Django周末速成班

文泰来 老师

扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: ninechapter
知乎专栏：http://zhuanlan.zhihu.com/jiuzhang
微博: http://www.weibo.com/ninechapter
官网: www.jiuzhang.com

1-1

第一节

我是班主任棋棋，
加我领取课程福利哦

讲师：文泰来

加班主任，进班级答疑群
快速获取面试资料/课程福利

关注公众号，了解大厂资讯

开课目的

- 同学A：课(tuo)业(yan)多(zheng)，大多数网课的周期又太长，很难坚持下去
- 同学B：下周就要校招了，我的简历还没什么货，从哪能有个合适的项目让我写上去就好了
- 同学C：我想自己写个网站，最好learning curve比较短，不知道有没有适合我的stack
- 同学D：我发现一百个人做网站有一百种做法，到底哪种才是最优解? 要是有个人能跟我统一说明就好了

开课目的

Django周末速成班

- 硬核学习，2天，16个小时，帮助大家在一个周末成长为合格的全栈工程师(偏后端)
- 从零开始，手把手带大家配置环境(随堂采用MacOS演示)，搭建前后端，做出一个完整的项目
- 和Industry结合，采用业界广泛运用的开发方法和工具，帮助大家理解开发当中的各种Best practice

Backend developer's toolkit

- Version control: Git
- IDE: VSCode
- Package management: HomeBrew, Pip
- Python Virtual Env: Pipenv
- Static content storage: AWS S3
- Website deployment: Heroku

课程需求

- 编程语言: Python (入门)
- 一台电脑(prefer mac)

- 如果你想要提前准备，可以申请以下账号：

- Heroku
- Github
- AWS

- 以及

Install VSCode (or you preferred IDE)

- 提前配置Terminal:

https://medium.com/swlh/power-up-your-terminal-using-oh-my-zsh-iterm2-c5a03f73a9fb

老师介绍

文泰来

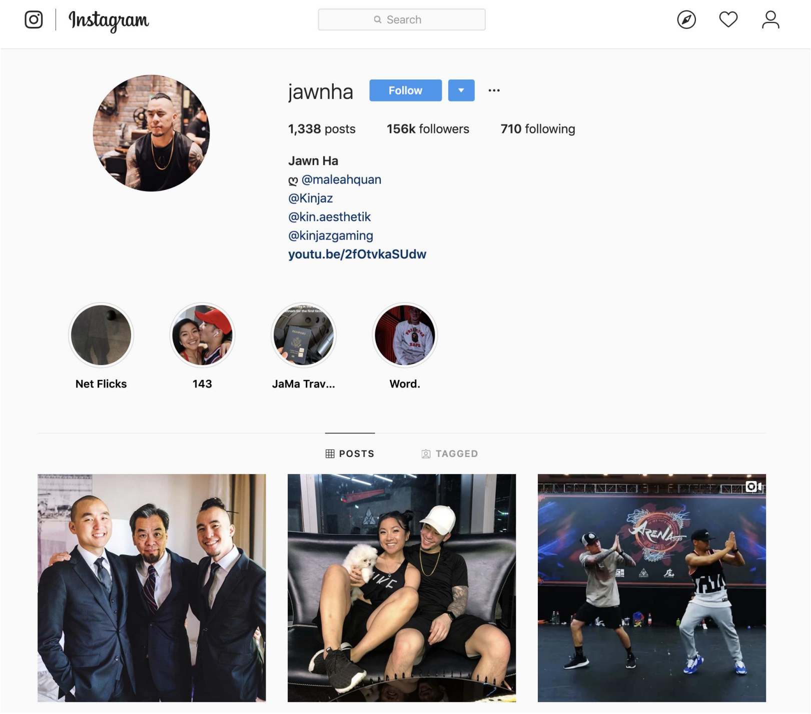
课程：面向对象设计， Django周末速成班

Email: wentailai@jiuzhang.com

Wechat: 九章文泰来

项目介绍

Instagram



jawnha

1,338 posts   156k followers   710 following

Jawn Ha
ღ @maleahquan
@Kinjaz
@kin.aesthetik
@kinjazgaming
youtu.be/2fOtvkaSUdw

项目介绍

九章版Instagram

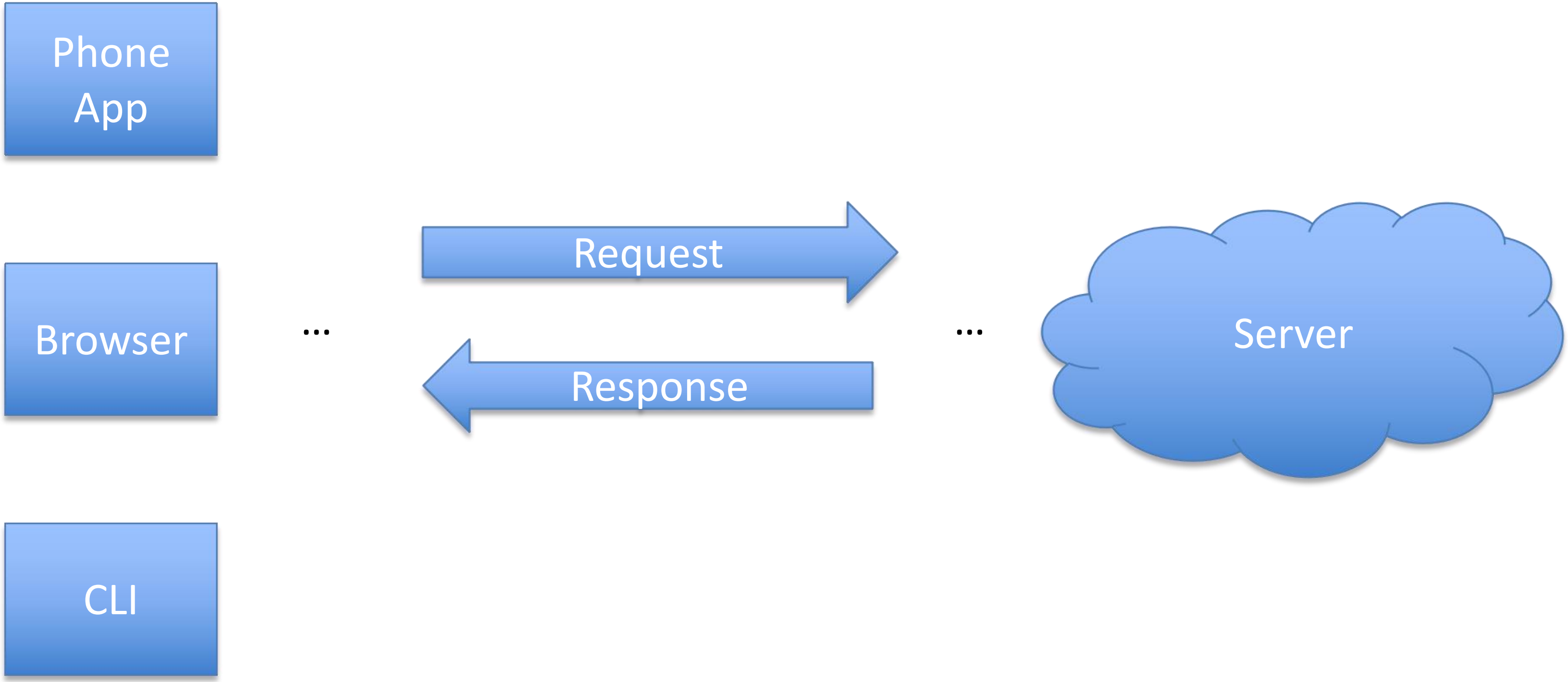
- 用户注册，登录，登出 → User模块
- 显示关注的人发的照片 → Master/Detail模块
- 评论，点赞，取消点赞 → Comments / Likes 模块
- 关注，取关，好友列表 → Friends 模块
- 上传照片 → Content模块

项目介绍

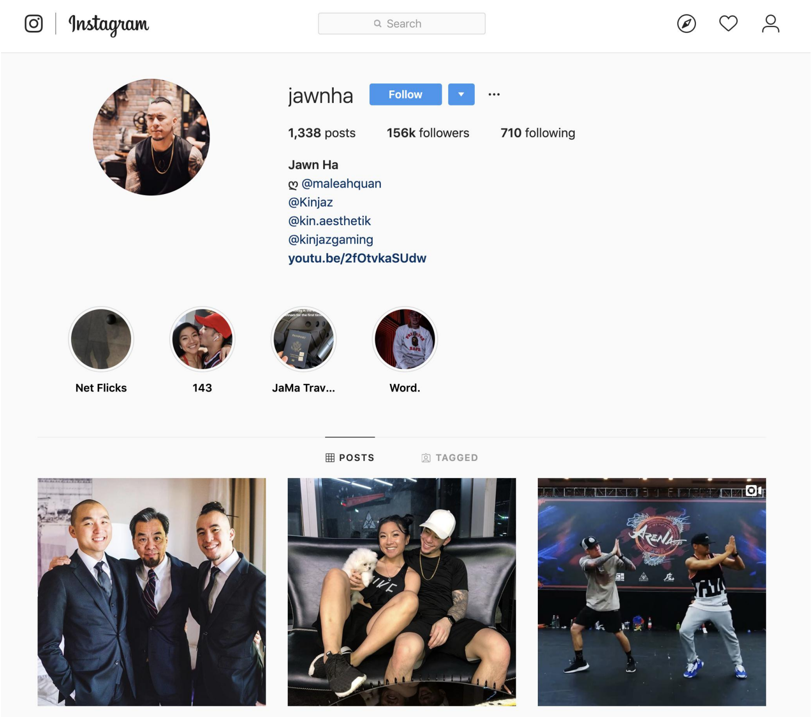
九章版Instagram

Demo: https://shrouded-retreat-79470.herokuapp.com/

Django周末速成班



Week 1 Project:



Instagram

Search

jawnha Follow ▾ ···

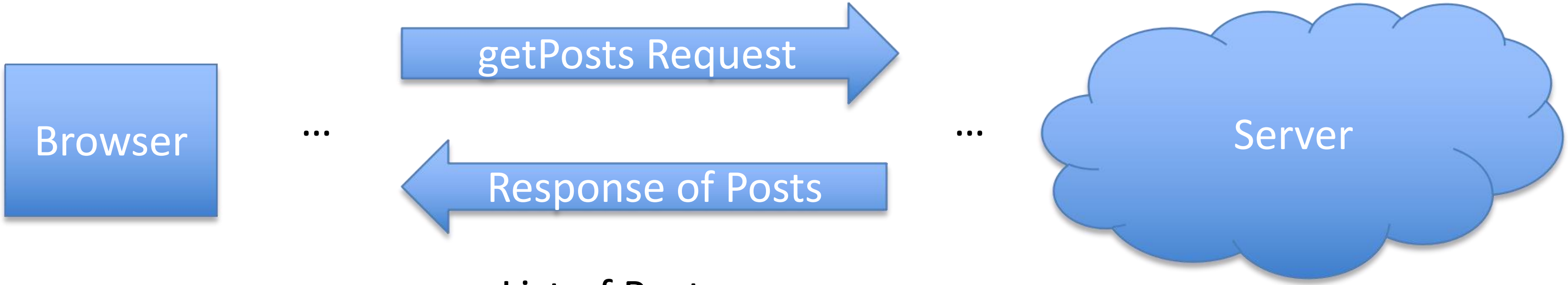
1,338 posts 156k followers 710 following

Jawn Ha
ღ @maleahquan
@Kinjaz
@kin.aesthetik
@kinjazgaming
youtu.be/2fOtvkaSUdw

Net Flicks 143 JaMa Trav... Word.

POSTS TAGGED

Project:



List of Posts
[“p1”, “p2”, “p3”]

Stacks

什么是前端/后端/全栈工程师？

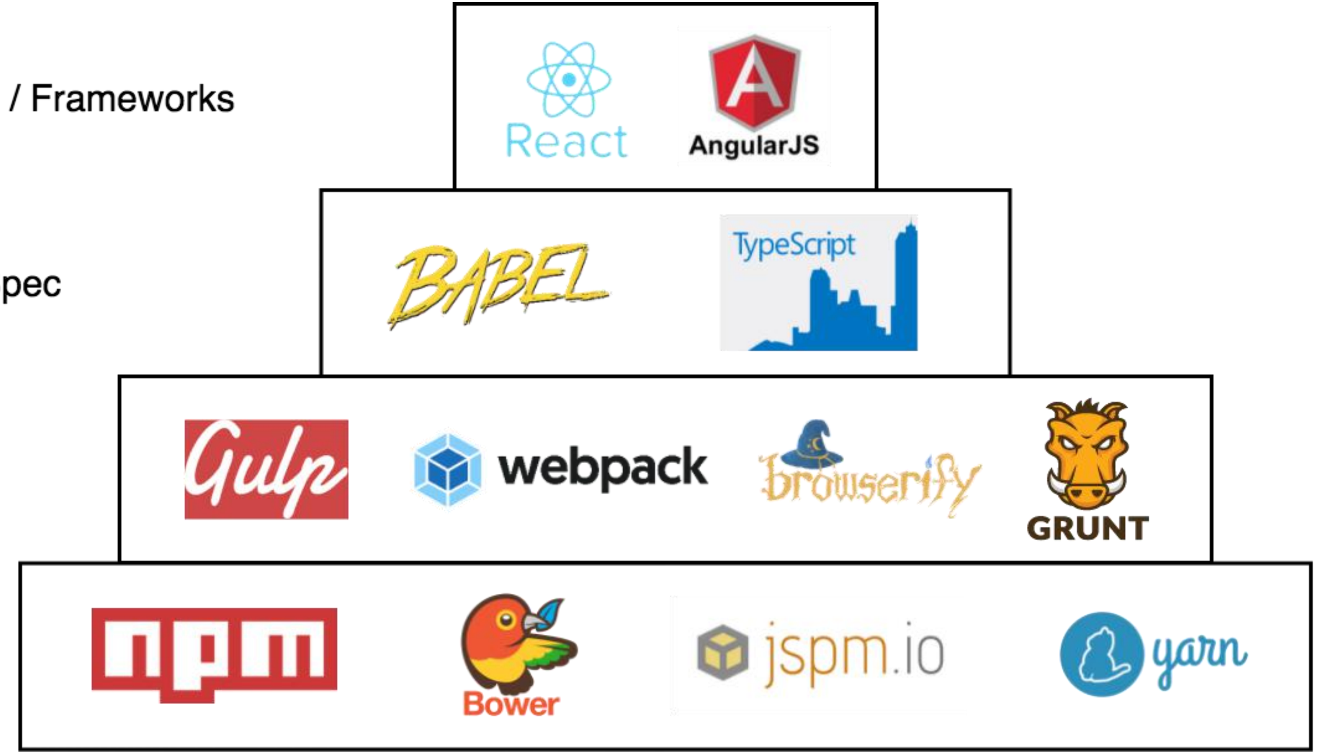
Stacks

UI Libraries / Frameworks

Language Spec

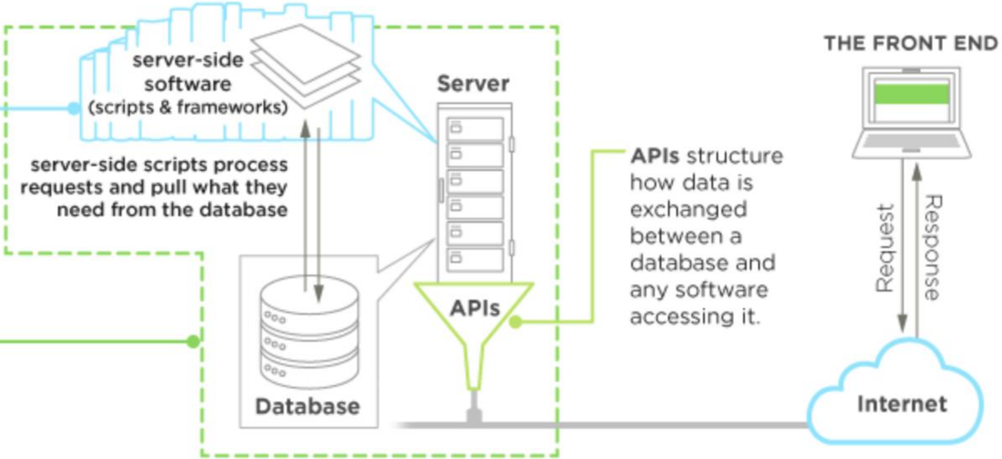
Build Tools

Package Managers



Stacks

BACK-END DEVELOPMENT & FRAMEWORKS IN SERVER SIDE SOFTWARE



FRAMEWORKS are libraries of server-side programming languages that construct the back-end structure of a site.

server-side scripts process requests and pull what they need from the database

APIs structure how data is exchanged between a database and any software accessing it.

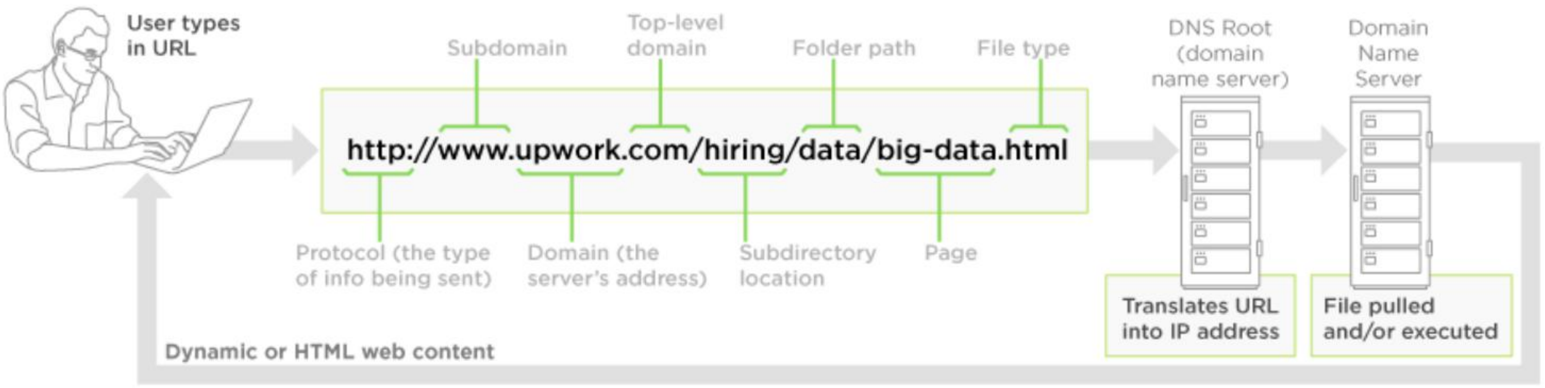
The **"STACK"** comprises the database, server-side framework, server, and operating system (OS).

Stacks

- Server

FROM URL TO YOU: HOW SERVERS WORK

When you type a URL (uniform resource locator) into a browser, it's broken down into parts that make it a very specific address. This is translated into an IP address, telling the server exactly where to look for a file.


User types in URL
Subdomain
Top-level domain
Folder path
File type
http://www.upwork.com/hiring/data/big-data.html
Protocol (the type of info being sent)
Domain (the server's address)
Subdirectory location
Page
DNS Root (domain name server)
Domain Name Server
Translates URL into IP address
File pulled and/or executed
Dynamic or HTML web content

For static content, the HTTP server sends an HTML file back to the browser, which is read and displayed. For dynamic content, a server first executes then returns the file.

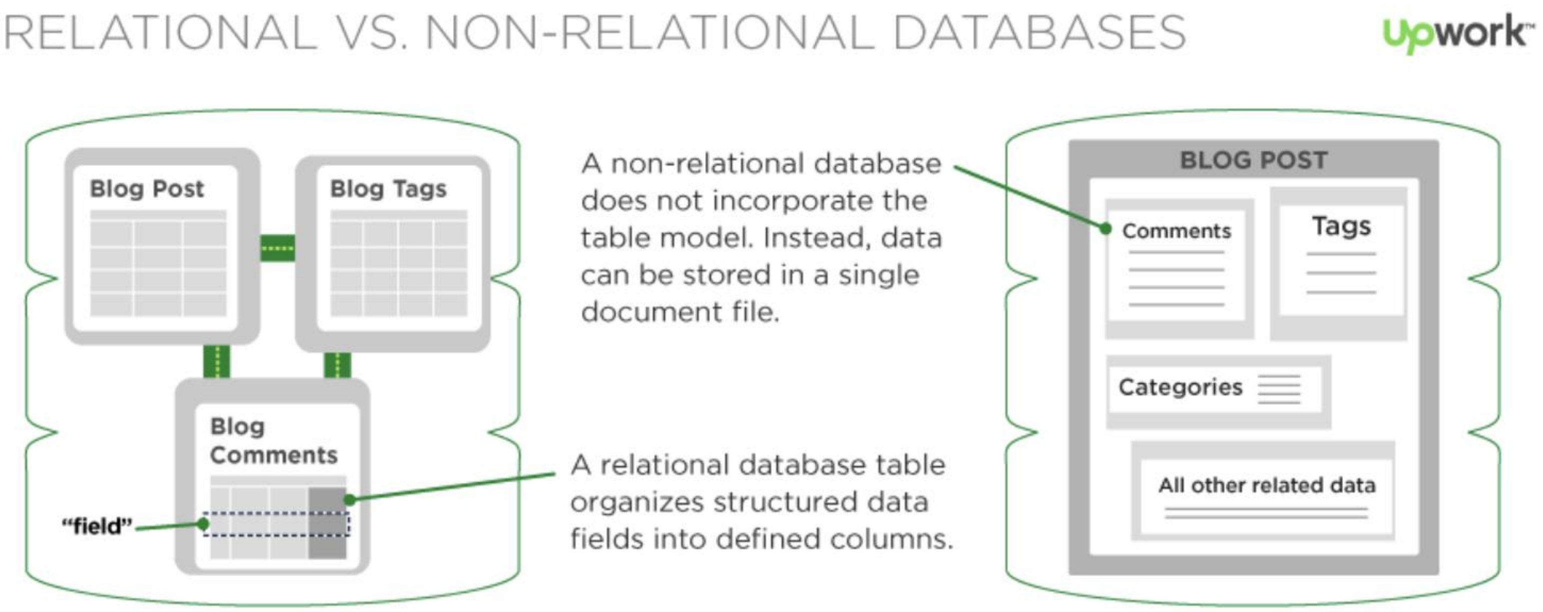
Stacks

- Server

  Apache v.s. NGINX

Stacks

- Database



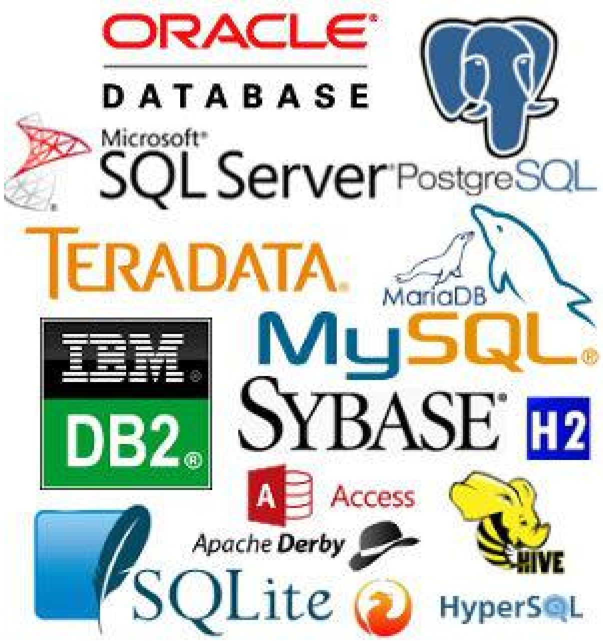
A non-relational database does not incorporate the table model. Instead, data can be stored in a single document file.

A relational database table organizes structured data fields into defined columns.

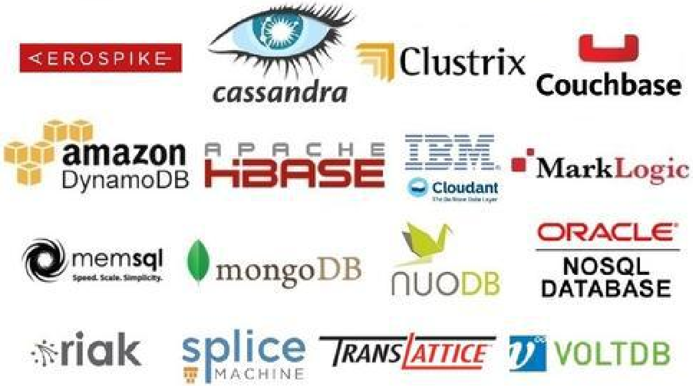
Stacks

- Database

Relational databases

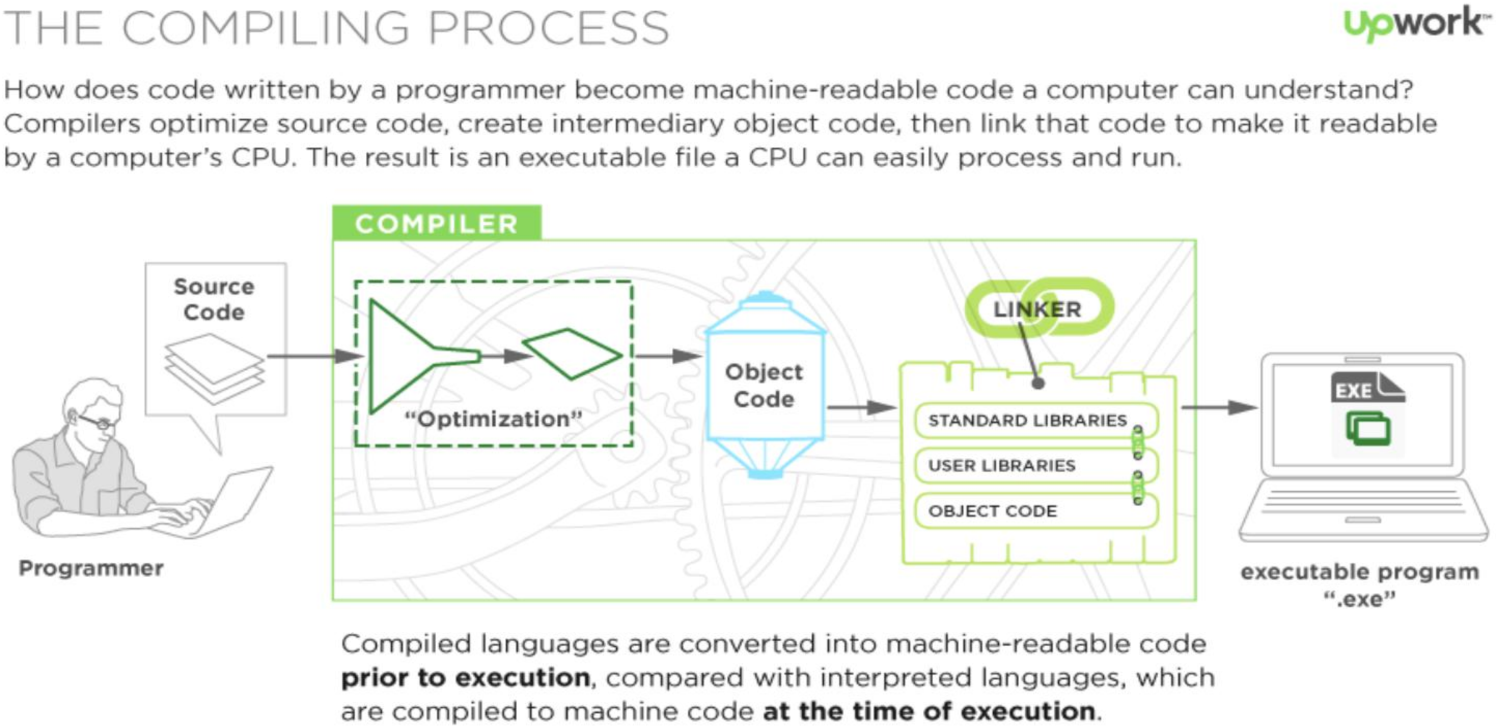


Non-relational databases



Stacks

- Code



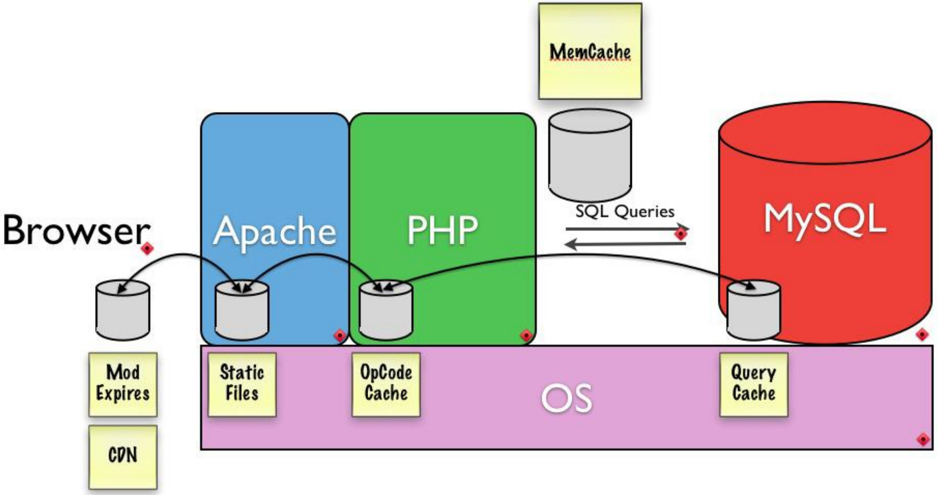
THE COMPILING PROCESS

How does code written by a programmer become machine-readable code a computer can understand? Compilers optimize source code, create intermediary object code, then link that code to make it readable by a computer's CPU. The result is an executable file a CPU can easily process and run.

Compiled languages are converted into machine-readable code **prior to execution**, compared with interpreted languages, which are compiled to machine code **at the time of execution**.

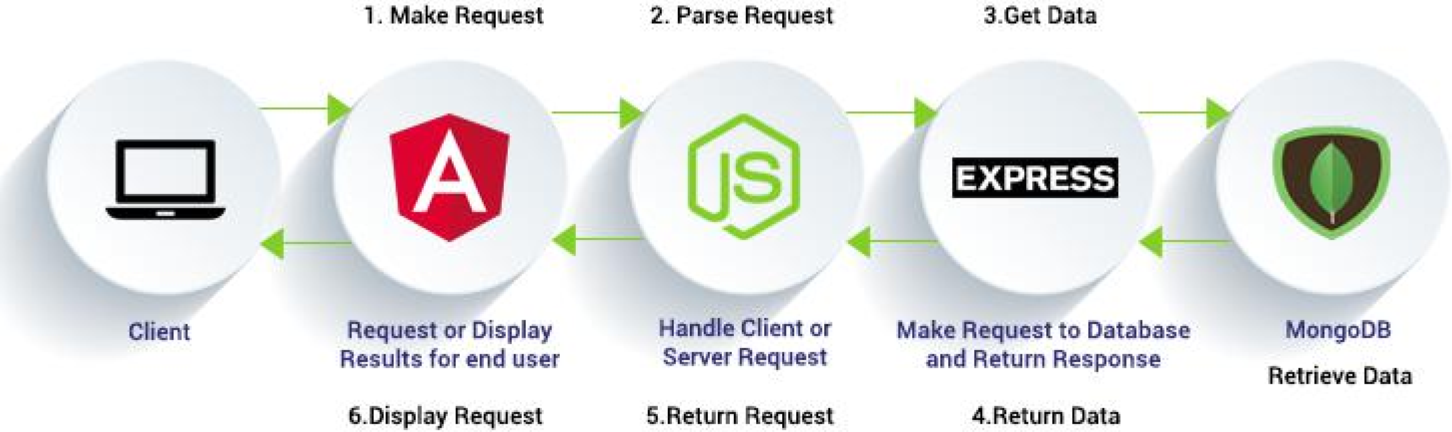
Stacks

LAMP:
Linux
MySQL
php
Apache
MySQL
PHP


MemCache
Browser
Apache
PHP
SQL Queries
MySQL
Mod Expires
CDN
Static Files
OpCode Cache
OS
Query Cache

Stacks

M.E.A.N stack



Stacks

- Framework:

When you're making a sandwich, it's much easier to buy pre-made, sliced bread from the store than it is to bake it on your own from scratch. Frameworks are your site's sliced bread, they speed up the process.

Stacks

- Framework 可以包括
  - Middleware
  - Libraries
  - API
  - Coding shortcuts
  - Performance boosting
  - Caching
  - Security

Stacks

- 常见的Backend Framework
  - 常见的编程语言都有自己的backend framework
    - Python: Django, Flask
    - Ruby: Ruby on Rails
    - PHP: CodeIgniter
    - Javascript: Express.js
    - Java: Spring MVC
    - …

- Backend framework for this camp:

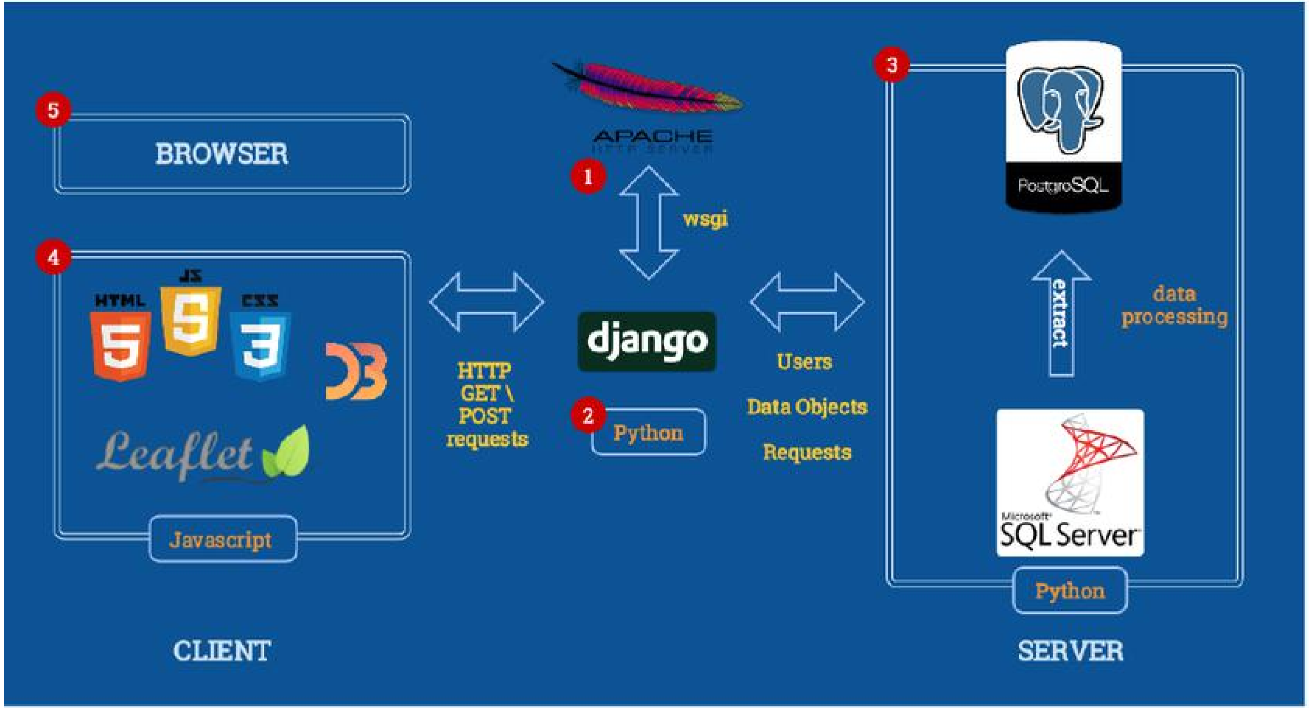
- Why Django?

INST
rdio
DISQUS
mozilla
bitbucket
by ATLASSIAN
LANYRD
the social conference directory
Mahalo
theguardian
The Washington Post
The New York Times
the ONION

- Why Django?
  - Python
  - Fast (Don’t Repeat Yourself)
  - Community
  - Robust enough (Tested by some of the largest websites)
  - Flexible / Friendly enough (super fast for prototype)
  - …

Django周末速成班

- Django backend stack example:



- What does Django include?
  - User authentication
  - Templates, routes, and views
  - Admin interface
  - Robust security
  - Supports multiple databases
  - …

Django周末速成班

- Enough talk, let’s get hands dirty.

Q & A

扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: ninechapter
知乎专栏：http://zhuanlan.zhihu.com/jiuzhang
微博: http://www.weibo.com/ninechapter
官网: www.jiuzhang.com

后端开发初体验

- 开发环境搭建

后端开发初体验

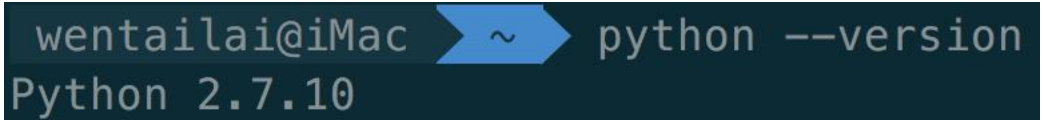
- 开发环境搭建

- Django 2.0
- Python 3.7.x

后端开发初体验

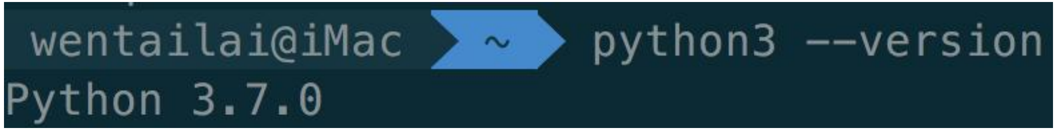
- **开发环境搭建 – Install Python 3**

Step 1: Verify the python version in your desktop



```
wentailai@iMac  ~  python --version
Python 2.7.10
```

For Mac, Python 2 is already installed



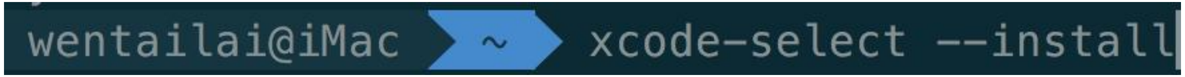
```
wentailai@iMac  ~  python3 --version
Python 3.7.0
```

Use this command to verify if you have python3 installed

后端开发初体验

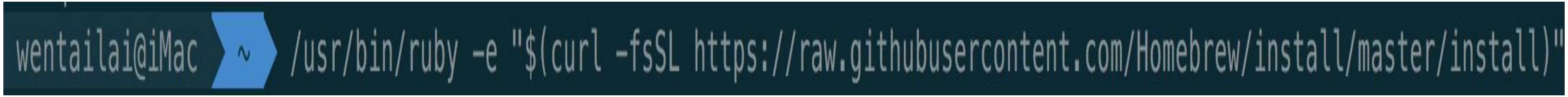
- **开发环境搭建 – Install Python 3**

Step 2: Install Xcode and Homebrew



```
wentailai@iMac  ~  xcode-select --install
```

This might take some time



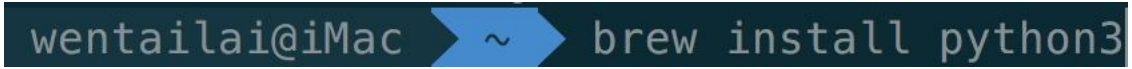
```
wentailai@iMac  ~  /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Homebrew is a package management tool for macOS.
Command can be found in this link: https://brew.sh/

后端开发初体验

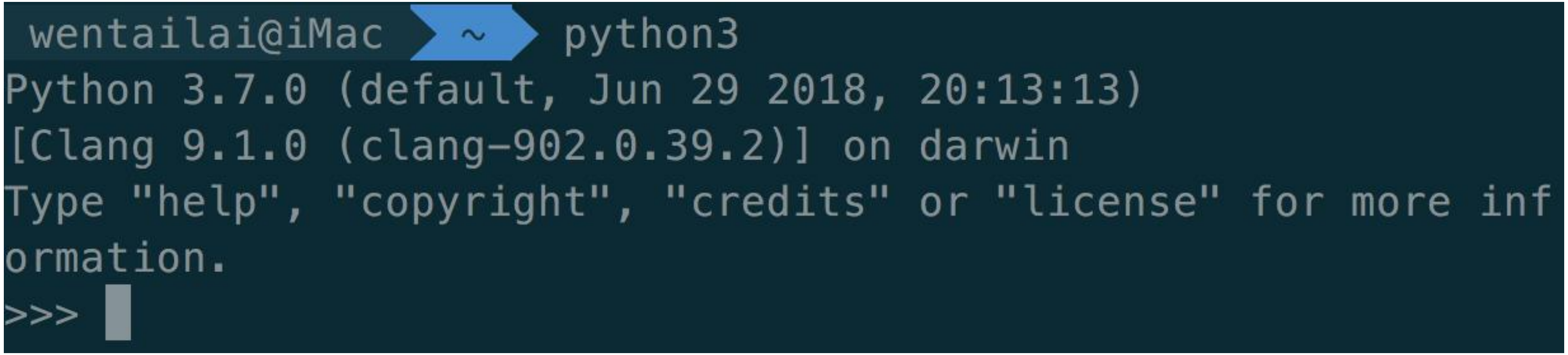
- **开发环境搭建 – Install Python 3**

Step 3: Install Python 3 and verify



```
wentailai@iMac  ~  brew install python3
```

Use Homebrew to install python3



```
wentailai@iMac  ~  python3
Python 3.7.0 (default, Jun 29 2018, 20:13:13)
[Clang 9.1.0 (clang-902.0.39.2)] on darwin
Type "help", "copyright", "credits" or "license" for more inf
ormation.
>>> 
```

You should see the same thing. Use “ctrl + d” to quit

后端开发初体验

- **开发环境搭建 – Install Virtual Environment**

Q:什么是Virtual Environment？

A: 提供一个隔离的环境，这个环境里包括一个project所需要的所有dependencies

Example:

Project A -> Django 2.0

Project B -> Django 1.1

后端开发初体验

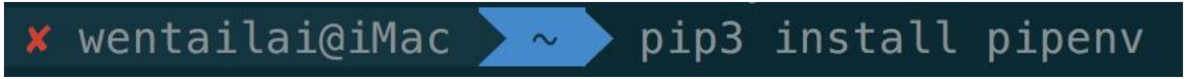
- 开发环境搭建 – **Install Virtual Environment**

V.S.

后端开发初体验

- 开发环境搭建 – **Install Virtual Environment**

Step 1: Install pipenv



```
✘ wentailai@iMac  ~  pip3 install pipenv
```

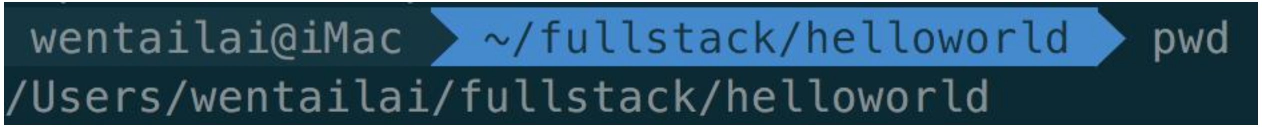
Type this command to install pipenv

(pip3 is a tool installed alone with python3)

后端开发初体验

- 开发环境搭建 – **Install Django**

Step 1: Create your work directory

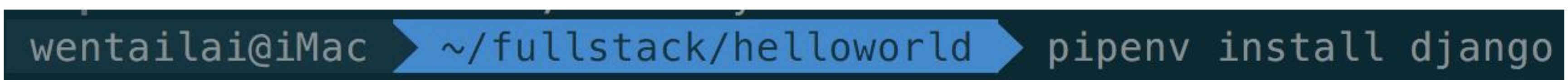


```
wentailai@iMac  ~/fullstack/helloworld  pwd
/Users/wentailai/fullstack/helloworld
```

后端开发初体验

- 开发环境搭建 – **Install Django**

Step 2: Install Django

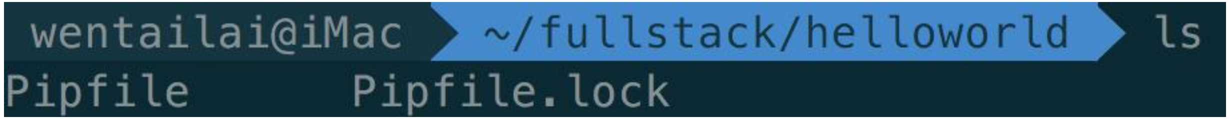


```
wentailai@iMac  ~/fullstack/helloworld  pipenv install django
```

后端开发初体验

- 开发环境搭建 – **Install Django**

Step 3: Check your current directory

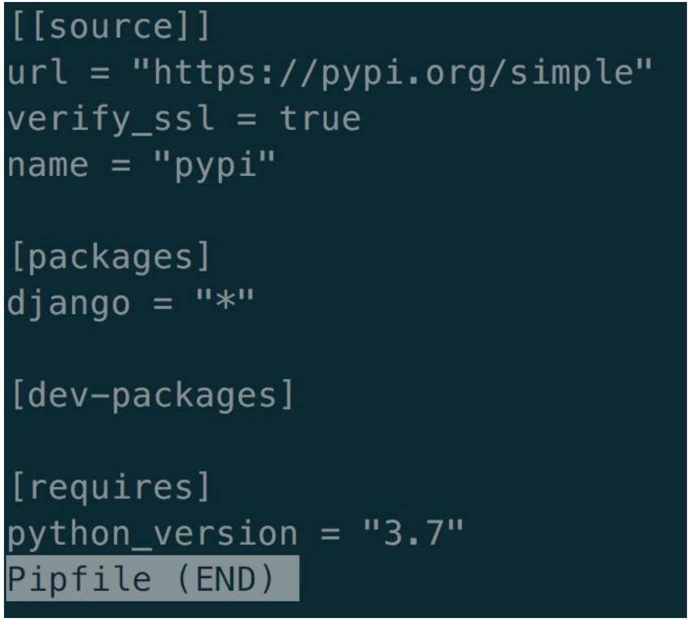


```
wentailai@iMac  ~/fullstack/helloworld  ls
Pipfile        Pipfile.lock
```

后端开发初体验

- **开发环境搭建 – Install Django**

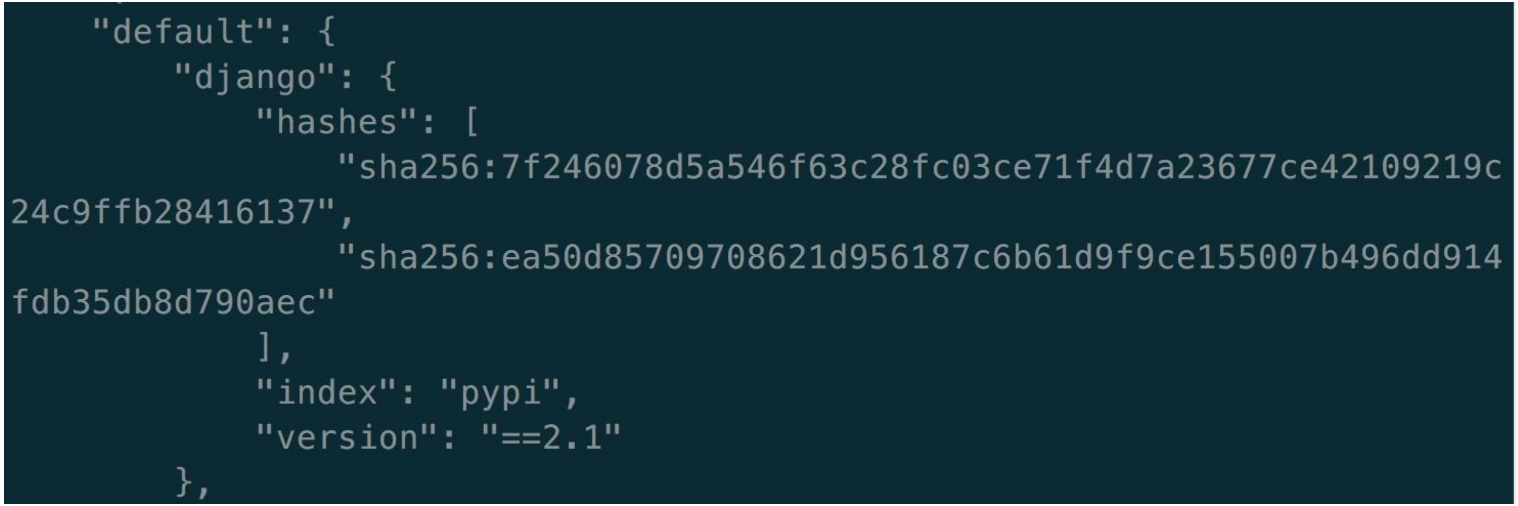
  Pipfile



```
[[source]]
url = "https://pypi.org/simple"
verify_ssl = true
name = "pypi"

[packages]
django = "*"

[dev-packages]

[requires]
python_version = "3.7"
Pipfile (END)
```

- 开发环境搭建 – Install Django

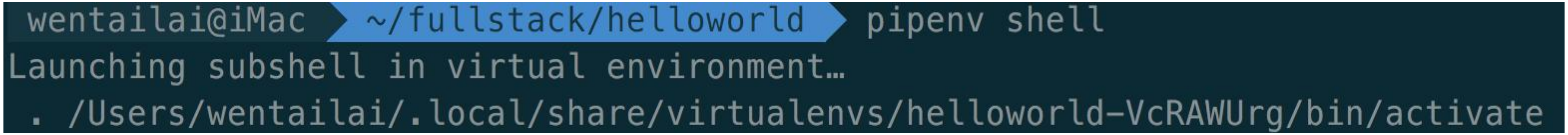
Pipfile.lock



```
    "default": {
        "django": {
            "hashes": [
                "sha256:7f246078d5a546f63c28fc03ce71f4d7a23677ce42109219c
24c9ffb28416137",
                "sha256:ea50d85709708621d956187c6b61d9f9ce155007b496dd914
fdb35db8d790aec"
            ],
            "index": "pypi",
            "version": "==2.1"
        },
```

- **开发环境搭建 – Install Django**

Step 4: Activate this virtual environment

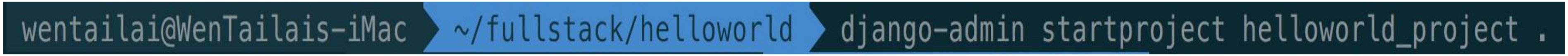


```
 wentailai@iMac  ~/fullstack/helloworld  pipenv shell
Launching subshell in virtual environment…
 . /Users/wentailai/.local/share/virtualenvs/helloworld-VcRAWUrg/bin/activate
```

后端开发初体验

- 开发环境搭建 – **Install Django**

Step 5: Create a new Django project !



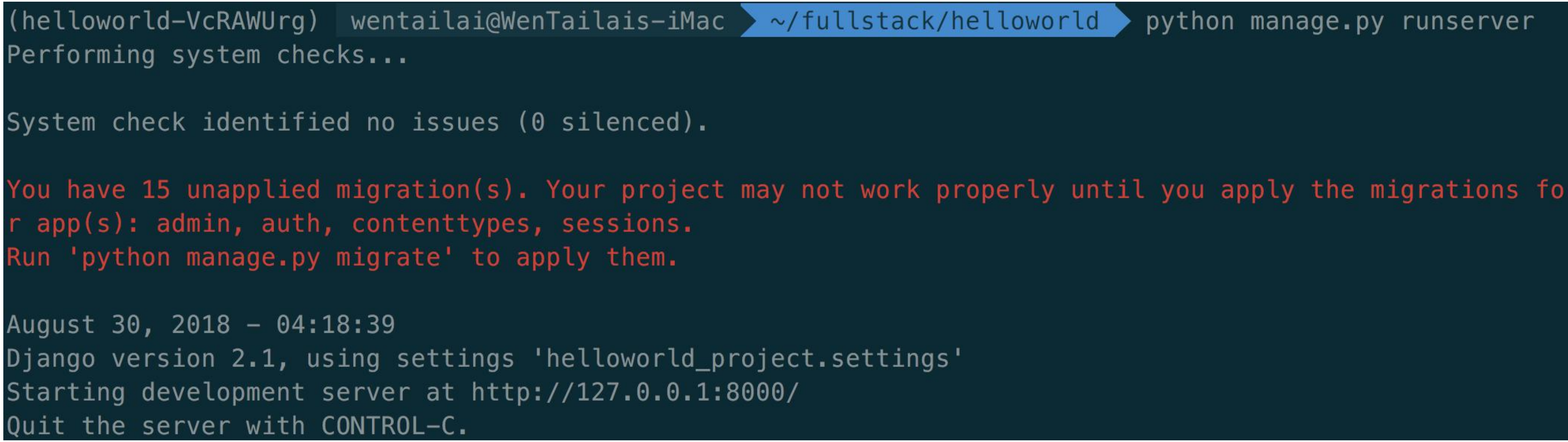
```
wentailai@WenTailais-iMac  ~/fullstack/helloworld  django-admin startproject helloworld_project .
```

```
|__ helloworld
      |__ helloworld_project
            |__ __init.py__
            |__ settings.py
            |__ urls.py
            |__ wsgi.py
      |__ manage.py
```

后端开发初体验

- 开发环境搭建 – **Run Server**

Step 1: Run local web server

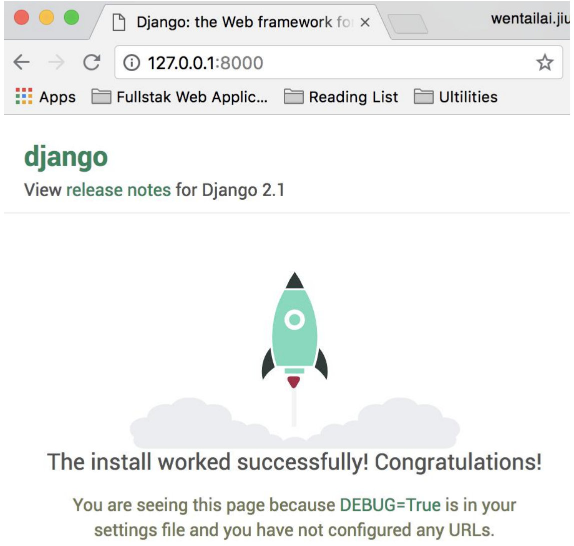


```
(helloworld-VcRAWUrg)  wentailai@WenTailais-iMac  ~/fullstack/helloworld  python manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).

You have 15 unapplied migration(s). Your project may not work properly until you apply the migrations fo
r app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.

August 30, 2018 - 04:18:39
Django version 2.1, using settings 'helloworld_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

后端开发初体验

- 开发环境搭建 – **Run Server**

Step 2: Verify the server is up and running



django

View release notes for Django 2.1

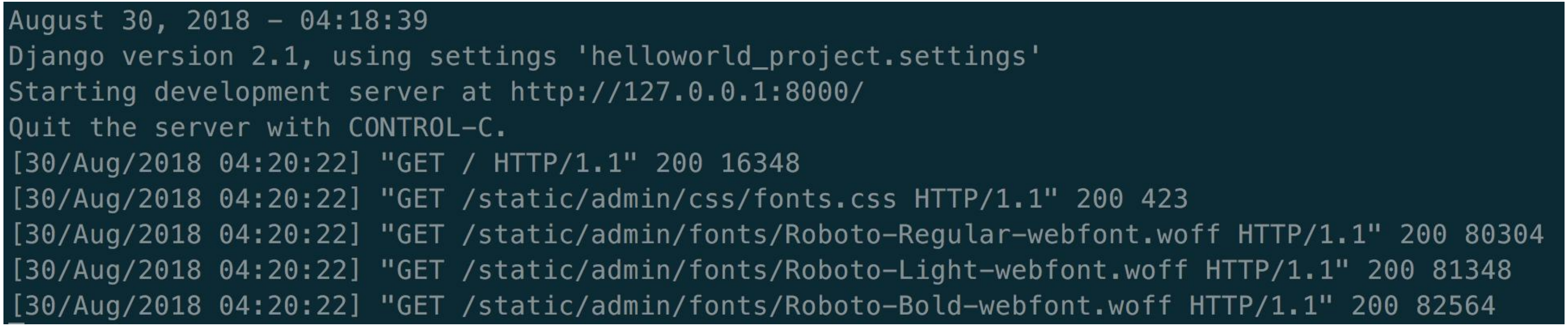
The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

后端开发初体验

- **开发环境搭建 – Run Server**

Step 3: Verify the server is getting request



```
August 30, 2018 - 04:18:39
Django version 2.1, using settings 'helloworld_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
[30/Aug/2018 04:20:22] "GET / HTTP/1.1" 200 16348
[30/Aug/2018 04:20:22] "GET /static/admin/css/fonts.css HTTP/1.1" 200 423
[30/Aug/2018 04:20:22] "GET /static/admin/fonts/Roboto-Regular-webfont.woff HTTP/1.1" 200 80304
[30/Aug/2018 04:20:22] "GET /static/admin/fonts/Roboto-Light-webfont.woff HTTP/1.1" 200 81348
[30/Aug/2018 04:20:22] "GET /static/admin/fonts/Roboto-Bold-webfont.woff HTTP/1.1" 200 82564
```

- 开发环境搭建 – **Run Server**

Step 4: Stop server and exit virtual environment

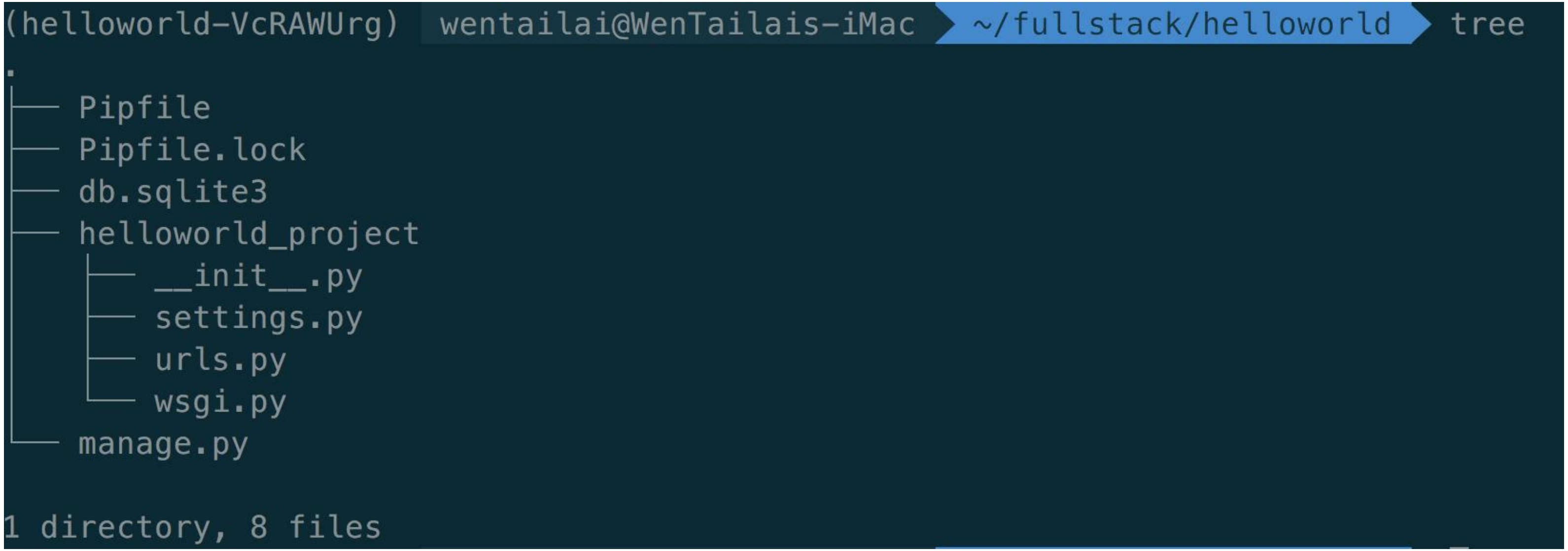
- Stop by press “ctrl + c”
- Exit virtual environment by type “exit”

后端开发初体验

- Now let’s take a close look at our server

后端开发初体验

- Now let’s take a closer look at our server

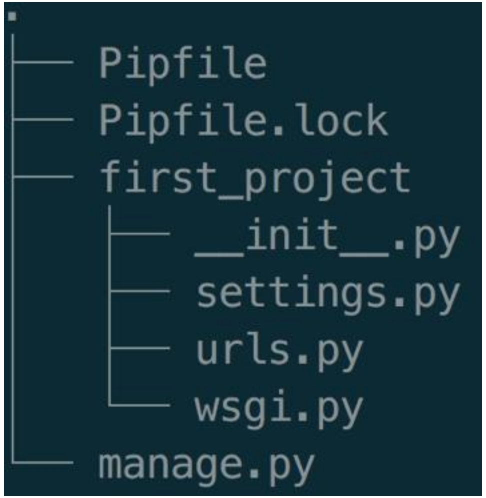


```
(helloworld-VcRAWUrg)  wentailai@WenTailais-iMac  ~/fullstack/helloworld  tree
.
├── Pipfile
├── Pipfile.lock
├── db.sqlite3
├── helloworld_project
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py

1 directory, 8 files
```

后端开发初体验

- Now let’s take a closer look at our server

  - We already know about Pipenv files



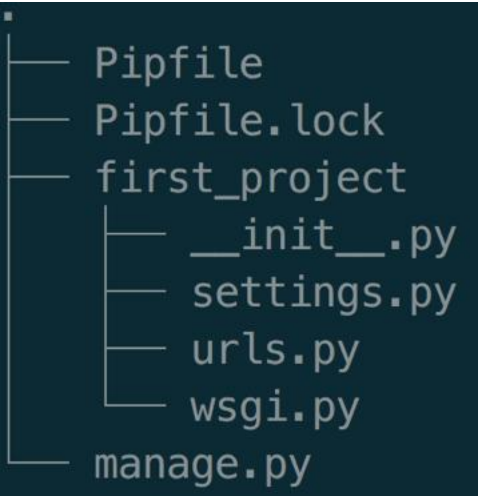
```
.
├── Pipfile
├── Pipfile.lock
├── first_project
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py
```

后端开发初体验

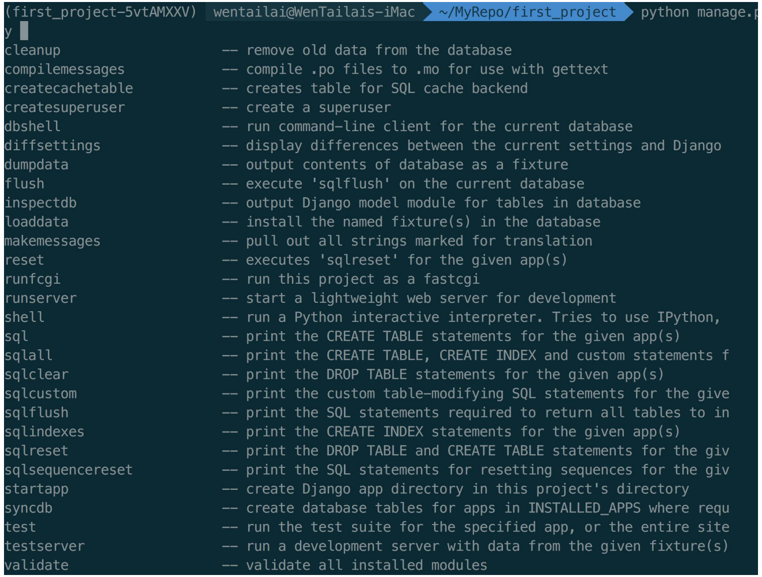
- Now let’s take a closer look at our server

  - manage.py:

    用于执行各种的django commands



```
.
├── Pipfile
├── Pipfile.lock
├── first_project
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py
```



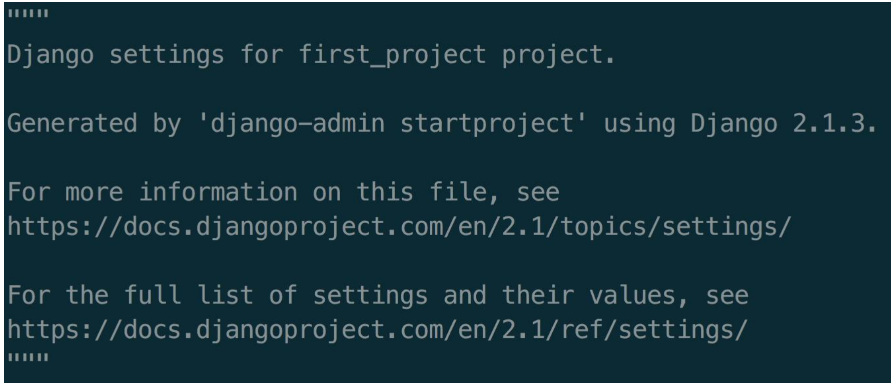
```
(first_project-5vtAMXXV)  wentailai@WenTailais-iMac  ~/MyRepo/first_project  python manage.p
y ▌
cleanup                    -- remove old data from the database
compilemessages            -- compile .po files to .mo for use with gettext
createcachetable           -- creates table for SQL cache backend
createsuperuser            -- create a superuser
dbshell                    -- run command-line client for the current database
diffsettings               -- display differences between the current settings and Django
dumpdata                   -- output contents of database as a fixture
flush                      -- execute 'sqlflush' on the current database
inspectdb                  -- output Django model module for tables in database
loaddata                   -- install the named fixture(s) in the database
makemessages               -- pull out all strings marked for translation
reset                      -- executes 'sqlreset' for the given app(s)
runfcgi                    -- run this project as a fastcgi
runserver                  -- start a lightweight web server for development
shell                      -- run a Python interactive interpreter. Tries to use IPython,
sql                        -- print the CREATE TABLE statements for the given app(s)
sqlall                     -- print the CREATE TABLE, CREATE INDEX and custom statements f
sqlclear                   -- print the DROP TABLE statements for the given app(s)
sqlcustom                  -- print the custom table-modifying SQL statements for the give
sqlflush                   -- print the SQL statements required to return all tables to in
sqlindexes                 -- print the CREATE INDEX statements for the given app(s)
sqlreset                   -- print the DROP TABLE and CREATE TABLE statements for the giv
sqlsequencereset           -- print the SQL statements for resetting sequences for the giv
startapp                   -- create Django app directory in this project's directory
syncdb                     -- create database tables for apps in INSTALLED_APPS where requ
test                       -- run the test suite for the specified app, or the entire site
testserver                 -- run a development server with data from the given fixture(s)
validate                   -- validate all installed modules
```

后端开发初体验

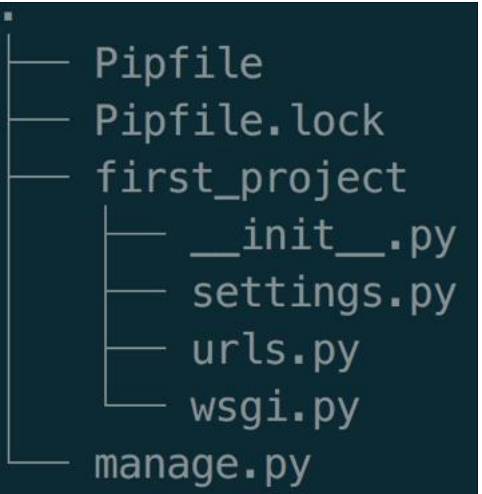
- Now let’s take a closer look at our server

  - settings.py:

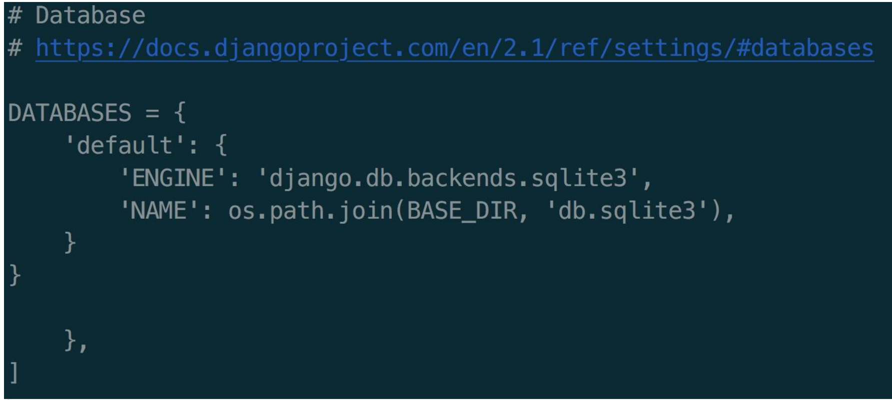
    project level settings, example:



```
"""
Django settings for first_project project.

Generated by 'django-admin startproject' using Django 2.1.3.

For more information on this file, see
https://docs.djangoproject.com/en/2.1/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/2.1/ref/settings/
"""
```



```
.
├── Pipfile
├── Pipfile.lock
├── first_project
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py
```



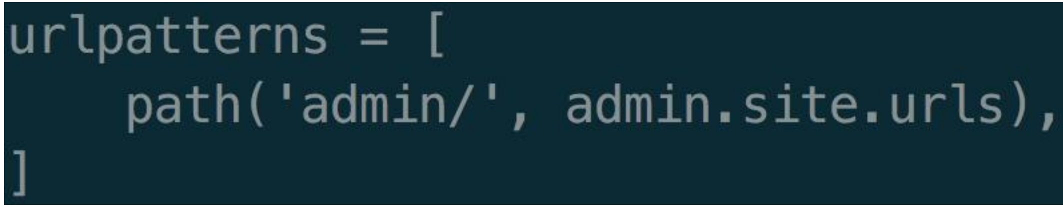
```
# Database
# https://docs.djangoproject.com/en/2.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}


    },
]
```

后端开发初体验

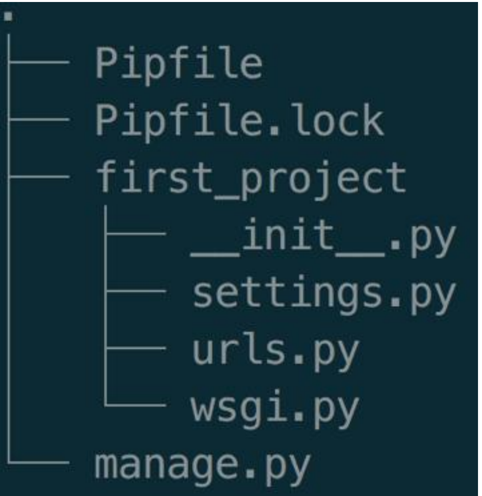
- Now let’s take a closer look at our server

  - urls.py:

    url routing management:



```
urlpatterns = [
    path('admin/', admin.site.urls),
]
```



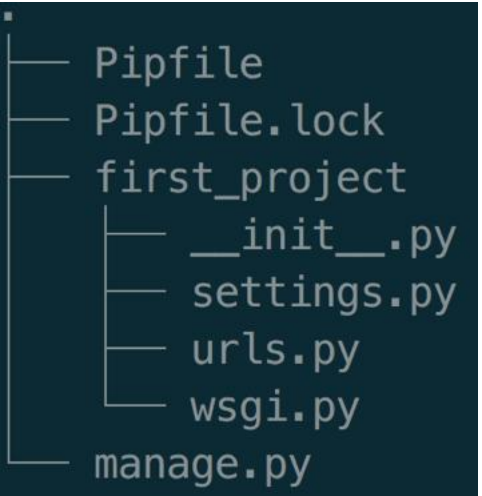
```
.
├── Pipfile
├── Pipfile.lock
├── first_project
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py
```

后端开发初体验

- Now let’s take a closer look at our server

- wsgi.py:

web server gateway interface (describe how server commnunicate with application)



```
.
├── Pipfile
├── Pipfile.lock
├── first_project
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py
```