その他ファイル

ここではsystemのその他ファイルについて説明する.

admin.py

adminサイトを管理するファイルである.

```
@admin.register(User)
class AdminUserAdmin(UserAdmin):
  fieldsets = (
    (None, {'fields': ('username', 'password')}),
    (_('Personal info'), {'fields': ('first_name','last_name',
'email','phone')}),
    (_('Permissions'), {'fields': ('is_active', 'is_staff',
'is_superuser','groups', 'user_permissions')}),
    (_('Important dates'), {'fields': ('last_login', 'date_joined')}),
  )
  list_display = ('username', 'email', 'first_name','last_name',
'is_staff')
  search_fields = ('username', 'first_name','last_name',  'email')
  filter_horizontal = ('groups', 'user_permissions')
```

上はadminページのユーザーページの設定である. Noneは一番上のブロック, *('Personal nfo')は次の個人情報のブロック*,('Permissions')は次のパーミッションのブロックのようにブロックで分かれていて, そのブロックの項目がfieldsである.

```
admin.site.register(~~~) # ~~~モデルをadminページで管理できるようにする
```

参考

- ユーザーカスタマイズ : https://qiita.com/okoppe8/items/10ae61808dc3056f9c8e
- 管理画面逆引き : https://qiita.com/zenwerk/items/044c149d93db097cdaf8

forms.py

ここではsystem/form.pyの説明をする.
forms.pyはwebページのフォーム(入力窓)を管理するファイルである.

```
#Bootstrap
  def __init__(self, *args, **kwargs):
    super().__init__(*args, **kwargs)
    for field in self.fields.values():
      field.widget.attrs['class'] = 'form-control'
```

上はフォームにBootstrapを適応するための処理である.

class Metaを使う事で，models.pyからmodelを持ってきて，フォームの項目が持ってきたmodelの要素になる.
widgetsは何も入力されていないフォームに例示を表示するためのディクショナリである.

前へ 目次