Coverage report: 100%

filter...

coverage.py v6.5.0, created at 2023-01-13 15:59 -0300

Module	*statements*	*missing*	*excluded*	*coverage*
core/forms.py	20	0	0	100%
core/models.py	112	0	0	100%
core/translation.py	23	0	0	100%
core/views.py	33	0	0	100%
Total	**188**	**0**	**0**	**100%**

coverage.py v6.5.0, created at 2023-01-13 15:59 -0300

Coverage for **core/models.py**: 100%

112 statements | 112 run | 0 missing | 0 excluded

« prev ^ index » next coverage.py v6.5.0, created at 2023-01-13 15:59 -0300

```python
from django.core.validators import MinValueValidator, MaxValueValidator
from django.db import models
from django.utils.translation import gettext_lazy as _

from stdimage import StdImageField

import uuid


def get_file_path(_instance, filename):
    ext = filename.split('.')[-1]
    filename = f'{uuid.uuid4()}.{ext}'
    return filename


class Base(models.Model):
    criados = models.DateField(_('Criado'), auto_now_add=True)
    modificado = models.DateField(_('Atualizado'), auto_now=True)
    ativo = models.BooleanField(_('Ativo?'), default=True)

    class Meta:
        abstract = True


class Service(Base):
    ICONE_CHOICES = (

        ('fa fa-2x fa-laptop service-icon bg-primary text-white mr-3', 'Laptop'),
        ('fa fa-2x fa-laptop-code service-icon bg-primary text-white mr-3', 'Laptop-code'),
        ('fab fa-2x fa-android service-icon bg-primary text-white mr-3', 'Android'),
        ('fab fa-2x fa-apple service-icon bg-primary text-white mr-3', 'Apple'),
        ('fa fa-2x fa-search service-icon bg-primary text-white mr-3', 'Search'),
        ('fa fa-2x fa-edit service-icon bg-primary text-white mr-3', 'Edit'),
        ('fa fa-2x fa-solid fa-robot service-icon bg-primary text-white mr-3', 'Robot'),
        ('fa fa-2x fa-solid fa-bolt service-icon bg-primary text-white mr-3', 'Lightning'),
        ('fa fa-2x fa-cog service-icon bg-primary text-white mr-3', 'Gear'),

    )

    servico = models.CharField(_('Serviço'), max_length=100)
    descricao = models.TextField(_('Descrição'), max_length=200)
    icone = models.CharField(_('Icone'), max_length=100, choices=ICONE_CHOICES)

    class Meta:
        verbose_name = _('Serviço')
        verbose_name_plural = _('Serviços')

    def __str__(self):
        return self.servico


class Education(Base):

    titulo = models.CharField(_('Título'), max_length=100)
    subtitulo = models.CharField(_('Subtítulo'), max_length=100)
    inicio = models.IntegerField(_('Ano início'), validators=[MinValueValidator(1900), MaxValueValidator(3000)])
    fim = models.IntegerField(_('Ano fim'),  validators=[MinValueValidator(1900), MaxValueValidator(3000)])
    descricao = models.TextField(_('Descrição'), max_length=200)

    class Meta:
        verbose_name = _('Educação')
        verbose_name_plural = _('Educação')

    def __str__(self):
        return self.titulo


class Experience(Base):
    titulo = models.CharField(_('Título'), max_length=100)
```

```
70      subtitulo = models.CharField(_('Subtítulo'), max_length=100)
71      inicio = models.IntegerField(_('Ano início'), validators=[MinValueValidator(1900), MaxValueValidator(3000)])
72      fim = models.IntegerField(_('Ano fim'), validators=[MinValueValidator(1900), MaxValueValidator(3000)])
73      descricao = models.TextField(_('Descrição'), max_length=300)
74
75      class Meta:
76          verbose_name = _('Experiência')
77          verbose_name_plural = _('Experiências')
78
79      def __str__(self):
80          return self.titulo
81
82
83  class Endereco(Base):
84      pais = models.CharField(_('País'), max_length=15)
85      estado = models.CharField(_('Estado'), max_length=15)
86      cidade = models.CharField(_('Cidade'), max_length=20)
87
88      class Meta:
89          verbose_name = _('Endereço')
90          verbose_name_plural = _('Endereço')
91
92      def __str__(self):
93          return self.cidade
94
95
96  class Sobre(Base):
97      titulo = models.CharField(_('Título'), max_length=100)
98      descricao = models.TextField(_('Descrição'), max_length=400)
99      nome = models.CharField(_('Nome'), max_length=40)
100     escolaridade = models.TextField(_('Escolariade'), max_length=45)
101     telefone = models.CharField(_('Telefone'), max_length=20)
102     endereco = models.ForeignKey('core.Endereco', verbose_name=_('Endereço'), on_delete=models.CASCADE)
103     nascimento = models.DateField(_('Nascimento'))
104     # experia_anos = models.CharField('Experiência', max_length=10)
105     email = models.EmailField(max_length=254)
106     # freelance = models.CharField('Freelance', max_length=15)
107     imagem = StdImageField(_('Imagem'), upload_to=get_file_path,
108                            variations={'thumb': {'width': 600, 'height': 600, 'crop': True}})
109
110     class Meta:
111         verbose_name = _('Sobre')
112         verbose_name_plural = _('Sobre')
113
114     def __str__(self):
115         return self.titulo
116
117
118 class Competencia(Base):
119     competencia = models.CharField(_('Competência'), max_length=40)
120
121     class Meta:
122         verbose_name = _('Competência')
123         verbose_name_plural = _('Competências')
124
125     def __str__(self):
126         return self.competencia
127
128
129 class HeaderPicture(Base):
130     foto = StdImageField(_('Imagem'), upload_to=get_file_path,
131                          variations={'thumb': {'width': 600, 'height': 600, 'crop': True}})
132
133     class Meta:
134         verbose_name = _('Foto Header')
135         verbose_name_plural = _('Fotos Header')
136
137     def __str__(self):
138         return f'{self.foto}'
139
140
141 class Skill(Base):
142     BAR_CHOICES = (
143         ('progress-bar bg-primary', _('Verde')),
144         ('progress-bar bg-warning', _('Amarelo')),
145         ('progress-bar bg-danger', _('Vermelho')),
```

```
146         ('progress-bar bg-dark', _('Preto')),
147         ('progress-bar bg-info', _('Azul')),
148     )
149 
150     nome = models.CharField(_('Nome'), max_length=50)
151     porcentagem = models.IntegerField(_('Porcentagem'), validators=[MinValueValidator(0), MaxValueValidator(100)]
152     cor = models.CharField(_('Cor'), max_length=100, choices=BAR_CHOICES)
153 
154     class Meta:
155         verbose_name = _('Habilidade')
156         verbose_name_plural = _('Habilidades')
157 
158     def __str__(self):
159         return f'{self.nome}'
160 
161 
162 class FiltrosProjetos(Base):
163     nome = models.CharField('Nome_filtro', max_length=30)
164 
165     class Meta:
166         verbose_name = _('FiltroProjeto')
167         verbose_name_plural = _('FiltrosProjeto')
168 
169     def __str__(self):
170         return f'{self.nome}'
171 
172 
173 class Projetos(Base):
174     nome = models.CharField(_('Nome'), max_length=50)
175     tag = models.ManyToManyField(FiltrosProjetos)
176     imagem = StdImageField(_('Imagem'), upload_to=get_file_path,
177                            variations={'thumb': {'width': 400, 'height': 300, 'crop': True}})
178     link = models.CharField(_('Link'), max_length=200)
179 
180     class Meta:
181         verbose_name = _('Projeto')
182         verbose_name_plural = _('Projetos')
183 
184     def __str__(self):
185         return f'{self.nome}'
```

Coverage for **core/forms.py**: 100%

20 statements | 20 run | 0 missing | 0 excluded

« prev ^ index » next coverage.py v6.5.0, created at 2023-01-13 15:51 -0300

```
1  from django import forms
2  from django.core.mail.message import EmailMessage
3  from django.utils.translation import gettext_lazy as _
4
5
6  class ContatoForm(forms.Form):
7      nome = forms.CharField(label=_('Nome'), max_length=100)
8      email = forms.CharField(label=_('E-mail'), max_length=100)
9      assunto = forms.CharField(label=_('Assunto'), max_length=100)
10     mensagem = forms.CharField(label=_('Mensagem'), widget=forms.Textarea())
11
12     def send_mail(self):
13         nome = self.cleaned_data['nome']
14         email = self.cleaned_data['email']
15         assunto = self.cleaned_data['assunto']
16         mensagem = self.cleaned_data['mensagem']
17
18         n = _('Nome')
19         e = _('E-mail')
20         a = _('Assunto')
21         m = _('Mensagem')
22
23         conteudo = f'{n}: {nome}\n{e}: {email}\n{a}: {assunto}\n{m}: {mensagem}
24
25         mail = EmailMessage(
26             subject=assunto,
27             body=conteudo,
28             from_email='contato@curriculo.com.br',
29             to=['contato@fusion.com.br', ],
30             headers={'Reply-To': email}
31         )
32         mail.send()
```

« prev ^ index » next coverage.py v6.5.0, created at 2023-01-13 15:51 -0300

Coverage for **core/translation.py**: 100%

23 statements | 23 run | 0 missing | 0 excluded

« prev ^ index » next coverage.py v6.5.0, created at 2023-01-13 15:51 -0300

```
1  from .models import Education, Service, Experience, Sobre, Competencia, FiltrosProjetos, Projeto
2
3  from modeltranslation.translator import TranslationOptions, register
4
5
6  @register(Education)
7  class EducationTranslationOptions(TranslationOptions):
8      fields = ('titulo', 'subtitulo', 'descricao')
9
10
11 @register(Service)
12 class ServiceTranslationOptions(TranslationOptions):
13     fields = ('servico', 'descricao')
14
15
16 @register(Experience)
17 class ExperienceTranslationOpions(TranslationOptions):
18     fields = ('titulo', 'subtitulo', 'descricao')
19
20
21 @register(Sobre)
22 class SobreTranslationOpions(TranslationOptions):
23     fields = ('titulo', 'descricao', 'escolaridade')
24
25
26 @register(Competencia)
27 class CompetenciaTranslationOpions(TranslationOptions):
28     fields = ('competencia',)
29
30
31 @register(FiltrosProjetos)
32 class FiltrosProjetoTranslationOpions(TranslationOptions):
33     fields = ('nome',)
34
35
36 @register(Projetos)
37 class ProjetosTranslationOpions(TranslationOptions):
38     fields = ('nome',)
```

« prev ^ index » next coverage.py v6.5.0, created at 2023-01-13 15:51 -0300

Coverage for **core/views.py**: 100%

33 statements | 33 run | 0 missing | 0 excluded

« prev ^ index » next coverage.py v6.5.0, created at 2023-01-13 15:51 -0300

```
1  from django.views.generic import FormView
2  from django.urls import reverse_lazy
3  from django.contrib import messages
4  from django.utils.translation import gettext as _
5  from django.utils import translation
6
7  from .models import Service, Experience, Education, Sobre, HeaderPicture, Competencia, Skill, FiltrosProjetos, Projeto
8
9  from .forms import ContatoForm
10
11
12 class IndexView(FormView):
13     template_name = 'index.html'
14     form_class = ContatoForm
15     success_url = reverse_lazy('index')
16
17     def get_context_data(self, **kwargs):
18         context = super(IndexView, self).get_context_data(**kwargs)
19         lang = translation.get_language()  # Pega a linguagem pelo navegador e adiciona na variável
20         context['servicos'] = Service.objects.order_by('?').all()
21         context['educacao'] = Education.objects.all()
22         context['experiencia'] = Experience.objects.all()
23         context['sobre'] = Sobre.objects.all()
24         context['picture'] = HeaderPicture.objects.all()
25         context['competencia'] = Competencia.objects.order_by('?').all()
26         context['habilidade'] = Skill.objects.order_by('?').all()
27         context['filtros'] = FiltrosProjetos.objects.order_by('?').all()
28         context['projetos'] = Projetos.objects.order_by('?').all()
29         context['lang'] = lang
30         translation.activate(lang)
31
32         return context
33
34     def form_valid(self, form, *args, **kwargs):
35         form.send_mail()
36         messages.success(self.request, _('E-mail enviado com sucesso!'))
37         return super(IndexView, self).form_valid(form, *args, **kwargs)
38
39     def form_invalid(self, form, *args, **kwargs):
40         messages.error(self.request, _('Erro ao enviar e-mail.'))
41         return super(IndexView, self).form_invalid(form, *args, **kwargs)
```

« prev ^ index » next coverage.py v6.5.0, created at 2023-01-13 15:51 -0300