')]
```

#### last\_name

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

#### first\_name

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

#### address

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

## postal\_code

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

#### city

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

## phone

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.

#### start\_date

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

#### end\_date

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

#### booking\_type

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

## booking\_subtype

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

## electricity

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

## deposit\_paid

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

#### tent\_length

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

#### tent\_width

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

#### vehicle\_length

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

#### cable\_length

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

#### adults

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

#### children\_over\_8

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

#### children under 8

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

#### pets

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

#### extra\_vehicle

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

#### extra\_tent

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

#### created\_at

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

## updated\_at

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

```
MAIN_TYPE_MAP = {'camping_car': 'camping_car', 'car_tent': 'tent', 'caravan':
'caravan', 'fourgon': 'caravan', 'tent': 'tent', 'van': 'caravan'}
```

## created\_at\_display()

Format creation date for admin display.

#### updated\_at\_display()

Format updated date for admin display.

## get\_season()

Determine season based on start\_date.

#### calculate\_total\_price(supplement=None)

Calculate total price including extras and supplements. Ensures nights >= 1 and applies correct base price depending on booking\_type and electricity.

#### calculate\_deposit()

Calculate 15% deposit of total price, rounded to 2 decimals.

## save(\*args, \*\*kwargs)

Automatically sets booking\_type from booking\_subtype, included\_people, and assigns a SupplementPrice if missing.

## check\_capacity() Checks availability for given dates and booking type. Raises ValidationError if capacity exceeded. clean() Validates business rules: - Capacity availability - Camping\_car pricing rules Raises ValidationError on violation. exception DoesNotExist Bases: ObjectDoesNotExist exception MultipleObjectsReturned Bases: MultipleObjectsReturned get\_booking\_subtype\_display(\*, field=<django.db.models.fields.CharField: booking\_subtype>) get\_booking\_type\_display(\*, field=<django.db.models.fields.CharField: booking\_type>) get\_electricity\_display(\*, field=<django.db.models.fields.CharField: electricity>) get\_next\_by\_created\_at(\*, field=<django.db.models.fields.DateTimeField: created\_at>, is\_next=True, \*\*kwargs) 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\_next\_by\_updated\_at(\*, field=<django.db.models.fields.DateTimeField: updated\_at>, is\_next=True, \*\*kwargs) get\_previous\_by\_created\_at(\*, field=<django.db.models.fields.DateTimeField: created\_at>, *is\_next=False*, \*\*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) get\_previous\_by\_updated\_at(\*, field=<django.db.models.fields.DateTimeField: updated at>, 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.

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

```
class bookings.models.MobileHome(*args, **kwargs)
```

Bases: Model

Stores mobile home info and translations.

#### Fields:

- name, description\_text: French version
- name\_en/es/de/nl, description\_en/es/de/nl: translated versions
- night\_price, week\_low/mid/high: nightly and weekly prices

- is\_worker\_home: reserved for workers
- worker\_price\_1p/2p/3p: weekly prices for workers

#### - save()

automatically translates name and description via DeepL API

#### **Security:**

- · Only numeric/text fields
- DeepL translations handled server-side
- No user input directly sent to API

#### name

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

#### name\_en

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

#### name\_es

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

#### name de

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

## name\_nl

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

#### description\_text

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

## description\_en

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

#### description\_es

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

## description\_de

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

#### description\_nl

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

## night\_price

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

## night\_price\_mid

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

#### week low

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

#### week\_mid

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

## week\_high

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

#### is\_worker\_home

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

#### worker\_price\_1p

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

## worker\_price\_2p

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

#### worker\_price\_3p

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

#### save(\*args, \*\*kwargs)

Automatically translate name and description to multiple languages using DeepL. Only executed if DEEPL\_API\_KEY is set and description\_text is provided.

#### exception DoesNotExist

Bases: ObjectDoesNotExist

## ${\tt exception MultipleObjectsReturned}$

Bases: MultipleObjectsReturned

#### id

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 bookings.models.SupplementMobileHome(\*args, \*\*kwargs)

Bases: TranslatableModel

Stores extra charges for mobile homes.

#### Fields:

- mobile\_home\_deposit: security deposit
- cleaning\_deposit: cleaning deposit
- bed\_linen\_rental: optional linen rental

## **Security:**

- · Numeric fields only
- Used for internal calculations, no user input processed

#### translations

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.

## exception DoesNotExist

Bases: ObjectDoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

#### bed\_linen\_rental

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## cleaning\_deposit

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

id

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

#### mobile\_home\_deposit

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## class bookings.models.SeasonInfo(\*args, \*\*kwargs)

Bases: TranslatableModel

Stores season start and end dates.

## Fields:

- low\_season\_start/end
- mid\_season\_start\_1/end\_1, mid\_season\_start\_2/end\_2
- high\_season\_start/end

#### **Security:**

- · Date fields only
- Used internally for price and availability calculations

## translations

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.

#### exception DoesNotExist

Bases: ObjectDoesNotExist

## exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

## high\_season\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## high\_season\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### id

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

#### low\_season\_end

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## low\_season\_start

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## mid\_season\_end\_1

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### mid\_season\_end\_2

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

#### mid\_season\_start\_1

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

## mid\_season\_start\_2

Descriptor for translated attributes.

This attribute proxies all get/set calls to the translated model.

Bases: TranslatedFieldsModel

#### exception DoesNotExist

Bases: TranslationDoesNotExist, DoesNotExist, DoesNotExist

#### exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

#### current\_year

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

get\_language\_code\_display(\*, field=<parler.utils.compat.HideChoicesCharField: language\_code>)

#### id

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

#### language\_code

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

#### master

The mandatory Foreign key field to the shared model.

#### master\_id

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

## price\_tourist\_tax

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

#### tourist\_tax\_date

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

class bookings.models.SeasonInfoTranslation(id, language\_code, low\_season\_start, low\_season\_end,

mid\_season\_start\_1, mid\_season\_end\_1, mid\_season\_start\_2, mid\_season\_end\_2, high\_season\_start, high\_season\_end, master)

Bases: TranslatedFieldsModel

## exception DoesNotExist

Bases: TranslationDoesNotExist, DoesNotExist, DoesNotExist

#### exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

```
get_language_code_display(*, field=<parler.utils.compat.HideChoicesCharField: language_code>)
get_next_by_high_season_end(*, field=<django.db.models.fields.DateField: high_season_end>,
                                 is next=True, **kwargs)
get_next_by_high_season_start(*, field=<django.db.models.fields.DateField: high season start>,
                                   is next=True, **kwargs)
get_next_by_low_season_end(*, field=<django.db.models.fields.DateField: low_season_end>,
                               is next=True, **kwargs)
get_next_by_low_season_start(*, field=<django.db.models.fields.DateField: low_season_start>,
                                  is_next=True, **kwargs)
get_next_by_mid_season_end_1(*, field=<django.db.models.fields.DateField: mid_season_end_1>,
                                  is_next=True, **kwargs)
get_next_by_mid_season_end_2(*, field=<django.db.models.fields.DateField: mid_season_end_2>,
                                  is_next=True, **kwargs)
get_next_by_mid_season_start_1(*, field=<django.db.models.fields.DateField: mid_season_start_1>,
                                    is_next=True, **kwargs)
get_next_by_mid_season_start_2(*, field=<django.db.models.fields.DateField: mid_season_start_2>,
                                    is next=True, **kwargs)
get_previous_by_high_season_end(*, field=<django.db.models.fields.DateField: high season end>,
                                     is_next=False, **kwargs)
get_previous_by_high_season_start(*, field=<django.db.models.fields.DateField:</pre>
                                       high_season_start>, is_next=False, **kwargs)
get_previous_by_low_season_end(*, field=<django.db.models.fields.DateField: low_season_end>,
                                    is_next=False, **kwargs)
get_previous_by_low_season_start(*, field=<django.db.models.fields.DateField: low_season_start>,
                                      is next=False, **kwargs)
get_previous_by_mid_season_end_1(*, field=<django.db.models.fields.DateField: mid_season_end_1>,
                                      is_next=False, **kwargs)
get_previous_by_mid_season_end_2(*, field=<django.db.models.fields.DateField: mid_season_end_2>,
                                      is_next=False, **kwargs)
get_previous_by_mid_season_start_1(*, field=<django.db.models.fields.DateField:</pre>
                                         mid_season_start_1>, is_next=False, **kwargs)
get_previous_by_mid_season_start_2(*, field=<django.db.models.fields.DateField:</pre>
                                         mid_season_start_2>, is_next=False, **kwargs)
```

## high\_season\_end

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

#### high\_season\_start

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

#### id

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

## language\_code

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

#### low\_season\_end

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

#### low\_season\_start

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

#### master

The mandatory Foreign key field to the shared model.

#### master\_id

#### mid\_season\_end\_1

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

#### mid\_season\_end\_2

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

## mid\_season\_start\_1

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

#### mid\_season\_start\_2

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>

## Bases: TranslatedFieldsModel

#### exception DoesNotExist

Bases: TranslationDoesNotExist, DoesNotExist, DoesNotExist

#### exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

## bed\_linen\_rental

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

#### cleaning\_deposit

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

## get\_language\_code\_display(\*, field=<parler.utils.compat.HideChoicesCharField: language\_code>)

#### id

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

#### language\_code

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

#### master

The mandatory Foreign key field to the shared model.

#### master\_id

#### mobile\_home\_deposit

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>

• Vues

# bookings.views.booking\_form(request)

Display and process the initial booking form where the user selects: - Booking subtype (tent, caravan, etc.) - Start and end dates - Initial booking preferences (electricity, number of people...)

Security: - Use Django forms for input validation. - Convert decimals and dates to safe formats before saving in session. - Validate capacity with Booking.check\_capacity to prevent overbooking.

# bookings.views.booking\_summary(request)

Display a booking summary before payment. Validates session data and shows: - Total price - Deposit amount - Remaining balance

#### bookings.views.booking\_details(request)

Collect customer's personal details. Once valid, create a Stripe Checkout session for deposit payment.

Security: - Never trust session directly, always validate with Django form. - Stripe key is read from Django settings.

# bookings.views.booking\_confirm(request)

Final step: - Verify data integrity - Save booking in DB - Send confirmation emails (admin + customer) - Clear session

Security: - Validate required fields before saving. - Remove all booking data from session after confirmation.

• Formulaires

# 

Bases: ModelForm Client booking form.

# Main fields:

• booking\_type: campsite type (tent, caravan, camper, etc.)

- start\_date / end\_date: arrival and departure dates
- adults, children\_over\_8, children\_under\_8, pets: number of people and pets
- electricity: whether electricity is required
- cable\_length: required if electricity is selected
- tent\_width / tent\_length, vehicle\_length: specific dimensions depending on type

#### **Security:**

- Dates cannot be in the past.
- Conditional fields are validated based on campsite type.
- All data is cleaned via clean() method.

```
BOOKING_TYPE_CHOICES_FOR_FORM = [('tent', 'Tente'), ('car_tent', 'Voiture Tente'),
('caravan', 'Caravane'), ('fourgon', 'Fourgon'), ('van', 'Van'), ('camping_car',
'Camping-car')]
SUBTYPE_TO_MAIN_TYPE = {'camping_car': 'camping_car', 'car_tent': 'tent',
'caravan': 'caravan', 'fourgon': 'caravan', 'tent': 'tent', 'van': 'caravan'}
ELECTRICITY_CHOICES = [('yes', 'Avec électricité'), ('no', 'Sans électricité')]
class Meta
    Bases: object
    model
       alias of Booking
    fields = ['booking_type', 'vehicle_length', 'tent_width', 'tent_length',
    'adults', 'children_over_8', 'children_under_8', 'pets', 'electricity',
    'cable_length', 'start_date', 'end_date']
    labels = {'cable_length': 'Longueur du câble électrique (m)',
    'children_over_8': 'Enfants +8 ans', 'children_under_8': 'Enfants -8 ans',
    'pets': 'Animaux', 'tent_length': 'Longueur de la tente (m)', 'tent_width':
    'Largeur de la tente (m)', 'vehicle_length': 'Longueur du véhicule (m)'}
    widgets = {'cable_length': <django.forms.widgets.NumberInput object>,
    'children_over_8': <django.forms.widgets.Select object>, 'children_under_8':
    <django.forms.widgets.Select object>, 'pets': <django.forms.widgets.Select</pre>
    object>, 'tent_length': <diango.forms.widgets.NumberInput object>,
    'tent_width': <django.forms.widgets.NumberInput object>, 'vehicle_length':
    <django.forms.widgets.NumberInput object>}
clean()
```

#### **Custom validation:**

- Arrival date cannot be in the past.
- Departure date must be after arrival.
- Conditional fields are required depending on campsite type and electricity.

```
base_fields = {'adults': <diango.forms.fields.IntegerField object>, 'booking_type':
<django.forms.fields.ChoiceField object>, 'cable_length':
<django.forms.fields.DecimalField object>, 'children_over_8':
<django.forms.fields.IntegerField object>, 'children_under_8':
<django.forms.fields.IntegerField object>, 'electricity':
<django.forms.fields.ChoiceField object>, 'end_date':
<django.forms.fields.DateField object>, 'pets': <django.forms.fields.IntegerField</pre>
object>, 'start_date': <django.forms.fields.DateField object>, 'tent_length':
<django.forms.fields.DecimalField object>, 'tent_width':
<django.forms.fields.DecimalField object>, 'vehicle_length':
<django.forms.fields.DecimalField object>}
declared_fields = {'adults': <django.forms.fields.IntegerField object>,
'booking_type': <django.forms.fields.ChoiceField object>, 'electricity':
<django.forms.fields.ChoiceField object>, 'end_date':
<django.forms.fields.DateField object>, 'start_date':
<django.forms.fields.DateField object>}
```

#### property media

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

```
class bookings.forms.BookingDetailsForm(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, field_order=None,
                                                use_required_attribute=None, renderer=None,
                                                bound field class=None)
```

Bases: Form

Client personal details form.

### **Security:**

- Email field validated automatically.
- Maximum length for all text fields to prevent injection.
- · All data cleaned via cleaned data.

## clean()

Centralized cleaning and validation: - Strips leading/trailing spaces for all text fields. - Normalizes email to lowercase. - Validates phone number contains only digits, spaces, +, -, ().

```
base_fields = {'address': <django.forms.fields.CharField object>, 'city':
<django.forms.fields.CharField object>, 'email': <django.forms.fields.EmailField</pre>
object>, 'first_name': <django.forms.fields.CharField object>, 'last_name':
<django.forms.fields.CharField object>, 'phone': <django.forms.fields.CharField</pre>
object>, 'postal_code': <django.forms.fields.CharField object>}
declared_fields = {'address': <django.forms.fields.CharField object>, 'city':
<django.forms.fields.CharField object>, 'email': <django.forms.fields.EmailField</pre>
object>, 'first_name': <django.forms.fields.CharField object>, 'last_name':
<django.forms.fields.CharField object>, 'phone': <django.forms.fields.CharField</pre>
object>, 'postal_code': <django.forms.fields.CharField object>}
```

# property media

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

```
• Admin
class bookings.admin.CapacityAdmin(model, admin site)
     Bases: ModelAdmin
     Admin for the Capacity model: displays booking types and maximum capacity.
     list_display = ('booking_type', 'max_places', 'number_locations',
     'number_mobile_homes')
     list_filter = ('booking_type',)
     search_fields = ('booking_type',)
     fieldsets = (("Nombre d'emplacements par type", {'description': "Nombre maximum
     d'emplacements disponibles pour chaque type de réservation.", 'fields':
     ('booking_type', 'max_places')}), ('Capacités totales du camping', {'fields':
     ('number_locations', 'number_mobile_homes')}))
     property media
class bookings.admin.SupplementPriceAdmin(model, admin_site)
     Bases: ModelAdmin
     Admin for the SupplementPrice model: display and manage extra pricing.
     list_display = ('extra_adult_price', 'child_over_8_price', 'child_under_8_price',
     'pet_price', 'extra_vehicle_price', 'extra_tent_price',
     'visitor_price_without_swimming_pool', 'visitor_price_with_swimming_pool')
     search_fields = ()
     fieldsets = (('Suppléments', {'description': "Tarifs des suppléments quelque soit
     la saison et le type d'hébergement.", 'fields': ('extra_adult_price',
     'child_over_8_price', 'child_under_8_price', 'pet_price', 'extra_vehicle_price',
     'extra_tent_price', 'visitor_price_without_swimming_pool',
     'visitor_price_with_swimming_pool')}),)
     property media
class bookings.admin.PriceAdminForm(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
     Custom form for PriceAdmin to add validation logic.
     class Meta
         Bases: object
         model
             alias of Price
         fields = '__all__'
```

#### clean()

Hook for doing any extra form-wide cleaning after Field.clean() has been called on every field. Any ValidationError raised by this method will not be associated with a particular field; it will have a special-case association with the field named 'all'.

```
base_fields = {'booking_type': <django.forms.fields.TypedChoiceField object>,
'is_worker': <django.forms.fields.BooleanField object>,
'price_1_person_with_electricity': <django.forms.fields.DecimalField object>,
'price_1_person_without_electricity': <django.forms.fields.DecimalField object>,
'price_2_persons_with_electricity': <django.forms.fields.DecimalField object>,
'price_2_persons_without_electricity': <django.forms.fields.DecimalField object>,
'season': <django.forms.fields.TypedChoiceField object>, 'supplements':
<django.forms.models.ModelChoiceField object>, 'weekend_price_without_electricity':
<django.forms.fields.DecimalField object>, 'weekend_price_without_electricity':
<django.forms.fields.DecimalField object>, 'worker_week_price':
<django.forms.fields.DecimalField object>}

declared_fields = {}

property media
```

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

class bookings.admin.PriceAdmin(model, admin\_site)

Bases: ModelAdmin

Admin interface for Price model.

- Uses custom form PriceAdminForm to validate fields
- Displays pricing info for different booking types, seasons, and workers
- · Organizes form fields into sections with descriptions
- List view shows both 1-person and 2-person prices, weekend and worker prices

#### form

alias of PriceAdminForm

```
list_display = ('booking_type', 'season', 'is_worker',
'price_1_person_with_electricity', 'price_2_persons_with_electricity',
'price_1_person_without_electricity', 'price_2_persons_without_electricity',
'worker_week_price', 'weekend_price_without_electricity',
'weekend_price_with_electricity')

list_filter = ('booking_type', 'season', 'is_worker')

search_fields = ('booking_type', 'season')

ordering = ('booking_type', 'season', 'is_worker')

exclude = ('included_people',)
```

```
fieldsets = (('Tarifs classiques', {'description': "Pour les tentes, caravanes et
     fourgons, saisir les tarifs 1 et 2 personnes.<br/>
br>Pour les camping-cars, le tarif est
    identique pour 1 ou 2 personnes.Merci de renseigner uniquement la ligne <strong>2
    personnes</strong> et laisser l'autre vide.", 'fields': ('booking_type', 'season',
     'price_1_person_with_electricity', 'price_2_persons_with_electricity',
     'price_1_person_without_electricity', 'price_2_persons_without_electricity')}),
     ('Tarifs ouvriers semaine', {'description': 'Prix spéciaux pour les ouvriers,
    électricité incluse.', 'fields': ('is_worker', 'worker_week_price')}), ('Tarifs
    ouvriers weekend', {'description': "Tarifs réduits le week-end quelque soit le type
    d'hébergement.", 'fields': ('weekend_price_without_electricity',
     'weekend_price_with_electricity')}))
    property media
class bookings.admin.OtherPriceAdmin(model, admin_site)
    Bases: TranslatableAdmin
    Admin for the OtherPrice model: manage current year and tourist tax.
    list_display = ('current_year', 'tourist_tax_date', 'price_tourist_tax')
    fieldsets = (('Année en cours', {'fields': ('current_year',)}), ('Taxe de séjour',
    {'fields': ('tourist_tax_date', 'price_tourist_tax')}))
    property media
class bookings.admin.BookingAdmin(model, admin_site)
    Bases: ModelAdmin
    Admin interface for Booking model.
       • Displays client info, booking details, capacities, and extra options
       • Allows editing of deposit status directly from list view
       • Provides filters by type, electricity, deposit, and dates
       • Readonly fields: created at display, updated at display
    list_display = ('last_name', 'first_name', 'start_date', 'end_date', 'booking_type',
     'booking_subtype', 'electricity', 'deposit_paid', 'created_at_display',
     'updated_at_display')
    list_editable = ('deposit_paid',)
    list_filter = ('booking_type', 'booking_subtype', 'electricity', 'deposit_paid',
     'start_date', 'end_date')
    search_fields = ('last_name', 'first_name', 'email', 'phone')
    ordering = ('-created_at',)
    readonly_fields = ('created_at_display', 'updated_at_display')
```

```
fieldsets = (('Informations client', {'fields': ('last_name', 'first_name',
     'address', 'postal_code', 'city', 'phone', 'email')}), ('Détails de la réservation',
     {'fields': ('start_date', 'end_date', 'booking_type', 'booking_subtype',
     'electricity', 'deposit_paid')}), ('Capacités et options', {'fields': ('adults',
     'children_over_8', 'children_under_8', 'pets', 'extra_vehicle', 'extra_tent')}),
     ('Détails supplémentaires', {'description': "Ces champs apparaissent uniquement
    pour certains types d'hébergements.", 'fields': ('tent_length', 'tent_width',
     'vehicle_length', 'cable_length')}), ('Dates de création et de mise à jour',
    {'fields': ('created_at_display', 'updated_at_display')}))
    property media
class bookings.admin.MobileHomeAdmin(model, admin site)
    Bases: ModelAdmin
    Admin for the MobileHome model: display pricing, information, and options.
    list_display = ('name', 'night_price', 'night_price_mid', 'week_low', 'week_mid',
     'week_high', 'is_worker_home')
    search_fields = ('name',)
    list_filter = ('is_worker_home',)
    fieldsets = (('Informations générales', {'fields': ('name', 'description_text')}),
     ('Prix à la nuitée', {'description': 'Prix à la nuitée uniquement en basse et
    moyenne saison (si le prix moyenne saison diffère, utilisez night_price_mid).',
     'fields': ('night_price', 'night_price_mid')}), ('Prix par semaine', {'fields':
     ('week_low', 'week_mid', 'week_high')}), ('Prix ouvriers', {'description':
     'Uniquement pour le mobil-home ouvrier. Les prix sont fixes selon le nombre de
    personnes.', 'fields': ('is_worker_home', 'worker_price_1p', 'worker_price_2p',
     'worker_price_3p')}))
    property media
class bookings.admin.SupplementMobileHomeAdmin(model, admin_site)
    Bases: TranslatableAdmin
    Admin for the SupplementMobileHome model: manage deposits and linen rental.
    list_display = ('mobile_home_deposit', 'cleaning_deposit', 'bed_linen_rental')
    fieldsets = (('Cautions et location de linge', {'fields': ('mobile_home_deposit',
     'cleaning_deposit', 'bed_linen_rental')}),)
    property media
class bookings.admin.SeasonInfoAdmin(model, admin site)
    Bases: TranslatableAdmin
    Admin for the SeasonInfo model: manage low, mid, and high season dates.
    list_display = ['low_season_start', 'low_season_end', 'mid_season_start_1',
     'mid_season_end_1', 'mid_season_start_2', 'mid_season_end_2', 'high_season_start',
     'high_season_end']
```

```
fieldsets = (('Saisons', {'fields': ('low_season_start', 'low_season_end',
     'mid_season_start_1', 'mid_season_end_1', 'mid_season_start_2', 'mid_season_end_2',
     'high_season_start', 'high_season_end')}),)
     property media
   • Configuration App
class bookings.apps.BookingsConfig(app name, app module)
     Bases: AppConfig
     Configuration for the 'bookings' app, which manages pricing and reservations.
     default_auto_field = 'django.db.models.BigAutoField'
     name = 'bookings'
     verbose_name = 'Prix et Réservations'
1.6 Tests Bookings
   • Modèles
bookings.tests.test_models.test_supplementprice_creation()
     Test creation of SupplementPrice object and its string representation.
bookings.tests.test_models.test_price_save_and_clean()
     Test saving a Price object, automatic field population, and validation rules.
bookings.tests.test_models.test_booking_save_total_and_deposit()
     Test booking save, total price calculation, and deposit computation.
bookings.tests.test_models.test_booking_capacity_validation()
     Test that capacity validation prevents overbooking.
bookings.tests.test_models.test_capacity_str()
     Test the string representation of Capacity.
bookings.tests.test_models.test_mobilehome_creation()
     Test MobileHome object creation and string representation.
bookings.tests.test_models.test_supplementmobilehome_creation()
     Test SupplementMobileHome object creation and string representation.
bookings.tests.test_models.test_seasoninfo_creation()
     Test SeasonInfo object creation and translation handling (French).
   Vues
bookings.tests.test_views.valid_booking_data()
     Returns a dictionary of valid booking data for form submission tests.
bookings.tests.test_views.client_details_data()
     Returns a dictionary of valid client personal details for form submission tests.
bookings.tests.test_views.supplements()
     Creates and returns a SupplementPrice object for price calculation tests.
```

```
bookings.tests.test_views.set_booking_session(client, booking_data)
     Stores booking data in client session for views that rely on session data.
bookings.tests.test_views.test_booking_form_valid(mock check capacity, client)
     Submitting a valid booking form should redirect to booking_summary and call check_capacity.
bookings.tests.test_views.test_booking_summary_displays_correct_prices(mock deposit,
                                                                                   mock total, client,
                                                                                   valid booking data)
     Booking summary view should correctly display total, deposit, and remaining balance.
bookings.tests.test_views.test_booking_details_creates_stripe_session(mock_deposit,
                                                                                  mock_stripe, client,
                                                                                  valid_booking_data,
                                                                                  client details data,
                                                                                  supplements)
     Booking details view should create a Stripe session for deposit payment and redirect to Stripe.
bookings.tests.test_views.test_booking_confirm_saves_booking_and_sends_emails(mock_send,
                                                                                           client.
                                                                                           valid_booking_data,
                                                                                           client_details_data)
     Booking confirmation should save the booking, mark deposit as paid, send emails, and clear session.
   · Formulaires
class bookings.tests.test_forms.TestBookingFormClassic
     Bases: object
     test_valid_data_tent(mock_check_capacity)
          Valid data for tent type with electricity should pass
     test_missing_required_field(mock_check_capacity)
          Missing conditional field (cable_length) should raise error
     test_past_start_date(mock_check_capacity)
          Start date in the past should raise error
     test_invalid_type_field(mock check capacity)
          Invalid booking type should raise error
class bookings.tests.test_forms.TestBookingDetailsForm
     Bases: object
     test_valid_data()
          All valid fields should pass and normalize email/strip spaces
     test_invalid_phone()
          Phone with invalid characters should raise error
     test_missing_required_fields()
          Required fields missing should raise errors
     pytestmark = [Mark(name='django_db', args=(), kwargs={})]
```

# **PYTHON MODULE INDEX**

# b bookings.admin, 36 bookings.apps, 40 bookings.forms, 33 bookings.models, 16 bookings.tests.test\_forms, 41 ${\tt bookings.tests.test\_models}, 40$ bookings.tests.test\_views, 40 bookings.views, 33 С core.apps, 12 core.models, 3 core.sitemaps, 12 core.templatetags, 12 core.tests.test\_models, 13 core.tests.test\_views, 13 core.views, 11 r reservations.apps, 15 reservations.forms, 14 reservations.tests.test\_forms, 16 reservations.tests.test\_views, 16 reservations.views, 14

44 Python Module Index

# **INDEX**

A	bed_linen_rental (book-
<pre>about_view() (in module core.views), 11</pre>	ings.models.SupplementMobileHome at-
ACCOMMODATION_CHOICES (reserva-	tribute), 28
$tions. forms. Reservation Request Form\ attribute),$	bed_linen_rental (book-
14	ings.models.SupplementMobileHomeTranslation
<pre>accommodations_view() (in module core.views), 11</pre>	attribute), 32
<pre>activities_view() (in module core.views), 11</pre>	Booking (class in bookings.models), 21
address (bookings.models.Booking attribute), 22	Booking.DoesNotExist, 25
adults (bookings.models.Booking attribute), 24	Booking. MultipleObjectsReturned, 25
alternates()(core.sitemaps.MultilingualStaticSitemap	booking_confirm() (in module bookings.views), 33
method), 13	<pre>booking_details() (in module bookings.views), 33 booking_form() (in module bookings.views), 33</pre>
arrivals_end_high (core.models.CampingInfo at-	booking_subtype (bookings.models.Booking attribute),
tribute), 3	22
arrivals_end_high(core.models.CampingInfoTranslation	on booking_summary() (in module bookings.views), 33
attribute), 7	booking_type (bookings.models.Booking attribute), 23
arrivals_end_low (core.models.CampingInfo at-	booking_type (bookings.models.Capacity attribute), 21
tribute), 3	
arrivals_end_low(core.models.CampingInfoTranslation	BOOKING_TYPE_CHOICES_FOR_FORM (book-
attribute), 7	ings.forms.BookingFormClassic attribute),
arrivals_start_high (core.models.CampingInfo at-	34
<pre>tribute), 3 arrivals_start_high</pre>	BookingAdmin (class in bookings.admin), 38
(core.models.CampingInfoTranslation at-	BookingDetailsForm (class in bookings.forms), 35
tribute), 7	BookingFormClassic (class in bookings.forms), 33
inouc,	BookingFormClassic.Meta (class in bookings.forms),
В	34
bar_hours_end (core.models.FoodInfo attribute), 6	bookings.admin
bar_hours_end (core.models.FoodInfoTranslation at-	module, 36
tribute), 8	bookings.apps
bar_hours_start (core.models.FoodInfo attribute), 6	module, 40
bar_hours_start (core.models.FoodInfoTranslation at-	bookings.forms
tribute), 9	module, 33
base_fields (bookings.admin.PriceAdminForm at-	bookings.models
tribute), 37	module, 16
base_fields (bookings.forms.BookingDetailsForm at-	bookings.tests.test_forms
tribute), 35	module, 41
base_fields (bookings.forms.BookingFormClassic at-	bookings.tests.test_models
tribute), 34	module, 40
base_fields (reserva-	bookings.tests.test_views
tions.forms.ReservationRequestForm attribute),	module, 40
15	bookings.views
	module, 33

BookingsConfig (class in bookings.apps), 40 bread_hours_end (core.models.FoodInfo attribute), 6 bread_hours_end (core.models.FoodInfoTranslation at- tribute), 9	<pre>city (bookings.models.Booking attribute), 23 clean() (bookings.admin.PriceAdminForm method), 36 clean() (bookings.forms.BookingDetailsForm method),</pre>
bread_hours_reservations (core.models.FoodInfo	clean() (bookings.forms.BookingFormClassic method),
attribute), 6	34 class() (backings models Rooking method) 25
bread_hours_reservations (core.models.FoodInfoTranslation attribute), 9	clean() (bookings.models.Booking method), 25 clean() (bookings.models.Price method), 19
bread_hours_start (core.models.FoodInfo attribute), 6	clean() (reservations.forms.ReservationRequestForm
bread_hours_start (core.models.FoodInfoTranslation	method), 14
attribute), 9	cleaning_deposit (book-
burger_food_days (core.models.FoodInfo attribute), 6	ings.models.SupplementMobileHome at-
${\tt burger\_food\_days}  (core.models.FoodInfoTranslation$	tribute), 28
attribute), 9	cleaning_deposit (book-
burger_food_hours_end (core.models.FoodInfo attribute), 6	ings.models.SupplementMobileHomeTranslation attribute), 32
burger_food_hours_end	<pre>client_details_data() (in module book-</pre>
(core.models.FoodInfoTranslation attribute), 9	ings.tests.test_views), 40
<pre>burger_food_hours_start (core.models.FoodInfo at-</pre>	core.apps
tribute), 6	module, 12
burger_food_hours_start	core.models
$(core.models. Food Info Translation\ attribute), 9$	module, 3
	core.sitemaps
C	module, 12
cable_length (bookings.models.Booking attribute), 23	core.templatetags
<pre>calculate_deposit() (bookings.models.Booking</pre>	module, 12
method), 24	core.tests.test_models
<pre>calculate_total_price() (bookings.models.Booking</pre>	module, 13
method), 24	core.tests.test_views
CampingInfo (class in core.models), 3	module, 13
<pre>CampingInfo.DoesNotExist, 3</pre>	core.views
CampingInfo.MultipleObjectsReturned, 3	module, 11
CampingInfoTranslation (class in core.models), 7	CoreConfig (class in core.apps), 12
CampingInfoTranslation.DoesNotExist, 7	created_at (bookings.models.Booking attribute), 24
CampingInfoTranslation.MultipleObjectsReturne	method), 24
Capacity (class in bookings.models), 21	<pre>current_year (bookings.models.OtherPrice attribute),</pre>
Capacity.DoesNotExist,21	20
Capacity.MultipleObjectsReturned, 21	current_year (bookings.models.OtherPriceTranslation
CapacityAdmin (class in bookings.admin), 36	attribute), 30
changefreq (core.sitemaps.MultilingualStaticSitemap attribute), 12	D
<pre>check_capacity() (bookings.models.Booking method),</pre>	<pre>declared_fields (bookings.admin.PriceAdminForm</pre>
child_over_8_price (book-	declared_fields (book-
ings.models.SupplementPrice attribute), 17	ings.forms.BookingDetailsForm attribute), 35
child_under_8_price (book-	declared_fields (book-
ings.models.SupplementPrice attribute), 17	ings.forms.BookingFormClassic attribute), 35
children_over_8 (bookings.models.Booking attribute),	declared_fields (reserva-
24	tions.forms.ReservationRequestForm attribute),
children_under_8 (bookings.models.Booking at- tribute). 24	15

<pre>default_auto_field (bookings.apps.BookingsConfig</pre>	<pre>fieldsets (bookings.admin.BookingAdmin attribute), 38</pre>
<pre>default_auto_field(core.apps.CoreConfig attribute),</pre>	fieldsets (bookings.admin.CapacityAdmin attribute), 36
default_auto_field (reserva-	fieldsets (bookings.admin.MobileHomeAdmin at-
tions.apps.ReservationsConfig attribute),	tribute), 39
15	fieldsets (bookings.admin.OtherPriceAdmin at-
departure_end (core.models.CampingInfo attribute), 4	tribute), 38
<pre>departure_end (core.models.CampingInfoTranslation</pre>	fieldsets (bookings.admin.PriceAdmin attribute), 37 fieldsets (bookings.admin.SeasonInfoAdmin at-
deposit_paid (bookings.models.Booking attribute), 23	tribute), 39
description_de (bookings.models.MobileHome attribute), 26	fieldsets (bookings.admin.SupplementMobileHomeAdmin attribute), 39
description_en (bookings.models.MobileHome attribute), 26	fieldsets (bookings.admin.SupplementPriceAdmin attribute), 36
description_es (bookings.models.MobileHome attribute), 26	first_name (bookings.models.Booking attribute), 22 FoodInfo (class in core.models), 5
description_nl (bookings.models.MobileHome attribute), 26	FoodInfo.DoesNotExist, 5 FoodInfo.MultipleObjectsReturned, 6
description_text (bookings.models.MobileHome attribute), 26	FoodInfoTranslation (class in core.models), 8 FoodInfoTranslation.DoesNotExist, 8
dryer_price (core.models.LaundryInfo attribute), 7	FoodInfoTranslation.MultipleObjectsReturned,
dryer_price (core.models.LaundryInfoTranslation attribute), 10	8 form (bookings.admin.PriceAdmin attribute), 37
	Total (bookings.ttmin.i recritima timebine), 37
E	G
electricity (bookings.models.Booking attribute), 23	<pre>get_booking_subtype_display()</pre>
ELECTRICITY_CHOICES (book-	ings.models.Booking method), 25
ings.forms.BookingFormClassic attribute), 34	<pre>get_booking_type_display()</pre>
ELECTRICITY_CHOICES (bookings.models.Booking at-	<pre>get_booking_type_display()</pre>
tribute), 22	ings.models.Capacity method), 21
ELECTRICITY_CHOICES (reserva-	<pre>get_booking_type_display()</pre>
tions.forms.ReservationRequestForm attribute),	ings.models.Price method), 20
14	get_electricity_display() (book-
email (bookings.models.Booking attribute), 23	ings.models.Booking method), 25
end_date (bookings.models.Booking attribute), 23	get_language_code_display() (book-
exclude (bookings.admin.PriceAdmin attribute), 37 extra_adult_price (book-	ings.models.OtherPriceTranslation method), 30
ings.models.SupplementPrice attribute),	get_language_code_display() (book-
17	ings. models. Seas on Info Translation  method),
extra_tent (bookings.models.Booking attribute), 24	30
extra_tent_price (bookings.models.SupplementPrice attribute), 17	get_language_code_display() (book-
extra_vehicle (bookings.models.Booking attribute), 24	ings.models.SupplementMobileHomeTranslation method), 32
extra_vehicle_price (book-	<pre>get_language_code_display()</pre>
ings.models.SupplementPrice attribute), 17	(core.models.CampingInfoTranslation method), 8
F	<pre>get_language_code_display()</pre>
	(core.models.FoodInfoTranslation method),
fields (bookings.admin.PriceAdminForm.Meta at-	9
tribute), 36	<pre>get_language_code_display()           (core.models.LaundryInfoTranslation method),</pre>
fields (bookings.forms.BookingFormClassic.Meta attribute), 34	(core.models.Launaryingo1ransiation method), 10

<pre>get_language_code_display()</pre>		<pre>get_previous_by_mid_season_end_1() (book-</pre>
(core.models.SwimmingPoolInfoTran method), 10	slation	ings.models.SeasonInfoTranslation method), 31
<pre>get_next_by_created_at()</pre>	(book-	<pre>get_previous_by_mid_season_end_2() (book-</pre>
ings.models.Booking method), 25		ings.models.SeasonInfoTranslation method),
<pre>get_next_by_end_date() (bookings.mode</pre>	els.Booking	31 get_previous_by_mid_season_start_1() (book-
<pre>get_next_by_high_season_end()</pre>	(book-	ings.models.SeasonInfoTranslation method),
ings.models.SeasonInfoTranslation	method),	31
31		<pre>get_previous_by_mid_season_start_2() (book-</pre>
<pre>get_next_by_high_season_start()</pre>	(book-	ings.models.SeasonInfoTranslation method),
ings.models.SeasonInfoTranslation	method),	31
31		<pre>get_previous_by_start_date()</pre>
<pre>get_next_by_low_season_end()</pre>	(book-	ings.models.Booking method), 25
ings.models.SeasonInfoTranslation	method),	<pre>get_previous_by_tourist_tax_date() (book-</pre>
31		ings.models.OtherPriceTranslation method),
<pre>get_next_by_low_season_start()</pre>	(book-	30
ings.models.SeasonInfoTranslation	method),	<pre>get_previous_by_updated_at() (book-</pre>
31	,	ings.models.Booking method), 25
<pre>get_next_by_mid_season_end_1()</pre>	(book-	<pre>get_season() (bookings.models.Booking method), 24</pre>
ings.models.SeasonInfoTranslation	method),	<pre>get_season_display()</pre>
31	,,	method), 20
<pre>get_next_by_mid_season_end_2()</pre>	(book-	<i>''</i>
ings.models.SeasonInfoTranslation	method),	Н
31	,,	high_season_end (bookings.models.SeasonInfo at-
<pre>get_next_by_mid_season_start_1()</pre>	(book-	tribute), 29
ings.models.SeasonInfoTranslation	method),	high_season_end (book-
31	,,	ings.models.SeasonInfoTranslation attribute),
<pre>get_next_by_mid_season_start_2()</pre>	(book-	31
ings.models.SeasonInfoTranslation	method),	high_season_start (bookings.models.SeasonInfo at-
31	,,	tribute), 29
<pre>get_next_by_start_date()</pre>	(book-	high_season_start (book-
ings.models.Booking method), 25	(	ings.models.SeasonInfoTranslation attribute),
<pre>get_next_by_tourist_tax_date()</pre>	(book-	31
ings.models.OtherPriceTranslation	method),	home_view() (in module core.views), 11
30	,,	nome_view() (in modute core.views), 11
<pre>get_next_by_updated_at()</pre>	(book-	
ings.models.Booking method), 25		id (bookings.models.Booking attribute), 25
<pre>get_previous_by_created_at()</pre>	(book-	id (bookings.models.Capacity attribute), 21
ings.models.Booking method), 25		· · · · · · · · · · · · · · · · · · ·
<pre>get_previous_by_end_date()</pre>	(book-	id (bookings.models.MobileHome attribute), 27
ings.models.Booking method), 25	•	id (bookings.models.OtherPrice attribute), 20
<pre>get_previous_by_high_season_end()</pre>	(book-	id (bookings.models.OtherPriceTranslation attribute), 30
ings.models.SeasonInfoTranslation	method),	id (bookings.models.Price attribute), 20
31		id (bookings.models.SeasonInfo attribute), 29 id (bookings.models.SeasonInfoTranslation attribute), 31
<pre>get_previous_by_high_season_start()</pre>	(book-	id (bookings.models.SupplementMobileHome attribute), 31
ings.models.SeasonInfoTranslation	method),	
31	,,	28
<pre>get_previous_by_low_season_end()</pre>	(book-	id (bookings.models.SupplementMobileHomeTranslation
ings.models.SeasonInfoTranslation	method),	attribute), 33
31	, ,	id (bookings.models.SupplementPrice attribute), 17
<pre>get_previous_by_low_season_start()</pre>	(book-	id (core.models.CampingInfo attribute), 4
ings.models.SeasonInfoTranslation	method),	id (core.models.CampingInfoTranslation attribute), 8
31	//	id (core.models.FoodInfo attribute), 6
		<pre>id (core.models.FoodInfoTranslation attribute), 9</pre>

id (core.models.LaundryInfo attribute), 7	list_display (bookings.admin.SeasonInfoAdmin
id (core.models.LaundryInfoTranslation attribute), 10	attribute), 39
id (core.models.SwimmingPoolInfo attribute), 5	${\tt list\_display} \ (bookings. admin. Supplement Mobile Home Admin$
id (core.models.SwimmingPoolInfoTranslation attribute),	attribute), 39
10	list_display (bookings.admin.SupplementPriceAdmin
included_people (bookings.models.Price attribute), 19	attribute), 36
<pre>infos_view() (in module core.views), 11</pre>	list_editable (bookings.admin.BookingAdmin at-
is_worker (bookings.models.Price attribute), 18	tribute), 38
is_worker_home (bookings.models.MobileHome attribute), 27	list_filter (bookings.admin.BookingAdmin attribute), 38
items() (core.sitemaps.MultilingualStaticSitemap method), 13	list_filter (bookings.admin.CapacityAdmin attribute), 36
,	list_filter (bookings.admin.MobileHomeAdmin at-
L	tribute), 39
labels (bookings.forms.BookingFormClassic.Meta at-	list_filter (bookings.admin.PriceAdmin attribute), 37
tribute), 34	location() (core.sitemaps.MultilingualStaticSitemap
language_code (book-	method), 13
ings.models.OtherPriceTranslation attribute),	low_season_end (bookings.models.SeasonInfo at-
30	tribute), 29
language_code (book- ings.models.SeasonInfoTranslation attribute),	low_season_end (book-
•	ings.models.SeasonInfoTranslation attribute),
32	32
language_code (book-	low_season_start (bookings.models.SeasonInfo
ings.models.SupplementMobileHomeTranslation attribute), 33	attribute), 29
language_code (core.models.CampingInfoTranslation	low_season_start (book-
attribute), 8	ings.models.SeasonInfoTranslation attribute),
language_code (core.models.FoodInfoTranslation at- tribute), 9	32
language_code (core.models.LaundryInfoTranslation	M
attribute), 10	MAIN_TYPE_MAP (bookings.models.Booking attribute), 24
language_code (core.models.SwimmingPoolInfoTranslation	master (bookings.models.OtherPriceTranslation at-
attribute), 10	tribute), 30
languages (core.sitemaps.MultilingualStaticSitemap at-	${\tt master}  (bookings.models. Season Info Translation  at-$
tribute), 12	tribute), 32
last_name (bookings.models.Booking attribute), 22	$\verb master  (bookings.models. Supplement Mobile Home Translation $
LaundryInfo (class in core.models), 6	attribute), 33
LaundryInfo.DoesNotExist,7	${\tt master}\ (core.models. Camping Info Translation\ attribute),$
LaundryInfo.MultipleObjectsReturned,7	8
LaundryInfoTranslation (class in core.models), 9	master (core.models.FoodInfoTranslation attribute), 9
LaundryInfoTranslation.DoesNotExist,9	master (core.models.LaundryInfoTranslation attribute),
LaundryInfoTranslation.MultipleObjectsReturne	d, 10
10	master (core.models.SwimmingPoolInfoTranslation at-
legal_view() (in module core.views), 12	tribute), 10
list_display (bookings.admin.BookingAdmin at-	${\tt master\_id}\ (bookings.models.OtherPriceTranslation\ at-$
tribute), 38	tribute), 30
list_display (bookings.admin.CapacityAdmin at- tribute), 36	<pre>master_id (bookings.models.SeasonInfoTranslation at- tribute), 32</pre>
list_display (bookings.admin.MobileHomeAdmin at-	master_id(bookings.models.SupplementMobileHomeTranslation
tribute), 39	attribute), 33
list_display (bookings.admin.OtherPriceAdmin at-	master_id (core.models.CampingInfoTranslation
tribute), 38	attribute), 8
list_display (bookings.admin.PriceAdmin attribute),	master_id (core.models.FoodInfoTranslation attribute),
11st_urspray (bookings.damin.FriceAdmin dirribule),	q

master_id (core.models.LaundryInfoTranslation at- tribute), 10	model (bookings.forms.BookingFormClassic.Meta attribute), 34
master_id (core.models.SwimmingPoolInfoTranslation	module
attribute), 10	bookings.admin, 36
max_places (bookings.models.Capacity attribute), 21	bookings.apps, 40
media (bookings.admin.BookingAdmin property), 39	bookings.forms, 33
media (bookings.admin.CapacityAdmin property), 36	bookings.models, 16
media (bookings.admin.MobileHomeAdmin property), 39	bookings.tests.test_forms, 41
media (bookings.admin.OtherPriceAdmin property), 38	bookings.tests.test_models, 40
media (bookings.admin.PriceAdmin property), 38	bookings.tests.test_views, 40
media (bookings.admin.PriceAdminForm property), 37	bookings.views, 33
media (bookings.admin.SeasonInfoAdmin property), 40	core.apps, 12
media (bookings.admin.SupplementMobileHomeAdmin	core.models, 3
property), 39	core.sitemaps, 12
media (bookings.admin.SupplementPriceAdmin prop-	core.templatetags, 12
erty), 36	core.tests.test_models, 13
media (bookings.forms.BookingDetailsForm property),	core.tests.test_views, 13
35	core.views, 11
media (bookings.forms.BookingFormClassic property),	reservations.apps, 15
35	reservations.forms, 14
media (reservations.forms.ReservationRequestForm	reservations.tests.test_forms, 16
property), 15	reservations.tests.test_views, 16
mid_season_end_1 (bookings.models.SeasonInfo	reservations.views, 14
attribute), 29	MultilingualStaticSitemap (class in core.sitemaps),
mid_season_end_1 (book-	12
ings.models.SeasonInfoTranslation attribute),	N
32 (haakinaa madala Sagaanlufa	
mid_season_end_2 (bookings.models.SeasonInfo	name (bookings.apps.BookingsConfig attribute), 40
attribute), 29	name (bookings.models.MobileHome attribute), 26
mid_season_end_2 (book- ings.models.SeasonInfoTranslation attribute),	name (core.apps.CoreConfig attribute), 12
ings.models.SeasonInfoTranslation attribute), 32	name (reservations.apps.ReservationsConfig attribute),
mid_season_start_1 (bookings.models.SeasonInfo at-	name_de (bookings.models.MobileHome attribute), 26
tribute), 29	name_en (bookings.models.MobileHome attribute), 26
mid_season_start_1 (book-	name_es (bookings.models.MobileHome attribute), 26
ings.models.SeasonInfoTranslation attribute),	name_nl (bookings.models.MobileHome attribute), 26
32	night_price (bookings.models.MobileHome attribute),
mid_season_start_2 (bookings.models.SeasonInfo at-	26
tribute), 29	night_price_mid (bookings.models.MobileHome at-
mid_season_start_2 (book-	<i>tribute</i> ), 26
ings.models.SeasonInfoTranslation attribute),	not_found_view() (in module core.views), 12
32	number_locations (bookings.models.Capacity at-
mobile_home_deposit (book-	tribute), 21
ings.models.SupplementMobileHome at-	number_mobile_homes (bookings.models.Capacity at-
tribute), 28	tribute), 21
mobile_home_deposit (book-	-
ings.models.SupplementMobileHomeTranslation	0
attribute), 33	objects (bookings.models.Booking attribute), 25
MobileHome (class in bookings.models), 25	objects (bookings.models.Capacity attribute), 21
MobileHome.DoesNotExist,27	objects (bookings.models.MobileHome attribute), 27
MobileHome.MultipleObjectsReturned,27	objects (bookings.models.OtherPriceTranslation
MobileHomeAdmin (class in bookings.admin), 39	attribute), 30
model (bookings.admin.PriceAdminForm.Meta at-	objects (bookings.models.Price attribute), 20
tribute), 36	

objects	(bookings.models.SeasonInfoTranslation attribute), 32	price_1_person_with_electricity ings.models.Price attribute), 19	(book-
objects	(bookings.models.SupplementMobileHomeTransle attribute), 33		(book-
objects		price_2_persons_with_electricity ings.models.Price attribute), 19	(book-
objects	(core.models.CampingInfoTranslation attribute), 8	<pre>price_2_persons_without_electricity     ings.models.Price attribute), 19</pre>	(book-
	(core.models.FoodInfoTranslation attribute), 9 (core.models.LaundryInfoTranslation attribute),	<pre>price_tourist_tax (bookings.models.Other</pre>	Price at-
	10	<pre>price_tourist_tax</pre>	(book-
	(core.models.SwimmingPoolInfoTranslation attribute), 10	ings.models.OtherPriceTranslation 30	attribute),
	g (bookings.admin.BookingAdmin attribute), 38	PriceAdmin (class in bookings.admin), 37	
	g (bookings.admin.PriceAdmin attribute), 37	PriceAdminForm (class in bookings.admin), 3	
	ice (class in bookings.models), 20	PriceAdminForm.Meta (class in bookings.adm	
	ice.DoesNotExist, 20	prices (bookings.models.SupplementPrice att	
OtherPr	ice.MultipleObjectsReturned, 20 iceAdmin( <i>class in bookings.admin</i> ), 38	priority (core.sitemaps.MultilingualStaticSi tribute), 12	temap at-
	<pre>iceTranslation (class in bookings.models), 29</pre>	<pre>privacy_view() (in module core.views), 12 pytestmark (bookings.tests.test_forms.TestBookings.test_forms.TestBookings.test</pre>	okingDetailsForm
	iceTranslation.DoesNotExist,30	attribute), 41	
OtherPr	iceTranslation.MultipleObjectsReturned $30$	'R	
Р		<pre>readonly_fields (bookings.admin.Bookings tribute), 38</pre>	Admin at-
pages	(core.sitemaps.MultilingualStaticSitemap attribute), 12	reservation_request_view() (in module tions.views), 14	reserva-
pet_pri	ce (bookings.models.SupplementPrice attribute), 17	ReservationRequestForm (class in tions.forms), 14	reserva-
pets (bo	okings.models.Booking attribute), 24	reservations.apps	
	ookings.models.Booking attribute), 23	module, 15	
pizza_f	ood_days (core.models.FoodInfo attribute), 6	reservations.forms	
pizza_f	ood_days (core.models.FoodInfoTranslation at-	module, 14	
	tribute), 9	reservations.tests.test_forms	
pool_op	ening_end (core.models.SwimmingPoolInfo at-	module, 16	
_	tribute), 5	reservations.tests.test_views	
pool_op	ening_end(core.models.SwimmingPoolInfoTrans		
_	attribute), 10	reservations.views	
pool_op	ening_start (core.models.SwimmingPoolInfo	module, 14	
,	attribute), 5	ReservationsConfig (class in reservations.a	pps), 15
	ening_start(core.models.SwimmingPoolInfoTro attribute), 11		
	end (core.models.CampingInfo attribute), 4	S	
_	end (core.models.CampingInfoTranslation attribute), 8	<pre>save() (bookings.models.Booking method), 24 save() (bookings.models.MobileHome method)</pre>	
	start (core.models.CampingInfo attribute), 4	save() (bookings.models.Price method), 19	,,
portal_	start (core.models.CampingInfoTranslation at-	save() (core.models.FoodInfo method), 5	
	tribute), 8	search_fields (bookings.admin.BookingAd	dmin at-
_	code (bookings.models.Booking attribute), 23	tribute), 38	
	lass in bookings.models), 18	search_fields (bookings.admin.Capa	cityAdmin
	oesNotExist, 19	attribute), 36	
Price.M	ultipleObjectsReturned, 19	search_fields (bookings.admin.MobileHeatribute), 39	omeAdmin

search_fields (bookings.admin.PriceAdmin attribute),	Т
37	tent_length (bookings.models.Booking attribute), 23
search_fields (book-	tent_width (bookings.models.Booking attribute), 23
ings.admin.SupplementPriceAdmin attribute), 36	test_about_view() (in module core.tests.test_views), 13
season (bookings.models.Price attribute), 18	test_accommodations_view() (in module
SEASON_CHOICES (bookings.models.Price attribute), 18	core.tests.test_views), 13
SeasonInfo (class in bookings.models), 28	test_activities_view() (in module
SeasonInfo.DoesNotExist, 29	core.tests.test_views), 13
SeasonInfo.MultipleObjectsReturned, 29	<pre>test_booking_capacity_validation() (in module</pre>
SeasonInfoAdmin (class in bookings.admin), 39	$bookings.tests.test\_models), 40$
SeasonInfoTranslation (class in bookings.models),	${\tt test\_booking\_confirm\_saves\_booking\_and\_sends\_emails()}$
30	(in module bookings.tests.test_views), 41
SeasonInfoTranslation.DoesNotExist, 30	<pre>test_booking_details_creates_stripe_session()</pre>
SeasonInfoTranslation.MultipleObjectsReturned,	(in module bookings.tests.test_views), 41
30 services_view() (in module core.views), 11	test_booking_form_valid() (in module book-
	ings.tests.test_views), 41
set_booking_session() (in module book- ings.tests.test_views), 40	test_booking_save_total_and_deposit() (in mod-
start_date (bookings.models.Booking attribute), 23	ule bookings.tests.test_models), 40
SUBTYPE_CHOICES (bookings.models.Booking attribute),	test_booking_summary_displays_correct_prices()
22	(in module bookings.tests.test_views), 41
SUBTYPE_TO_MAIN_TYPE (book-	test_campinginfo_fixture() (in module
ings.forms.BookingFormClassic attribute),	core.tests.test_models), 13
34	test_capacity_str() (in module book-ings.tests.test_models), 40
SupplementMobileHome (class in bookings.models), 27	test_foodinfo_fixture() (in module
SupplementMobileHome.DoesNotExist, 28	core.tests.test_models), 13
SupplementMobileHome.MultipleObjectsReturned,	test_form_cable_requires_if_electricity_yes()
28	(in module reservations.tests.test_forms), 16
SupplementMobileHomeAdmin (class in book-	test_form_dates_invalid() (in module reserva-
ings.admin), 39	tions.tests.test_forms), 16
SupplementMobileHomeTranslation (class in book-	test_form_missing_required_field() (in module
ings.models), 32	reservations.tests.test_forms), 16
${\tt SupplementMobileHomeTranslation.DoesNotExist},$	test_form_tent_requires_dimensions() (in mod-
32	ule reservations tests test forms) 16
SupplementMobileHomeTranslation.MultipleObject	tsReturned test_form_valid() (in module reserva-
32	tions.tests.test_forms), 16
SupplementPrice (class in bookings.models), 16	<pre>test_form_vehicle_requires_length() (in module</pre>
SupplementPrice.DoesNotExist, 17	reservations.tests.test_forms), 16
SupplementPrice.MultipleObjectsReturned, 17	<pre>test_get_reservation_request_view() (in module</pre>
SupplementPriceAdmin (class in bookings.admin), 36	reservations.tests.test_views), 16
supplements (bookings.models.Price attribute), 19	test_home_view() (in module core.tests.test_views), 13
<pre>supplements() (in module bookings.tests.test_views),</pre>	<pre>test_infos_view() (in module core.tests.test_views),</pre>
40	13
supplements_id (bookings.models.Price attribute), 20	test_invalid_phone() (book-
SwimmingPoolInfo (class in core.models), 4	$ings.tests.test\_forms. TestBooking Details Form$
SwimmingPoolInfo.DoesNotExist, 5	method), 41
SwimmingPoolInfo.MultipleObjectsReturned, 5	test_invalid_type_field() (book-
SwimmingPoolInfoTranslation (class in	ings.tests.test_forms.TestBookingFormClassic
<pre>core.models), 10 SwimmingPoolInfoTranslation.DoesNotExist, 10</pre>	method), 41
SwimmingPoolInfoTranslation.MultipleObjectsRe	test_laundryinfo_fixture() (in module
10	core.tests.test_models), 15
10	<pre>test_legal_view() (in module core.tests.test_views),</pre>
	13

test_missing_required_field() (book-	translations (core.models.LaundryInfo attribute), 6
ings.tests.test_forms.TestBookingFormClassic method), 41	translations (core.models.SwimmingPoolInfo attribute), 4
test_missing_required_fields() (book-	TYPE_CHOICES (bookings.models.Booking attribute), 22
ings.tests.test_forms.TestBookingDetailsForm method), 41	TYPE_CHOICES (bookings.models.Price attribute), 18
test_mobilehome_creation() (in module book- ings.tests.test_models), 40	U
test_not_found_view() (in module	updated_at (bookings.models.Booking attribute), 24
core.tests.test_views), 13	<pre>updated_at_display()</pre>
test_past_start_date() (book-	
ings.tests.test_forms.TestBookingFormClassic	V
method), 41	<pre>valid_booking_data() (in module book-</pre>
test_post_invalid_reservation() (in module	ings.tests.test_views), 40
<pre>reservations.tests.test_views), 16 test_post_valid_reservation_with_datetime_and</pre>	valid_form_data() (in module reserva- label() tions.tests.test_forms), 16
(in module reservations.tests.test_views), 16	valid_reservation_data() (in module reserva-
test_price_save_and_clean() (in module book-	tions.tests.test_views), 16
ings.tests.test_models), 40	vehicle_length (bookings.models.Booking attribute),
test_privacy_view() (in module	23
<pre>core.tests.test_views), 13 test_seasoninfo_creation() (in module book-</pre>	verbose_name (bookings.apps.BookingsConfig attribute), 40
$ings.tests.test\_models), 40$	verbose_name (core.apps.CoreConfig attribute), 12
test_services_view() (in module	visitor_price_with_swimming_pool (book-
core.tests.test_views), 13	ings.models.SupplementPrice attribute),
test_supplementmobilehome_creation() (in mod-	17
ule bookings.tests.test_models), 40	<pre>visitor_price_without_swimming_pool (book-</pre>
test_supplementprice_creation() (in module bookings.tests.test_models), 40	ings.models.SupplementPrice attribute), 17
test_swimmingpoolinfo_fixture() (in module core.tests.test_models), 13	W
test_valid_data() (book-	washing_machine_price (core.models.LaundryInfo at-
ings.tests.test_forms.TestBookingDetailsForm	tribute), 7
method), 41	washing_machine_price
test_valid_data_tent() (book-	(core.models.LaundryInfoTranslation at-
$ings.tests.test\_forms.TestBookingFormClassic$	tribute), 10 week_high (bookings.models.MobileHome attribute), 27
method), 41	week_low (bookings.models.MobileHome attribute), 27 week_low (bookings.models.MobileHome attribute), 27
TestBookingDetailsForm (class in book-	week_mid (bookings.models.MobileHome attribute), 27 week_mid (bookings.models.MobileHome attribute), 27
ings.tests.test_forms), 41	weekend_price_with_electricity (book-
TestBookingFormClassic (class in book-	ings.models.Price attribute), 19
<pre>ings.tests.test_forms), 41 tourist_tax_date</pre>	weekend_price_without_electricity (book-
tourist_tax_date (bookings.models.OtherPrice attribute), 21	ings.models.Price attribute), 19
tourist_tax_date (book-	${\tt welcome\_afternoon\_end}  ({\it core.models.CampingInfo}$
ings.models.OtherPriceTranslation attribute),	attribute), 4
30	welcome_afternoon_end
translations (bookings.models.OtherPrice attribute),	(core.models.CampingInfoTranslation at- tribute), 8
translations (bookings.models.SeasonInfo attribute),	${\tt welcome\_afternoon\_start}~(core.models. Camping Info$
28	attribute), 4
translations(bookings.models.SupplementMobileHome	welcome_afternoon_start
attribute), 28	(core.models.CampingInfoTranslation at-
translations (core.models.CampingInfo attribute), 3	tribute), 8
translations (core models FoodInfo attribute) 5	welcome_end (core.models.CampingInfo attribute), 4

- welcome\_end (core.models.CampingInfoTranslation attribute), 8
- welcome\_start (core.models.CampingInfo attribute), 4
- $\begin{tabular}{ll} welcome\_start & (core.models.CampingInfoTranslation \\ & attribute), \ 8 \end{tabular}$
- widgets (bookings.forms.BookingFormClassic.Meta attribute), 34
- worker\_price\_1p (bookings.models.MobileHome attribute), 27
- worker\_price\_2p (bookings.models.MobileHome attribute), 27
- worker\_price\_3p (bookings.models.MobileHome attribute), 27