まとめ

リダイレクト・・・redirect(from django.shortcuts import redirect)を用いて行う。
例) return redirect('app:sample', id=1) # app_nameがappでnameがsampleの関数にid=1を引数として遷移する

エラーハンドリング・・・DEBUGをFalseにして、プロジェクトのurls.pyに設定したいステータスコードのハンドラーを追加する
handler404 = views.関数名(404エラーの場合に実行したい関数を指定(関数には引数を2つとる))

モデルを呼び出し、値が存在しない場合は404を返す・・・get_object_or_404, get_list_or_404

ログインのパスワード暗号化・・・settings.pyにPASSWORD_HASHERS変数を追加する

django.contrib.auth.authenticate(username, password): パスワードが正しいかチェック。
django.contrib.auth.login(request, user): ログインを行う
django.contrib.auth.logout: ログアウトを行う
django.contrib.auth.decorators.login_required: ログインが必要な関数にデコレータとして付与すると、ログインしていない場合にはエラーにできる
{% if user.is_authenticated %}: ログインしている場合だけ、実行される。

まとめ

validate_password(password, user):ユーザのパスワードが適切かチェックをする

バリデーターの自作・・・クラスを作成して、中に__init__, validate, get_help_textを定義して、settings.pyのAUTH_PASSWORD_VALIDATORSで、作成したバリデーターを指定する。

AbstractUser: すでに存在するフィールドをそのまま流用して、usernameフィールドを削除したい場合用いるとよい
AbstractBaseUser: 初めからUserを作り変えたい場合に用いる

まとめ

admin画面のカスタマイズ

fields: 編集画面で表示するフィールドの順番を変更する。
search_fields: 検索に利用したいフィールドを記述する。
list_filter: 特定のフィールドでフィルターをできるようにする。
list_display: 一覧画面で表示するフィールドを指定する。
list_display_links: 編集画面への遷移をするリンクに指定するフィールドを変更する。
list_editable: 一覧画面で編集できるようにするフィールドを指定する。

modelsに追加する要素
class Metaに以下のことを追加することで、表示内容を変えることができる
ordering: 画面の並びを変える
verbose_name_plural: 管理画面の表示を変える

テンプレートを利用して、管理画面を上書き修正する。
https://github.com/django/django/tree/master/django/contrib/admin/templates
admin: 管理画面ページ一般
registration: ログアウト、パスワード変更等