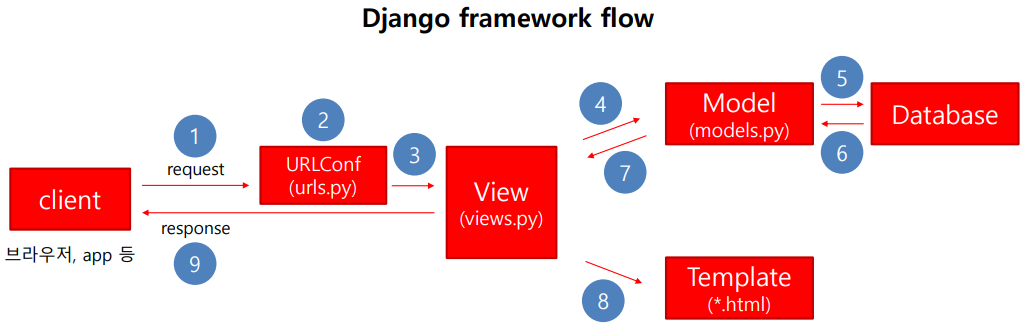
**[ 3 ] Django 프로젝트 생성과 데이터**

* Django Framework 호름도
* 프로젝트 및 애플리케이션 생성
* 프로젝트 설정 변경(settings.py)
* 데이터베이스 만들기(models.py) – 모델에서 데이터 검색, 수정, 삭제
* 관리자 계정 생성 및 웹서버 실행
* 관리자 페이지 접속

1. Django Framework flow



1. 프로젝트 및 애플리케이션 생성

Django-admin startproject ch02

move ch02 ch02\_students

cd ch02\_students

python manage.py startapp students

1. 소스

ch02/settings.py에 앱 등록 추가

INSTALLED\_APPS = [  
 **'django.contrib.admin'**,

**'django.contrib.auth'**,

**'django.contrib.contenttypes'**,

**'django.contrib.sessions'**,

**'django.contrib.messages'**,

**'django.contrib.staticfiles'**,

**'students.apps.StudentsConfig'**,*## 입등록 추가*

]

Anaconda prompt의 ch02\_student폴더에서

* python manage.py migrate 실행 ; 사용자 및 그룹 테이블을 생성
* python manage.py createsuperuser 실행 ; 관리자 계정 생성
* python manage.py runserver 8000 ; 포트를 지정하지 않고 서버 구동 시킬 수도 있음(default는 8000)
* http://localhost:8000/admin 을 브라우저로 접근하면 기본적으로 제공해주는 관리자 모드 접근 가능

ㅡstudents/models.py

**from** django.db **import** models  
  
**class** Student(models.Model):  
 s\_name = models.CharField(max\_length=1000)  
 s\_major = models.CharField(max\_length=100)  
 s\_age = models.IntegerField(default=0)  
 s\_grade = models.IntegerField(default=0)  
 s\_gender = models.CharField(max\_length=30)  
  
 **def** \_\_str\_\_(self):  
 **return** self.s\_name

student/admin.py

**from** django.contrib **import** admin  
**from** students.models **import** Student  
  
admin.site.register(Student) *# model 등록*

Anaconda prompt의 ch02\_student폴더에서

* python manage.py makemigrations ; 변경사항이 있는지 검색
* python manage.py migrate
* python manage.py shell ; 장고 shell 몯 실행
* 레코드 추가(create) – 데이터 생성은 다음의 과정 실행

python manage.py shell

from students.models import Student

qs = Student(s\_name=’Hone’, s\_major=’computer’, s\_age=21, s\_grade=2, s\_gender=’M’)

qs.save()

qs = Student(s\_name='kim', s\_major='dance', s\_age=20, s\_grade=1, s\_gender='F')

qs.save()

qs = Student(s\_name='Yi', s\_major='MIS', s\_age=40, s\_grade=1, s\_gender='F')

qs.save()

* 레코드 읽기(read) – 데이터 검색

Student.objects.all() # 레코드 전체 읽기

qs = Student.objects.all()

print(qs)

qs[0]

qs[0].s\_gender

qs = Student.objects.get(s\_name='Hong') # 조건에 맞는 하나 읽기

qs

print(type(qs))

* 필터로 읽기(filter read)

qs = Student.objects.filter(s\_age\_\_lt=30)

\_\_lt : ~ 보다 작다

\_\_lte ; ~보다 작거나 같다

\_\_gt ; ~보다 크다

\_\_gte ; ~보다 크거나 같다

\_\_isnull ; null인 자료 검색

\_\_contains ; 특정 문자열을 포함하는 자료 검색데이터 정렬

qs = Student.objects.order\_by('s\_age') # s\_age필터 기준으로 오름차순 정렬하여 가져옴

qs = Student.objects.order\_by('-s\_age') # 내림차순 정렬하여 가져옴

* 데이터 수정

qs = Student.objects.get(s\_name='Hong')

qs.s\_age=30

qs.save()

* 데이터 삭제

qs = Student.objects.get(s\_name='Hong')

qs.delete()