Python. Подготовка к собеседованию

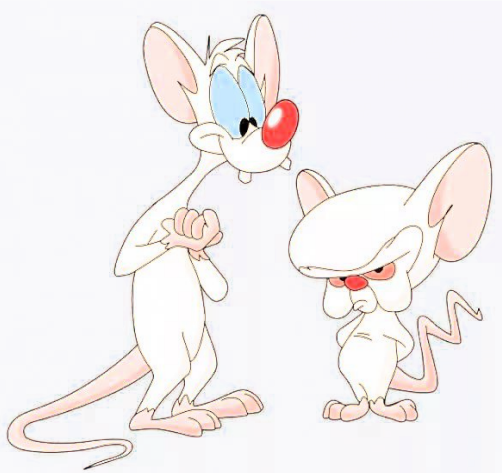
Урок 4

Фреймворк Django

Основные понятия ORM, структура и особенности проектирования

Цели урока

Повторить пошаговый процесс разработки приложений на Python/Django.

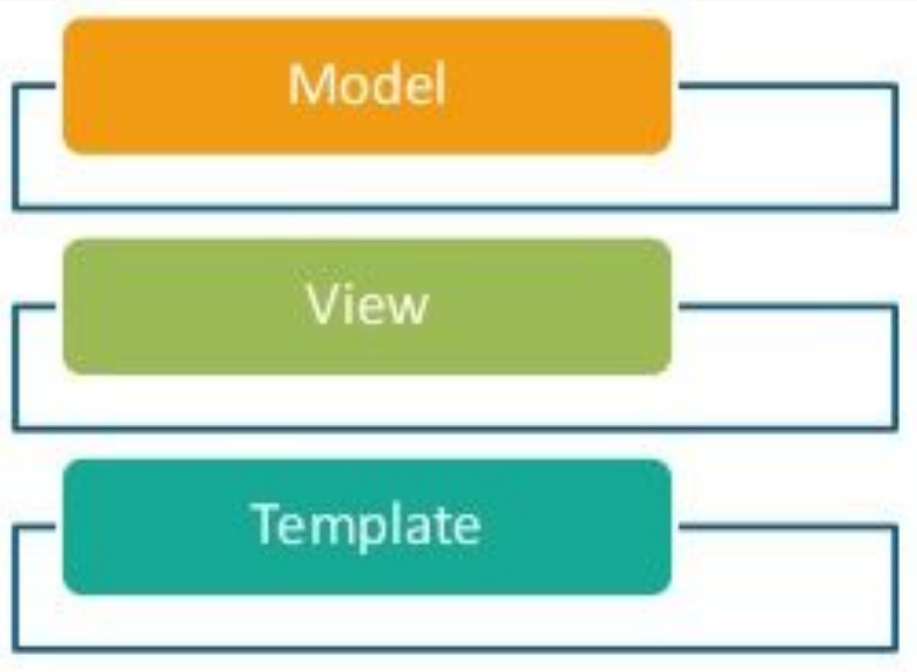


- Преимущества Django, шаблон проектирования;
- Создание проекта и приложения, настройка файла settings.py;
- Запуск сервера разработки, создание моделей, подключение админки;
- Создание url-адресов, привязка контроллеров, проектирование шаблонов.
- Логика работы приложения на Python/Django, развертывание проекта.

Django и его преимущества

- Минимальные затраты времени на изучение.
- Высокий уровень функциональности.
- Широкий набор встроенных функций.
- Безопасность приложений.

Шаблон проектирования Django

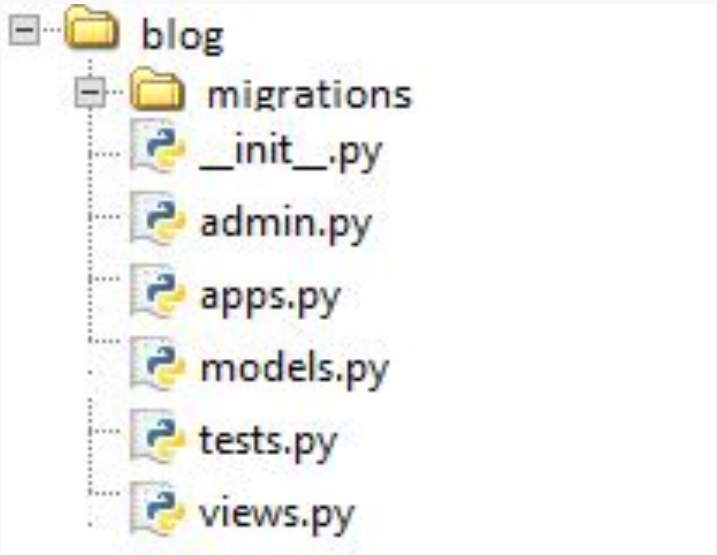


Модель-Представление-Шаблон

Создание проекта Django

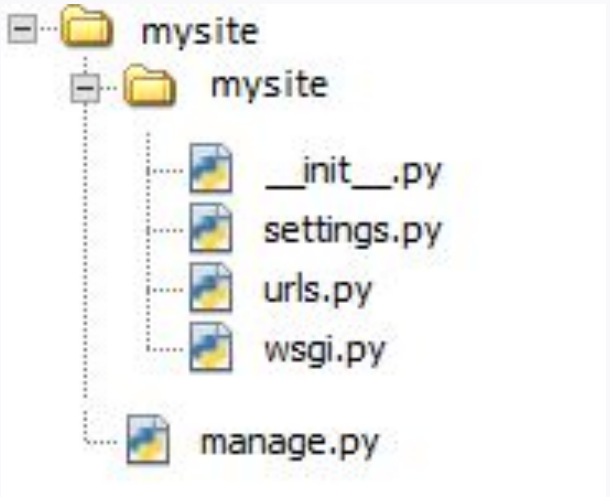
django-admin.py startproject имя_проекта

Создание приложения Django



python mange.py startapp имя_приложения

Файл settings.py и его настройка

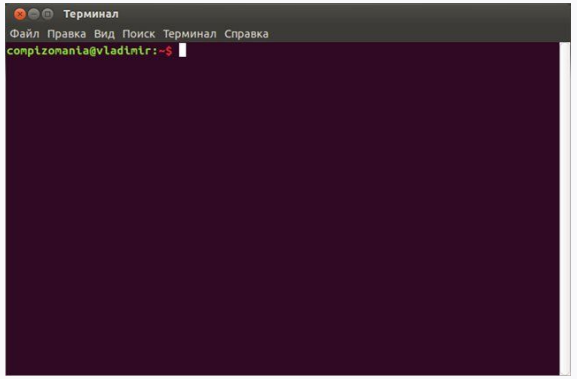



```
DATABASES = {
  'default': {
      'ENGINE': 'django.db.backends.sqlite3',
      'NAME': '/home/dotcloud/ramen.db',
      'USER': '',
      'PASSWORD': '',
      'HOST': '',
      'PORT': '',
  }
}
```

```
INSTALLED_APPS = (
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'index',
)
```

Запуск сервера разработки Django

python mange.py runserver



127.0.0.1:8000 или localhost:8000

Модели в Django

Models

A model is the single, definitive source of information about your data. It contains the essential fields and behaviors of the data you're storing. Generally, each model maps to a single database table.

The basics:

- Each model is a Python class that subclasses **`django.db.models.Model`**.
- Each attribute of the model represents a database field.
- With all of this, Django gives you an automatically-generated database-access API; see Making queries.

Quick example

This example model defines a **`Person`**, which has a **`first_name`** and **`last_name`**:

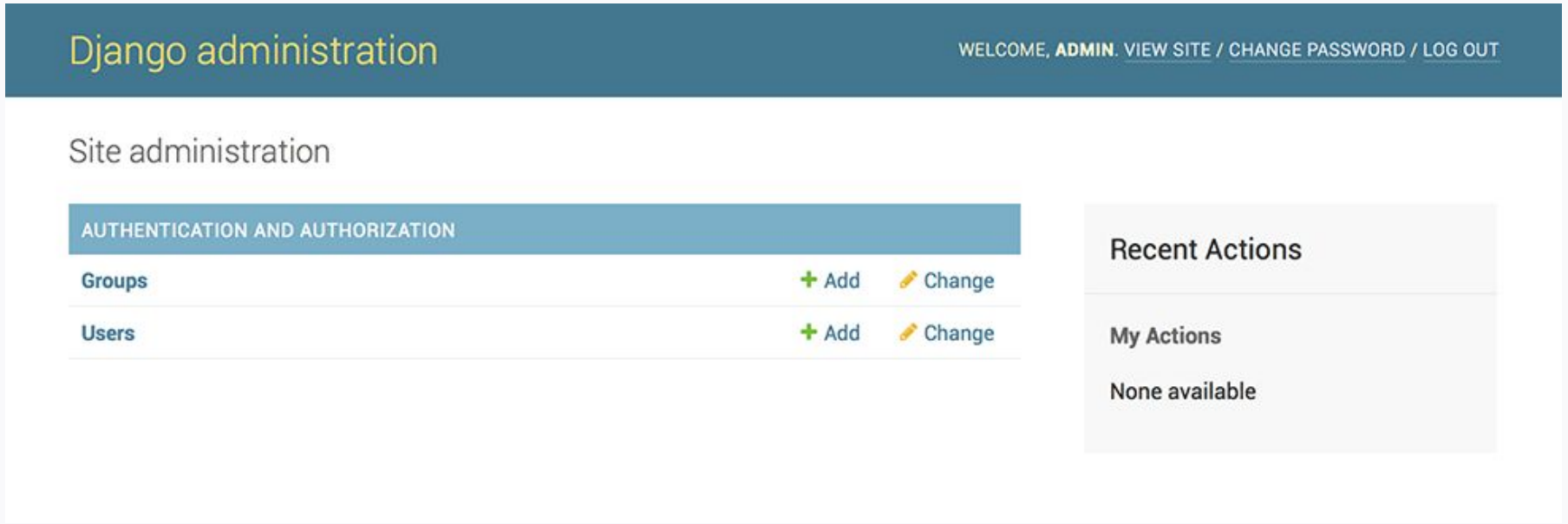
```
from django.db import models

class Person(models.Model):
    first_name = models.CharField(max_length=30)
    last_name = models.CharField(max_length=30)
```

python mange.py makemigrations имя_приложения

python mange.py migrate имя_приложения

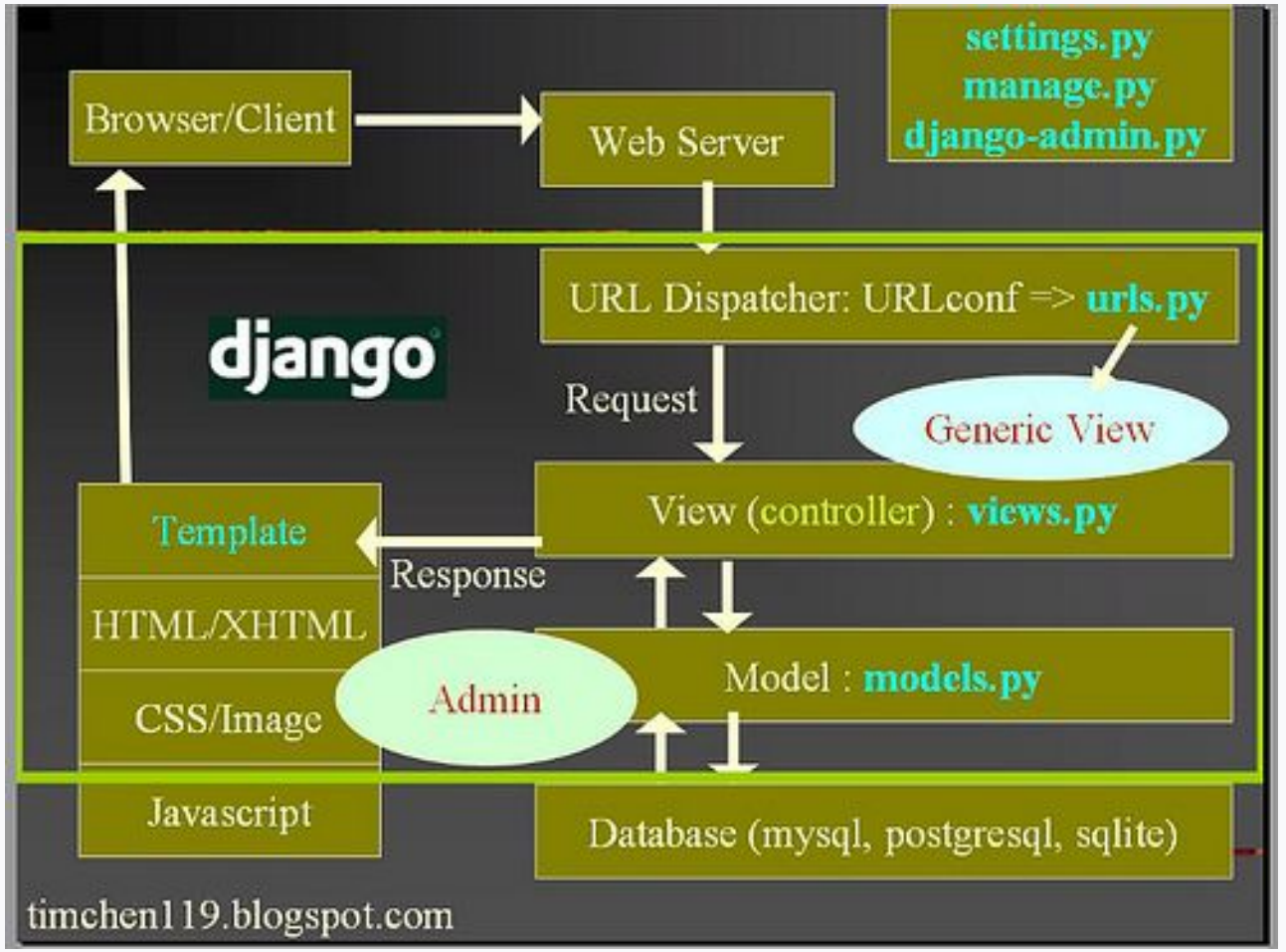
Админка в Django



python mange.py createsuperuser

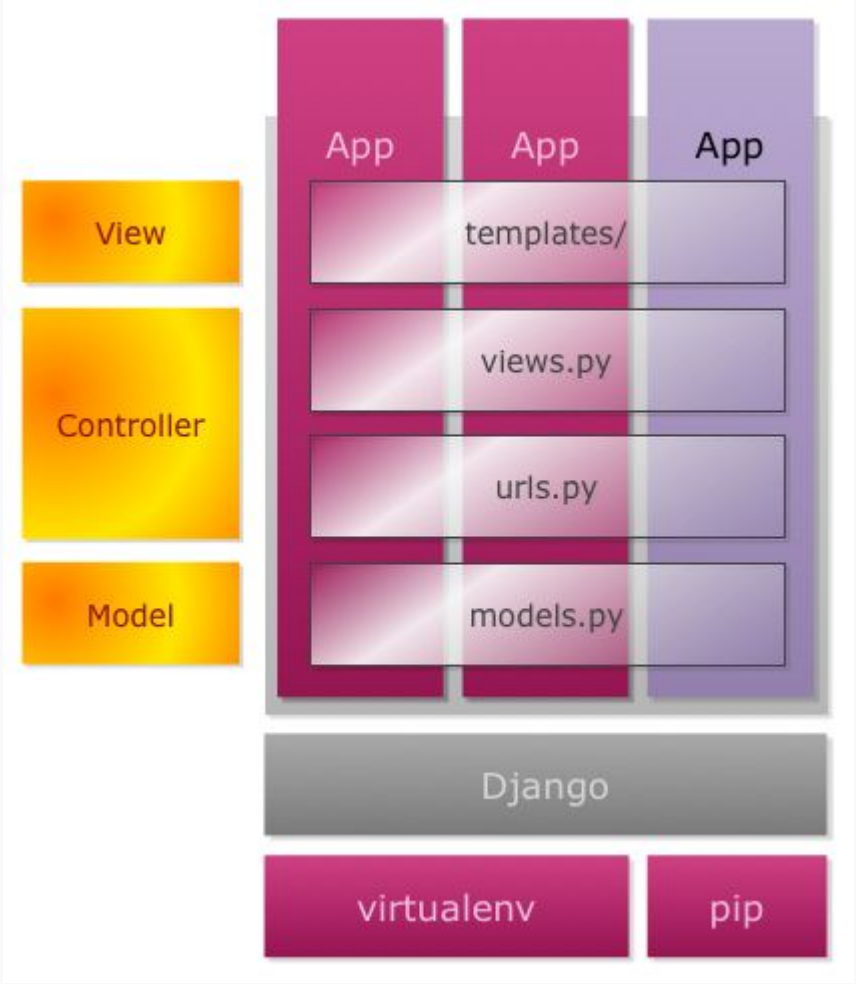
admin.site.register(имя_модели)

Привязка URL-адресов



Контроллеры в Django

- Функции-контроллеры
- Классы-контроллеры



Шаблоны в Django

Использование тегов шаблонов

{% имя_тега %}

Использование контекста

{{ имя_переменной_контекста.имя_параметра }}

Подключение статики

{% load staticfiles %}

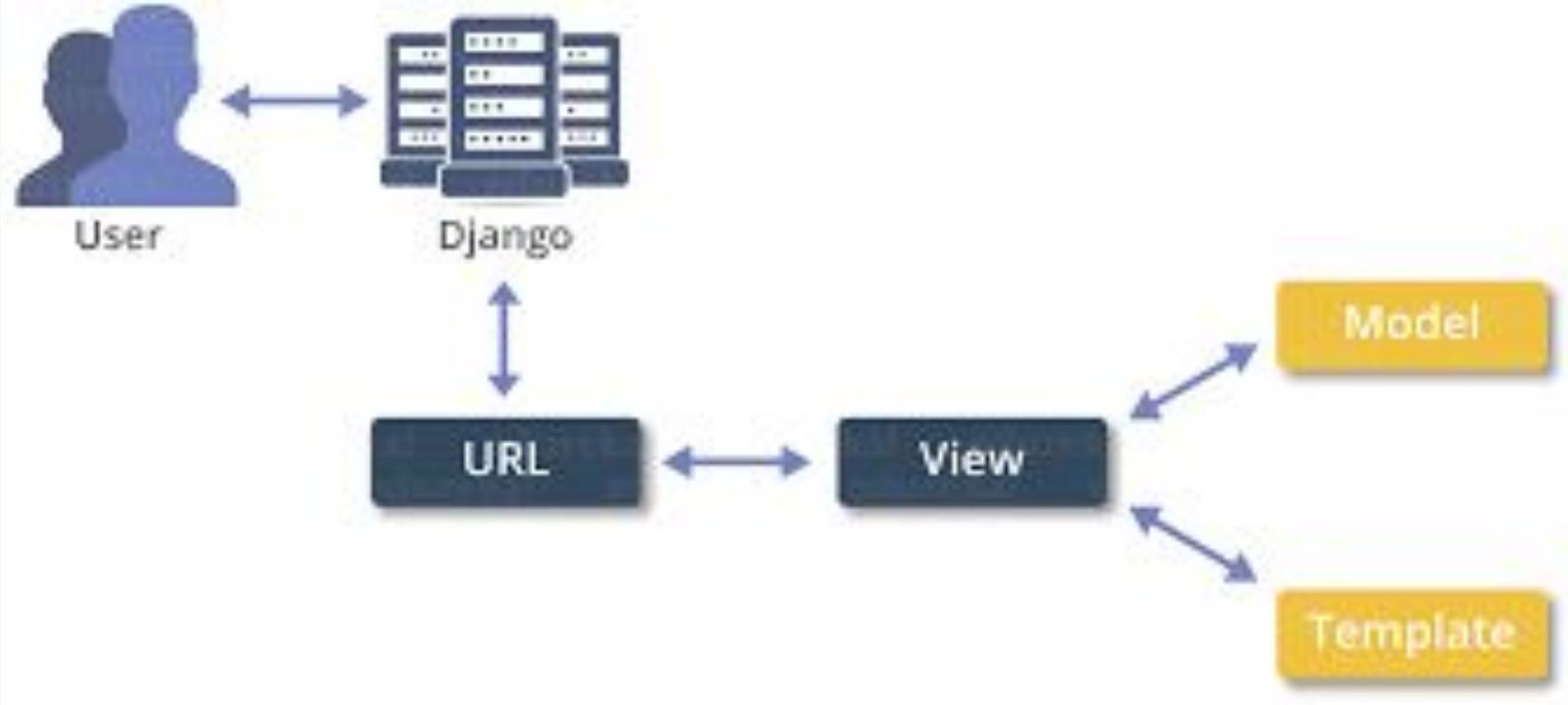
<link rel=”stylesheet” href=”{% static ‘css/style.css’%}”>

Наследование шаблонов

{% block <имя_блока> %}:

{% endblock %}

Логика работы Django-приложения



Развертывание Django-проекта

Практическое задание

Практическое задание

Реализовать Python/Django-проект “Каталог товаров”, который должен состоять из двух страниц:

1. Главная страница со списком товаров и кнопкой открытия формы добавления товара.
2. Страница с формой добавления товара.

Приложение должно функционировать в синхронном режиме: пользователь нажимает кнопку «Добавить товар», переходит на предназначенную для этого страницу, указывает в форме необходимые данные и нажимает «Сохранить». После этого он перенаправляется на главную страницу, где в табличной форме отображается список всех товаров. Проект должен быть привязан к базе данных **SQLite3** (БД по умолчанию) и иметь минимальную сложность стилевого оформления.

Дополнительные материалы

1. https://tproger.ru/translations/create-your-first-django-app/.
2. https://habr.com/post/226419/.
3. https://proglib.io/p/nginx/.
4. https://www.ibm.com/developerworks/ru/library/os-django/index.html.
5. https://habr.com/post/61918/.
6. https://adw0rd.com/2016/02/02/django-nginx-uwsgi/.
7. https://habr.com/post/267721/.