Djagno 기반의 웹프로그래밍

이 소 영

yisy0703@naver.com

Web Frameworks for Python(https://wiki.python.org/moin/WebFrameworks)
장고 공식 사이트(https://www.djangoproject.com/)
장고 공식 소스 저장소(http://github.com/django/django)
장고 참조 문서 영어(https://docs.djangoproject.com/en/3.2/)

Agenda

Django(장고) 기반의 파이썬 웹 프로그래밍

Ch 09 사용자 인증

1. Auth App
2. 회원가입 구현
3. 로그인/아웃 구현
4. Qauth 라이브러리 활용

Chapter 09. 사용자 인증

- AUTHENTICATION(https://github.com/django/django/blob/master/django/conf/global_settings.py#LC515)
- auth/models.py(https://github.com/django/django/blob/master/django/contrib/auth/models.py#LC324)
- User 주요 함수(https://github.com/django/django/blob/master/django/contrib/auth/base_user.py#LC47)
- auth/context_processors.py(https://github.com/django/django/blob/master/django/contrib/auth/context_processors.py#LC46)

auth/forms.py

- 회원가입 폼(https://github.com/django/django/blob/3.2/django/contrib/auth/forms.py#LC75)
- 로보 폼(https://github.com/django/django/blob/3.2/django/contrib/auth/forms.py#LC160)
- 암호변경 리셋 요청폼(https://github.com/django/django/blob/3.2/django/contrib/auth/forms.py#LC238)
- 암호 변경폼(https://github.com/django/django/blob/3.2/django/contrib/auth/forms.py#LC360)
- 암호 설정폼(https://github.com/django/django/blob/3.2/django/contrib/auth/forms.py#LC316)

Signals

- Signals](https://docs.djangoproject.com/en/3.2/ref/signals/)
- dispatch/dispatcher.py(https://github.com/django/django/blob/master/django/dispatch/dispatcher.py)
- db/models/signals.py(https://github.com/django/django/blob/master/django/db/models/signals.py)
- auth/signals.py(https://github.com/django/django/blob/master/django/contrib/auth/signals.py)

Email

- Django Email library(https://docs.djangoproject.com/en/3.2/topics/email/)
- 장고 Email 기본 환경 변수(https://github.com/django/django/blob/master/django/conf/global_settings.py#L184)

auth/views.py

- auth/views.py(https://github.com/django/django/blob/master/django/contrib/auth/views.py)
- user_logged_in(https://github.com/django/django/blob/3.2/django/dispatch/dispatcher.py#LC24)

Session

- SessionMiddleware(https://github.com/django/django/blob/3.2/django/contrib/sessions/middleware.py)
- SessionStore(https://github.com/django/django/blob/master/django/contrib/sessions/backends/db.py)
- MiddlewareMixin(https://github.com/django/django/blob/3.2/django/utils/deprecation.py#LC82)

이메일 환경 변수

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.naver.com'
EMAIL_PORT = 465
EMAIL_HOST_USER = 'ID'
EMAIL_HOST_PASSWORD = '비밀번호'
EMAIL_USE_SSL = True
```

accounts/models.py

accounts/login_form.html

```
    <ul>
        {% for provider in providers %}
            <li>
                {% if provider.social_app %}
                    <a href="{% provider_login_url provider.id %}">{{provider.name}}</a>
                {% else %}
                    <a>
                        Provider {{provider.name}}설정이 필요합니다.
                    </a>
                {% endif %}
            </li>
        {% endfor %}
    </ul>
```

```
post_save.connect(on_send_mail, sender=settings.AUTH_USER_MODEL)
```

accounts/login_form.html

```
<ul>
    {% for provider in providers %}
        <li>
            {% if provider.social_app %}
                <a href="{% provider_login_url provider.id %}">{{provider.name}}</a>
            {% else %}
                <a>
                    Provider {{provider.name}}설정이 필요합니다.
                </a>
            {% endif %}
        </li>
    {% endfor %}
</ul>
```

1. Auth App

Auth 앱

1. Auth App

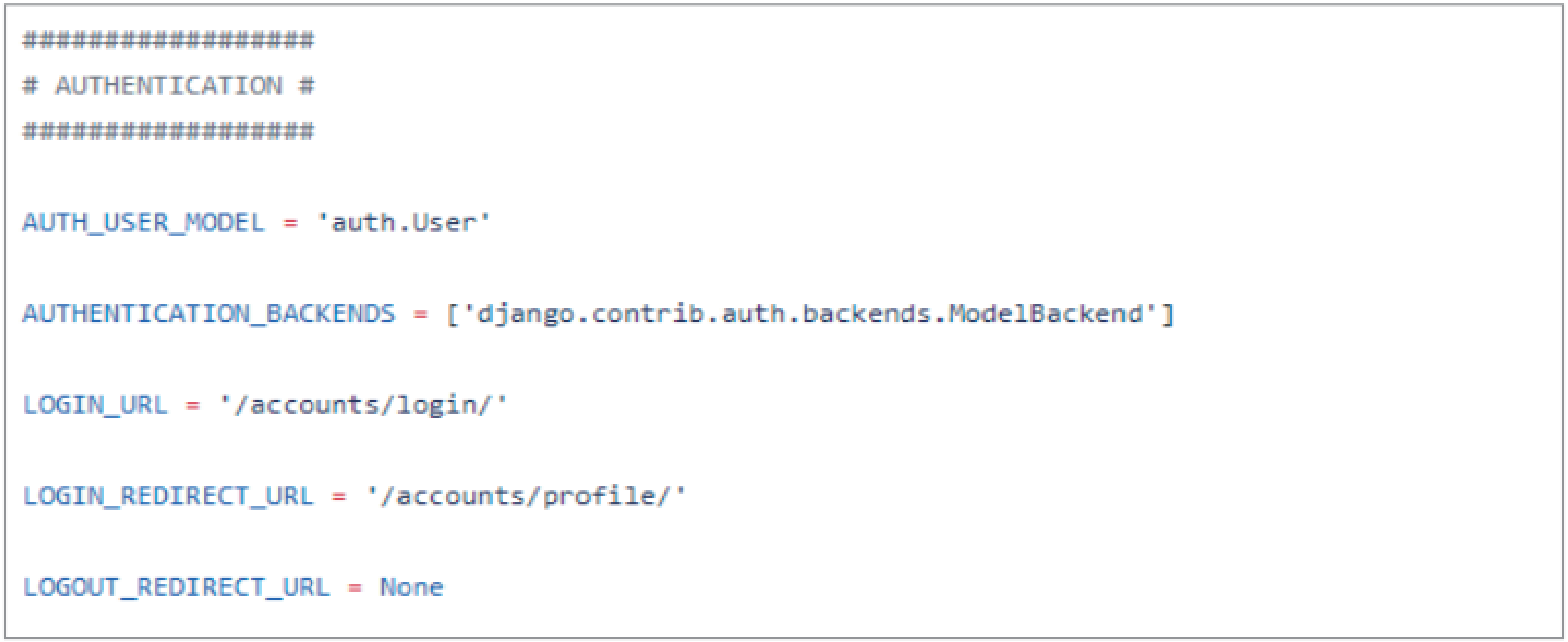
- LOGIN_URL = '/accounts/login/'
  - 기본 로그인 페이지 URL 지정
  - login_required 장식자에서 사용
- LOGIN_REDIRECT_URL = '/accounts/profile/'
  - 로그인 완료 후 next 인자가 없으면 이동되는 페이지 URL
  - next 가 지정된 경우는 해당 URL로 이동
- LOGOUT_REDIRECT_URL = None
  - 로그아웃 후 next_page가 지정되지 않은 경우 이동할 URL
  - next_page가 지정된 경우 해당 URL로 이동
  - next_page 미지정, LOGOUT_REDIRECT_URL=None이면 'registration/logged_out.html' 페이지 이동
- AUTH_USER_MODEL = 'auth.User'
  - 인증에 사용할 커스텀 User 모델 지정. '앱이름.모델명'

※ AUTHENTICATION
(https://github.com/django/django/blob/master/django/conf/global_settings.py#LC515)

Auth 앱

1. Auth App

- django.conf.global_settings.py



```
##################
# AUTHENTICATION #
##################

AUTH_USER_MODEL = 'auth.User'

AUTHENTICATION_BACKENDS = ['django.contrib.auth.backends.ModelBackend']

LOGIN_URL = '/accounts/login/'

LOGIN_REDIRECT_URL = '/accounts/profile/'

LOGOUT_REDIRECT_URL = None
```

※ AUTHENTICATION
(https://github.com/django/django/blob/master/django/conf/global_settings.py#LC515)

Auth 앱

1. Auth App

django/contrib/auth/base_user.py

```
class AbstractBaseUser(models.Model):
    password = models.CharField(max_length=128)
    last_login = models.DateTimeField(blank=True, null=True)
    is_active = True
    ~ 생략 ~
```

django/contrib/auth/models.py

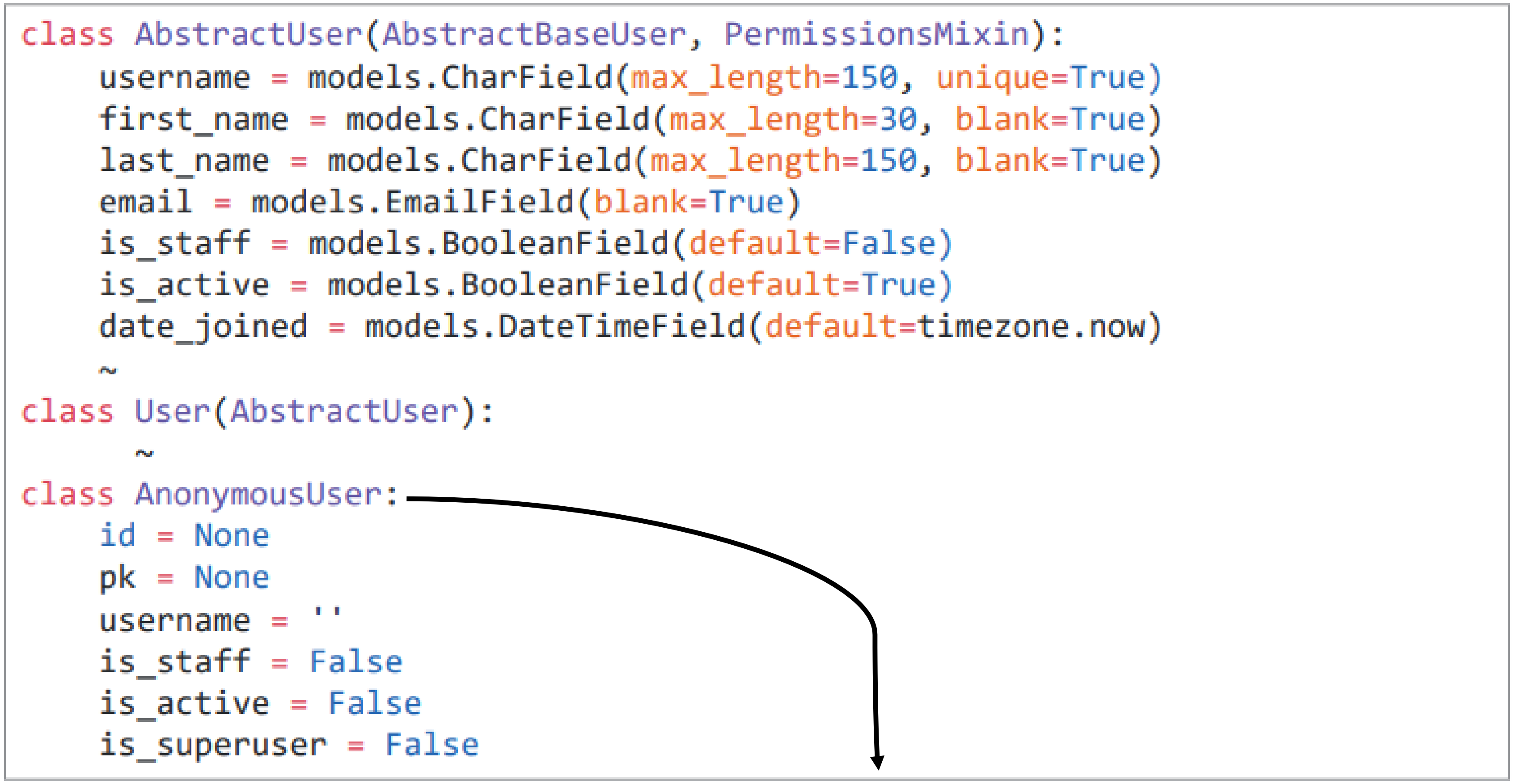
```
class PermissionsMixin(models.Model):
     is_superuser = models.BooleanField(False)
     ~ 생략 ~
```

- auth/models.py
  (https://github.com/django/django/blob/master/django/contrib/auth/models.py#LC324)

Auth 앱

1. Auth App

- django/contrib/auth/models.py



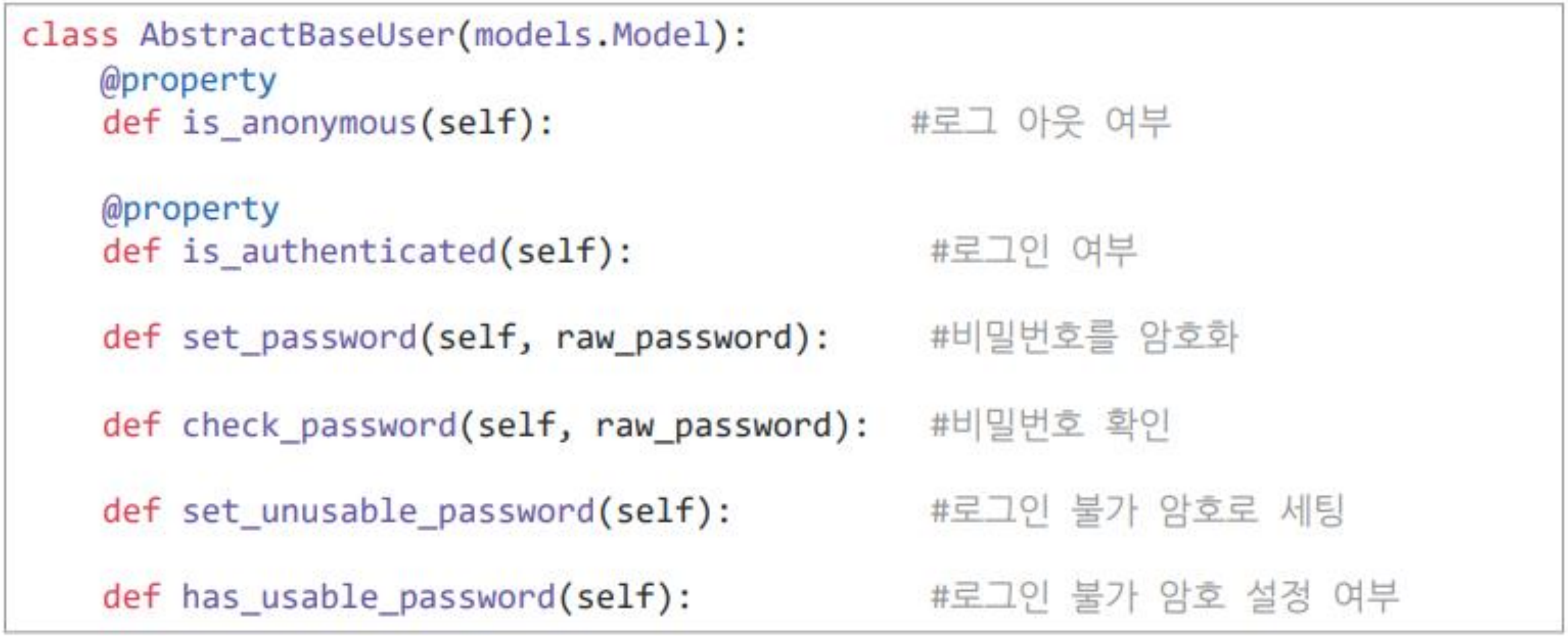
```
class AbstractUser(AbstractBaseUser, PermissionsMixin):
    username = models.CharField(max_length=150, unique=True)
    first_name = models.CharField(max_length=30, blank=True)
    last_name = models.CharField(max_length=150, blank=True)
    email = models.EmailField(blank=True)
    is_staff = models.BooleanField(default=False)
    is_active = models.BooleanField(default=True)
    date_joined = models.DateTimeField(default=timezone.now)
    ~
class User(AbstractUser):
      ~
class AnonymousUser:
    id = None
    pk = None
    username = ''
    is_staff = False
    is_active = False
    is_superuser = False
```

https://github.com/django/django/blob/main/django/contrib/auth/models.py#LC403

Auth 앱

1. Auth App

User 주요 속성과 함수



```
class AbstractBaseUser(models.Model):
    @property
    def is_anonymous(self):                         #로그 아웃 여부

    @property
    def is_authenticated(self):                     #로그인 여부

    def set_password(self, raw_password):           #비밀번호를 암호화

    def check_password(self, raw_password):         #비밀번호 확인

    def set_unusable_password(self):                #로그인 불가 암호로 세팅

    def has_usable_password(self):                  #로그인 불가 암호 설정 여부
```

Auth 앱

1. Auth App

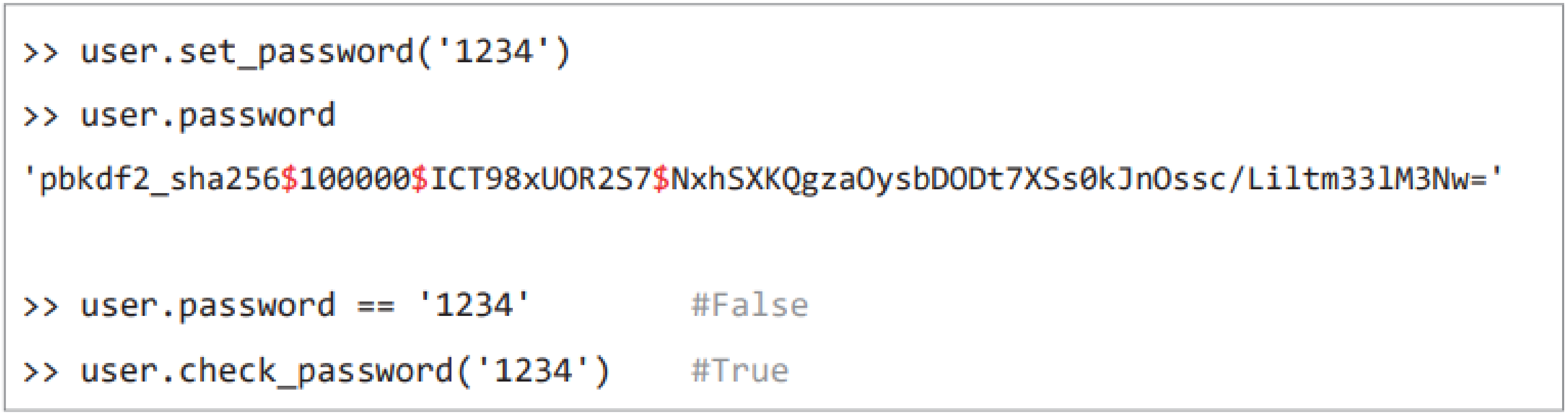
User 주요 속성과 함수

```
>> from django.contrib.auth import get_user_model
>> User = get_user_model()
>> user = User.objects.first()
>> user.is_authenticated      #True
>> user.is_anonymous          #False
>> user.password
'pbkdf2_sha256$100000$8j1afpleMsaN$0HudIy/MGGUSmx8VCPTeY+b91kSgNo/Fmfh1ISaP23A='
    해싱함수       횟수       seed           결과값
```

Auth 앱

1. Auth App

● User 주요 속성과 함수



```
>> user.set_password('1234')
>> user.password
'pbkdf2_sha256$100000$ICT98xUOR2S7$NxhSXKQgzaOysbDODt7XSs0kJnOssc/Liltm331M3Nw='


>> user.password == '1234'          #False
>> user.check_password('1234')      #True
```

set_unusable_password()

1. Auth App

- 로그인 불가 암호 세팅

- 외부 서비스 인증으로 가입된 회원의 경우, 직접 로그인은 불가.
  외부 서비스를 통한 로그인을 허용

- 외부 서비스 인증(OAutho) 연동
  - django-allauth 라이브러리
  - python-social-auth 라이브러리

```
>> user.set_unusable_password()
>> user.password
'!kNDbkLVYUUdOtqoMrlj0ewhsYoOR7vM2jfUNYrvZ'
>>user.check_password('1234')
False
```

User 모델 클래스

1. Auth App

- 방법1

```
from django.contrib.auth.models import User

User.objects.all()
```

- 방법2

```
 from django.contrib.auth import get_user_model

 User = get_user_model()
```

- 인증 User 모델을 다른 모델로 변경가능하기 때문에 방법2가 유연성이 있음

User 모델 변경

1. Auth App

방법1

- Profile 모델 생성후 User 모델과 1:1 관계 매핑

```
from django.conf import settings
from django.contrib.auth.models import User
from django.db import models

class Article(models.Model):
        # author = models.ForeignKey(User)
        # author = models.ForeignKey('auth.User')
        author = models.ForeignKey(settings.AUTH_USER_MODEL)
```

방법2

- Custom User 모델 생성
- django.contrib.auth.models.AbstractUser 상속
- Custom User 모델 설정 : '앱이름.모델명'
  - settings.AUTH_USER_MODEL 값으로 지정 (기본값: 'auth.User')

뷰에서 현재 로그인 유저 획득

- FBV 뷰 : request.user
- CBV 뷰 : self.request.user
- context_processors를 통해 user 제공
- 유저 타입
  - 로그인 상태 : settings.AUTH_USER_MODEL 클래스
  - 로그아웃 상태 : django.contrib.auth.models.AnonymousUser

```
# blog패키지의 views모듈
def index(request):
  print('로그인 한 사람 :', request.user)
  post_list = Post.objects.all()
  return render(request,
                'blog/index.html',
                {'post_list':post_list})
```

```
<li><a href="#">{{user}}님</a></li>
```

공통 템플릿

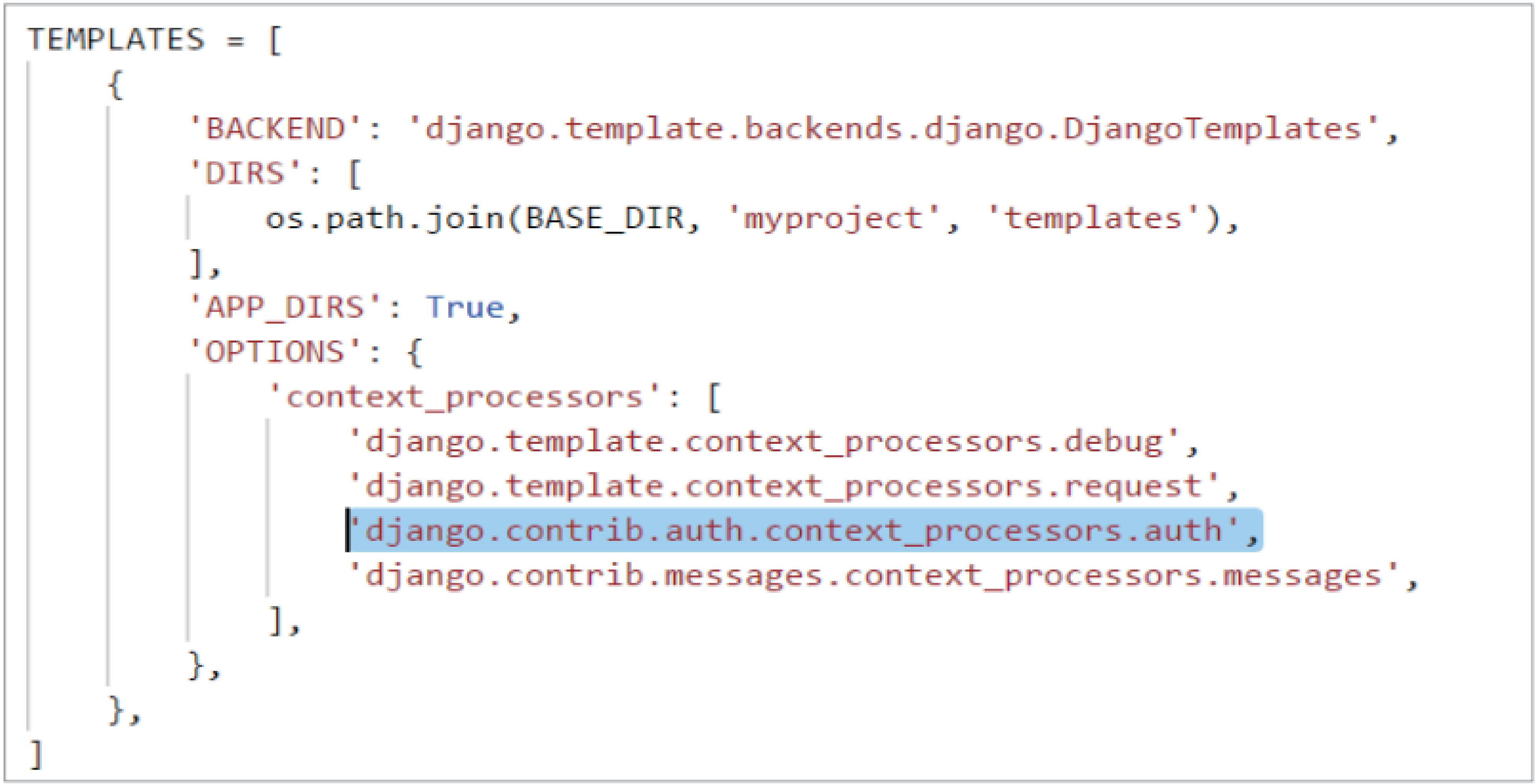
1. Auth App

```
# myproject/templates/layout.html

{% load static %}

<!doctype html>
<html>
<head>
        <meta charset="utf-8" />
        <title>{% block title %}TheBrains Article{% endblock %}</title>
</head>
<body>
    회원 가입 / 로그인 / {{user}}
    <h1>TheBrains</h1>

    {% block content %}
    {% endblock %}

    <hr/>
    Copyright © 2003 TheBrains All Right Reserved
</body>
</html>
```

로그인 시 : user.is_authenticated == True 또는 user.is_anonymous==False

context_processors

1. Auth App

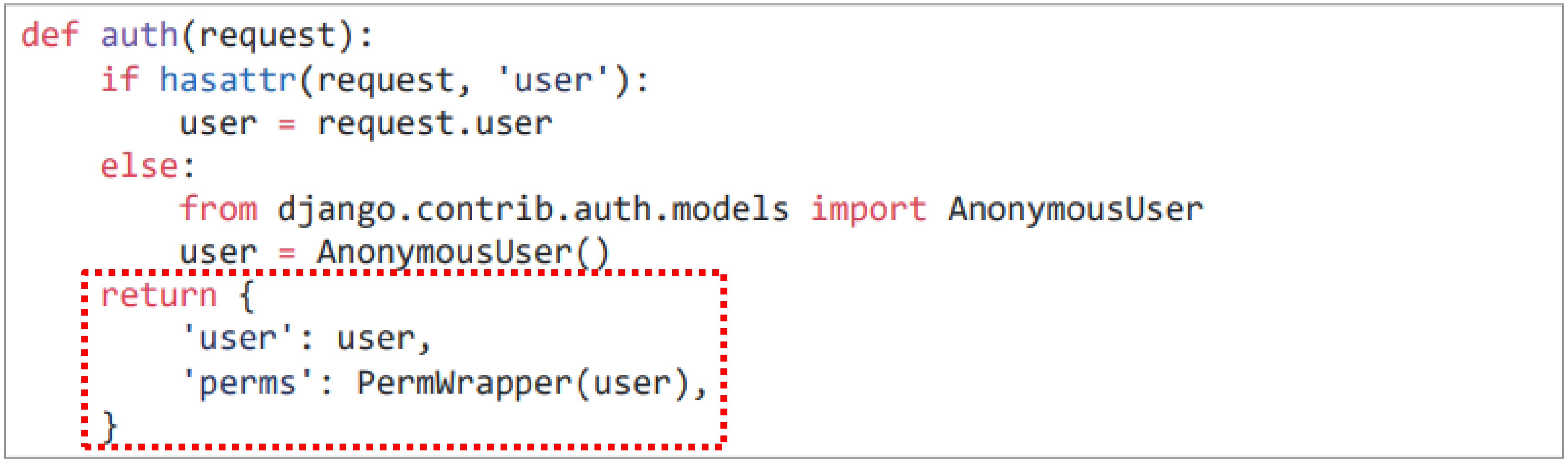


```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [
            os.path.join(BASE_DIR, 'myproject', 'templates'),
        ],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

context_processors

1. Auth App

- django.contrib.auth.context_processors.auth



```python
def auth(request):
    if hasattr(request, 'user'):
        user = request.user
    else:
        from django.contrib.auth.models import AnonymousUser
        user = AnonymousUser()
    return {
        'user': user,
        'perms': PermWrapper(user),
    }
```

https://github.com/django/django/blob/main/django/contrib/auth/context_processors.py#LC49

회원 인증 Form 클래스

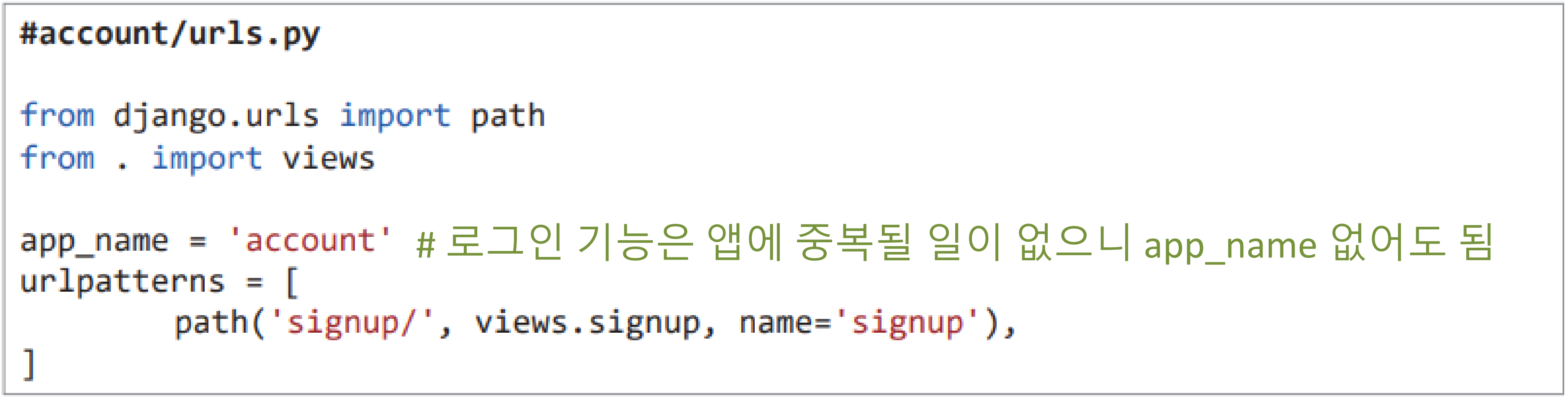
1. Auth App

- UserCreationForm : 회원가입
  - https://github.com/django/django/blob/main/django/contrib/auth/forms.py#LC78
- AuthenticationForm : 로그인
  - https://github.com/django/django/blob/main/django/contrib/auth/forms.py#LC163
- PasswordResetForm : 암호 변경 리셋 요청
  - https://github.com/django/django/blob/main/django/contrib/auth/forms.py#LC241
- PasswordChangeForm : 암호 변경
  - https://github.com/django/django/blob/main/django/contrib/auth/forms.py#LC363
- SetPasswordForm : 암호 설정
  - https://github.com/django/django/blob/main/django/contrib/auth/forms.py#LC319

2. 회원가입 구현

회원 가입 구현

2. 회원가입 구현



```python
#account/urls.py

from django.urls import path
from . import views

app_name = 'account' # 로그인 기능은 앱에 중복될 일이 없으니 app_name 없어도 됨
urlpatterns = [
        path('signup/', views.signup, name='signup'),
]
```

회원 가입 구현

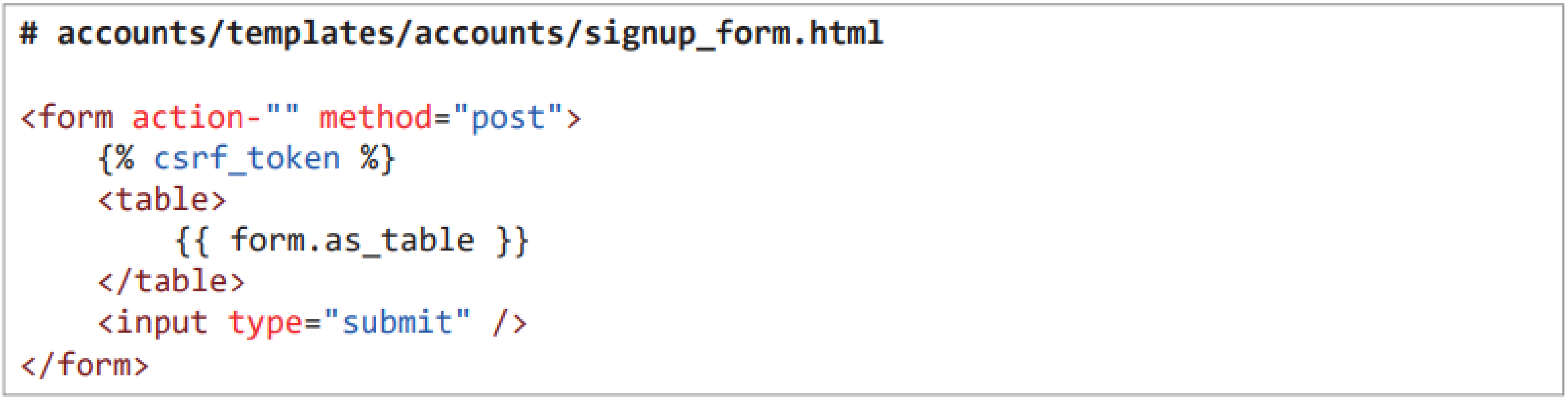
2. 회원가입 구현

```python
# accounts/views.py

from django.shortcuts import render, redirect
from django.contrib.auth.forms import UserCreationForm
from django.conf import settings

def signup(request):
    if request.method == 'POST':
        form = UserCreationForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect(settings.LOGIN_URL)
    else:
        form = UserCreationForm()

    return render(request, 'accounts/signup_form.html',{'form':form})
```

https://github.com/django/django/blob/main/django/conf/global_settings.py#LC504

회원 가입 구현

2. 회원가입 구현



```
# accounts/templates/accounts/signup_form.html

<form action-"" method="post">
    {% csrf_token %}
    <table>
        {{ form.as_table }}
    </table>
    <input type="submit" />
</form>
```

UserCreationForm 커스텀(1)

2. 회원가입 구현

```
#accounts/forms.py

from django.contrib.auth.forms import UserCreationForm

class SignupForm(UserCreationForm):
    class Meta(UserCreationForm.Meta):
          fields= UserCreationForm.Meta.fields +('email',)
```

```
#accounts/views.py

from .forms import SignupForm

def signup(request):
    if request.method == 'POST':
        form = SignupForm(request.POST)
        if form.is_valid():
                form.save()
                return redirect(settings.LOGIN_URL)
    else:
        form = SignupForm()
    return render(request, 'accounts/signup_form.html',{'form':form})
```

UserCreationForm 커스텀(2)

2. 회원가입 구현

```
# accounts/form.py
from django.contrib.auth.forms import UserCreationForm
from django import forms
from accounts.models import Profile

class SignupForm(UserCreationForm):
  phone_number = forms.CharField(label="전화", max_length=20)
  address = forms.CharField(label="주소", max_length=50)
  class Meta(UserCreationForm.Meta):
    fields = UserCreationForm.Meta.fields + ('email',)
  def save(self, commit=True):
    user = super().save()
    profile = Profile(user=user,
                      phone_number=self.cleaned_data['phone_number'],
                      address=self.cleaned_data['address'])
    profile.save()
    return profile
```

Email 보내는 library

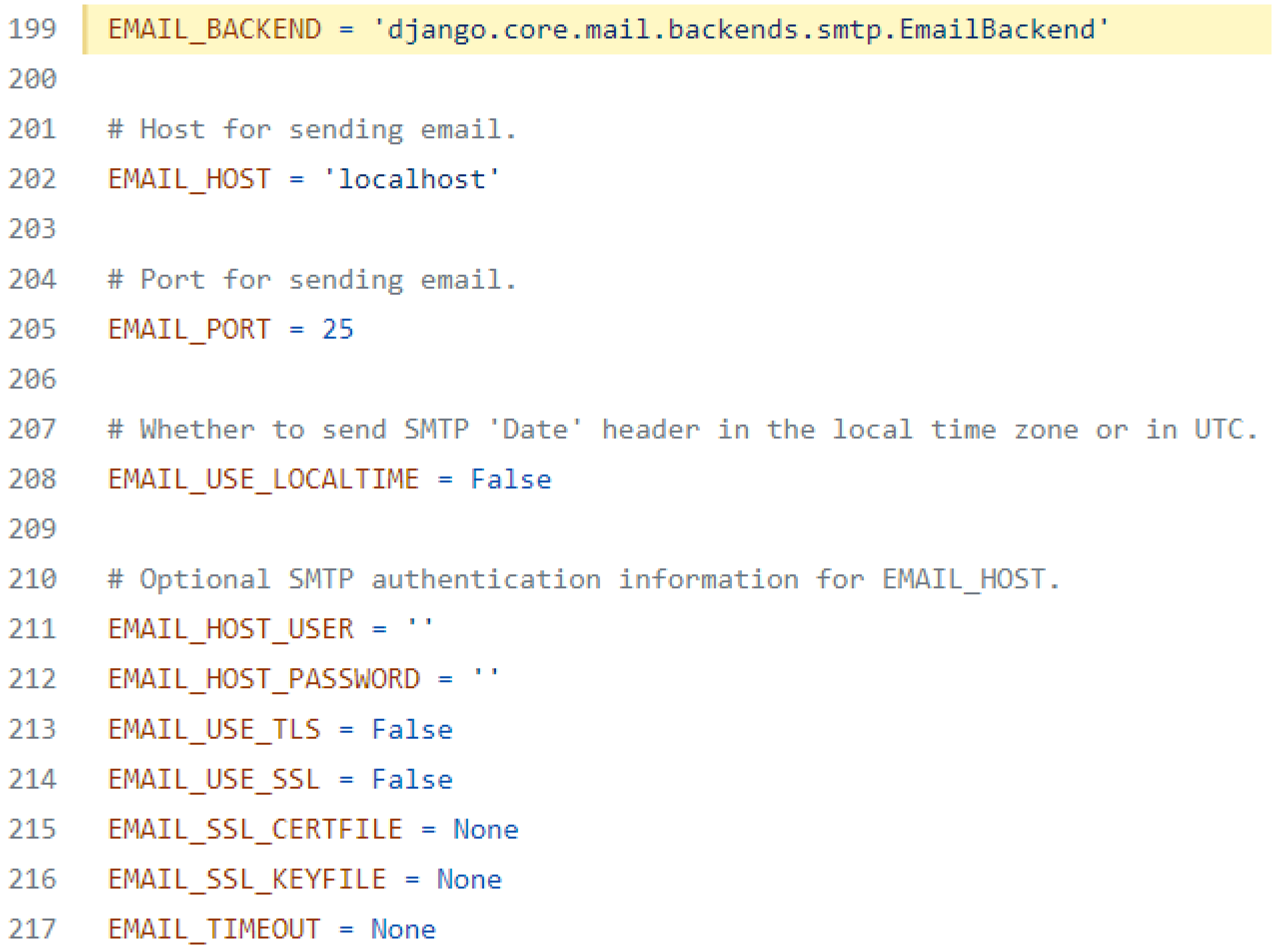
2. 회원가입 구현

- **Djagno Email library** ⇦
  - https://docs.djangoproject.com/ko/3.2/topics/email
- Django 3rd party Email library
- Python 3rd party Email library
  - https://github.com/vinta/awesome-python#email

Django Email

2. 회원가입 구현

- https://github.com/django/django/blob/main/django/conf/global_settings.py#L199



```python
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'

# Host for sending email.
EMAIL_HOST = 'localhost'

# Port for sending email.
EMAIL_PORT = 25

# Whether to send SMTP 'Date' header in the local time zone or in UTC.
EMAIL_USE_LOCALTIME = False

# Optional SMTP authentication information for EMAIL_HOST.
EMAIL_HOST_USER = ''
EMAIL_HOST_PASSWORD = ''
EMAIL_USE_TLS = False
EMAIL_USE_SSL = False
EMAIL_SSL_CERTFILE = None
EMAIL_SSL_KEYFILE = None
EMAIL_TIMEOUT = None
```

https://docs.djangoproject.com/en/3.2/topics/email/#s-smtp-backend

Django Email

2. 회원가입 구현

- smtp.EmailBackend – Naver SMTP 연동

```
# settings.py

# EMAIL_BACKEND = 'django.core.mail.backends.console.EmailBackend'
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.naver.com'
EMAIL_PORT = 465
EMAIL_HOST_USER = 'naver-id@naver.com'
EMAIL_HOST_PASSWORD = '비밀번호'
EMAIL_USE_SSL = True
DEFAULT_FROM_EMAIL = EMAIL_HOST_USER


# 네이버 메일에서 POP3/STMP 사용으로 설정
```

Django Email

2. 회원가입 구현

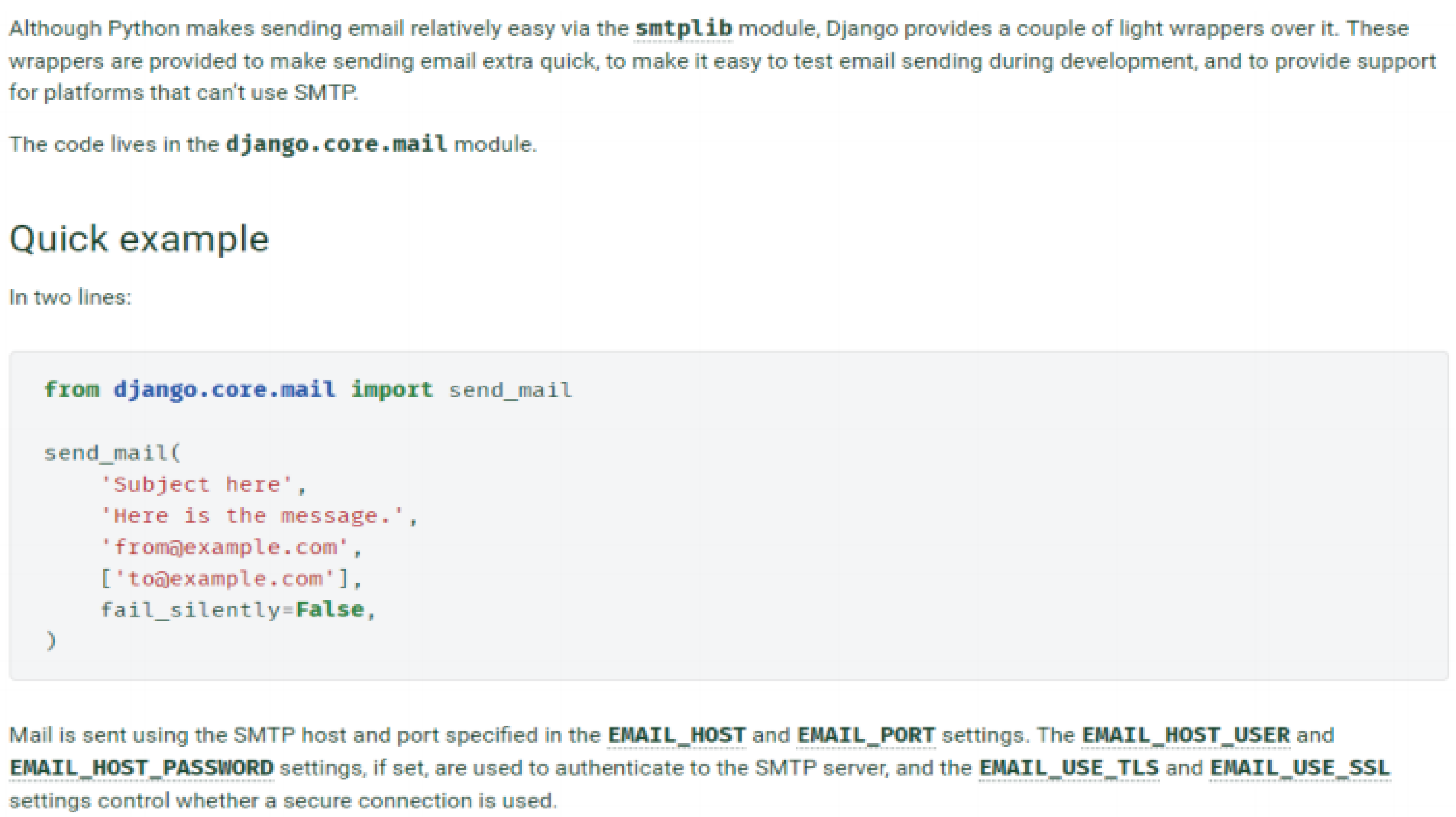
- smtp.EmailBackend – Google SMTP 연동

```
# settings.py

# EMAIL_BACKEND = 'django.core.mail.backends.console.EmailBackend'
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_PORT = 465
EMAIL_HOST_USER = 'gmail-id@gmail.com'
EMAIL_HOST_PASSWORD = '비밀번호'
EMAIL_USE_SSL = True
DEFAULT_FROM_EMAIL = EMAIL_HOST_USER


# 구글 보안에서 보안등급을 낮게 설정
```

Django Email

2. 회원가입 구현

- Sending email



Although Python makes sending email relatively easy via the **smtplib** module, Django provides a couple of light wrappers over it. These wrappers are provided to make sending email extra quick, to make it easy to test email sending during development, and to provide support for platforms that can't use SMTP.

The code lives in the **django.core.mail** module.

Quick example

In two lines:

```
from django.core.mail import send_mail

send_mail(
    'Subject here',
    'Here is the message.',
    'from@example.com',
    ['to@example.com'],
    fail_silently=False,
)
```

Mail is sent using the SMTP host and port specified in the **EMAIL_HOST** and **EMAIL_PORT** settings. The **EMAIL_HOST_USER** and **EMAIL_HOST_PASSWORD** settings, if set, are used to authenticate to the SMTP server, and the **EMAIL_USE_TLS** and **EMAIL_USE_SSL** settings control whether a secure connection is used.

- https://github.com/django/django/blob/main/django/core/mail/__init__.py

Email 보내기

2. 회원가입 구현

- 이벤트 처리 : 회원가입 완료 후 가입인사 이메일 전송
  - Signal = 이벤트

```
# accounts/models.py

from django.db.models.signals import post_save
from django.conf.global_settings import AUTH_USER_MODEL

def on_send_mail(sender, **kwargs):
    if kwargs['created']:
        user = kwargs['instance']

post_save.connect(on_send_mail, sender=AUTH_USER_MODEL)
```

참조 : **https://docs.djangoproject.com/en/3.2/ref/signals/**
https://github.com/django/django/blob/main/django/db/models/signals.py
https://github.com/django/django/blob/main/django/dispatch/dispatcher.py#LC22

Django Email

2. 회원가입 구현

● 이메일 보내기

```
# accounts/models.py
from django.conf import settings
from django.core.mail import send_mail
from django.db import models
from django.db.models.signals import post_save

class Profile(models.Model):
  user = models.OneToOneField(settings.AUTH_USER_MODEL, on_delete=models.CASCADE)
  phone_number = models.CharField(max_length=20)
  address = models.CharField(max_length=50)

def on_send_mail(sender, **kwargs):
  if kwargs['created']:
        user = kwargs['instance']
        send_mail( '가입인사’, '가입을 환영합니다’,
                  '보내는naver-id@naver.com',
                 [user.email],
        fail_silently=False)

post_save.connect(on_send_mail, sender=settings.AUTH_USER_MODEL)
# post_save.connect(on_send_mail, sender=Profile)
```

Django Email

2. 회원가입 구현

- smtp.EmailBackend – Google SMTP 연동

```
# settings.py
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_PORT = 465
EMAIL_HOST_USER = config(‘EMAIL_HOST_USER’)
EMAIL_HOST_PASSWORD = config('EMAIL_HOST_PASSWORD‘)
EMAIL_USE_SSL = True
DEFAULT_FROM_EMAIL = EMAIL_HOST_USER

# manage.py와 같은 폴더에 .env
EMAIL_HOST_USER=‘naver-id@naver.com'
EMAIL_HOST_PASSWORD=‘비밀번호‘
```

3. 로그인/아웃 구현

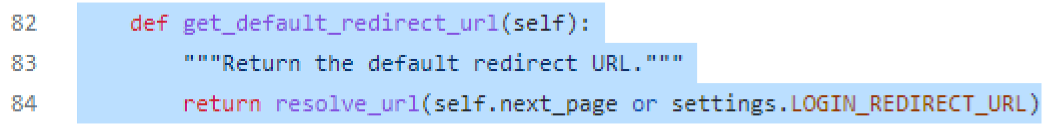
로그인/로그아웃 구현

3. 로그인/아웃 구현

https://github.com/django/django/blob/main/django/contrib/auth/views.py

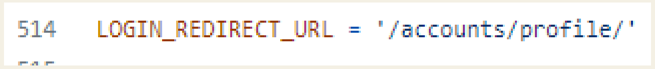
```python
# accounts/urls.py

from django.conf import settings
from django.urls import path
from django.contrib.auth import views as auth_views
from . import views

app_name = 'accounts'
urlpatterns = [
    path('signup/', views.signup, name='signup'),
    path('profile/', views.profile, name='profile')
    path('login/', auth_views.LoginView.as_view(
            template_name='accounts/login_form.html'), name='login'),
    path('logout/', auth_views.LogoutView.as_view(
                    next_page=settings.LOGIN_URL), name='logout'),
]
```

로그인 성공시 settings.LOGIN_REDIRECT_URL로



```python
    def get_default_redirect_url(self):
        """Return the default redirect URL."""
        return resolve_url(self.next_page or settings.LOGIN_REDIRECT_URL)
```

https://github.com/django/django/blob/main/django/conf/global_settings.py#LC514



```python
514    LOGIN_REDIRECT_URL = '/accounts/profile/'
```

로그인/로그아웃 구현

3. 로그인/아웃 구현

```
# accounts/views.py

def profile(request):
        return render(request, 'accounts/profile.html')
```

```
# accounts/templates/accounts/profile.html

<h1> {{user}}'s Profile </h1>

<ul>
    <li> name : {{user}}</li>
    <li> email : {{user.email}}</li>
    <li> is_staff : {{user.is_staff}}</li>
    <li> is_superuser : {{user.is_superuser}}</li>
    <li>{{user.date_joined}}</li>
</ul>
```

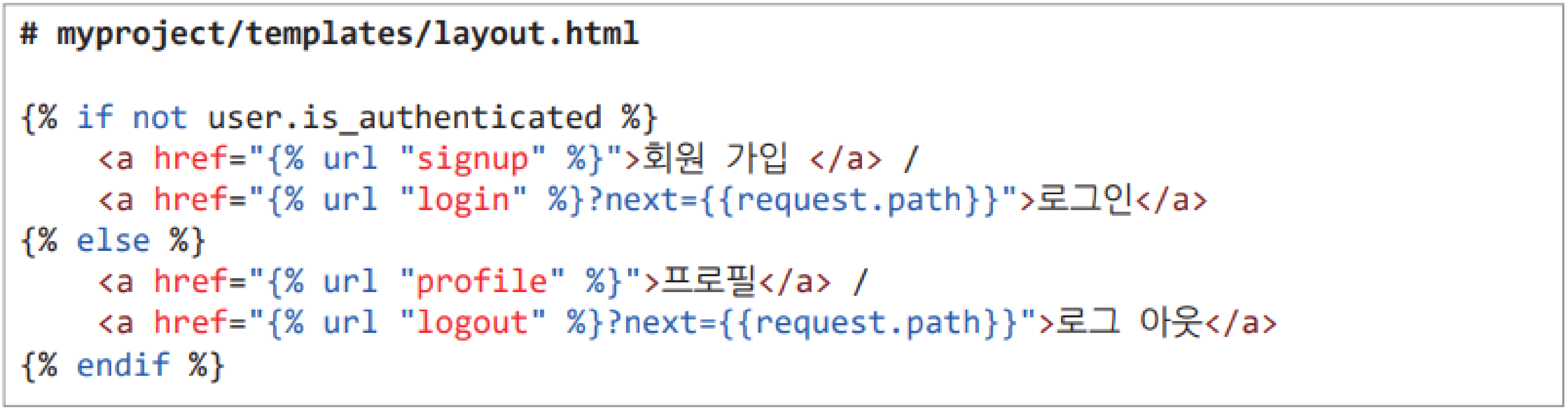
로그인/로그아웃 구현

3. 로그인/아웃 구현

```
# accounts/templates/accounts/login_form.html

<form action="" method="post">
    {% csrf_token %}
    <table>
        {{ form.as_table}}
    </table>
    <input type="submit" />
</form>
```

로그인/로그아웃 구현

3. 로그인/아웃 구현



```
# myproject/templates/layout.html

{% if not user.is_authenticated %}
    <a href="{% url "signup" %}">회원 가입 </a> /
    <a href="{% url "login" %}?next={{request.path}}">로그인</a>
{% else %}
    <a href="{% url "profile" %}">프로필</a> /
    <a href="{% url "logout" %}?next={{request.path}}">로그 아웃</a>
{% endif %}
```

{{request.path}} : 현재 페이지 요청경로