آموزش کاربردی جنگو (Django)

قوی‌ترین فریم‌ورک توسـعه وب اپلیکیشن در پایتون

مقدماتی تا پیشرفته

Django 3.2.x

Django 4.x

Django 5.x

تألیف: مهندس علیرضا عظیم‌زاده میلانی

انتشارات پندار پارس

انتشارات پندارپارس

دفتر فروش: انقلاب، ابتدای کارگرجنوبی، کوی رشتچی، شماره ١٤، واحد ١٦ **www.pendarepars.com**

تلفن: ٦٦٥٧٢٣٣٥ - تلفکس: ٦٦٩٢٦٥٧٨همراه: ٠٩١٢٢٤٥٢٣٤٨ **info@pendarepars.com**

نام کتاب : آموزش کاربردی جنگو (Django)، قوی‌ترین فریم‌ورک توسعه وب‌اپلیکیشن در پایتون

ناشر : **انتشارات پندار پارس**

ترجمه و تالیف : علیرضا عظیم‌زاده میلانی

چاپ نخست : **مرداد ١٤٠٠**

شمارگان : **٥٠٠ نسخه**

طرح جلد : **رامین شکرالهی**

چاپ، صحافی : **روز**

قیمت : **198000 تومان** **شابک : ٢-٠١-٧٧٨٥-٦٢٢-٩٧٨**

فهرست

درباره مؤلف

مهندس علیرضا عظیم‌زاده میلانی، فارغ‌التحصیل مقطع کارشناسی‌ارشد از دانشگاه خواجه نصیرالدین طوسی تهران – گرایش شبکه‌های کامپیوتری می‌باشند. ایشان بیش‌از ۶ سال تجربه کاری در زمینه‌های تخصصی تست نفوذ وب و شبکه، امن‌سازی و مدیریت وب‌سرورهای لینوکسی، مشاوره و تدریس دوره‌های امنیت شبکه در ورکشاپ‌های دانشگاهی، سازمانی و... دارند.

- ✓ کشف چندین آسیب‌پذیری امنیتی و ثبت در پایگاه‌داده آرشیو اکسپلویت‌ها (Exploit-DB).
- ✓ جزو 100 نفر برتر نخستین مسابقه طراحی Instant-View در شبکه اجتماعی تلگرام.
- ✓ سخنران کنفرانس‌ها و همایش‌های علمی درباره موضوعات "راهکارهای نفوذ و مقابله با تهدیدهای امنیتی در شبکه‌های کامپیوتری"، "نوآوری با پایتون"، "درآمدزایی قانونمند از صنعت سایبری" و....
- ✓ تألیف و ترجمه چندین کتاب در زمینه‌های تخصصی نرم‌افزار و فناوری‌اطلاعات:
  ۱) هک و امنیت با توزیع لینوکسی Back-Track
  ۲) مدیریت وب سرور با توزیع لینوکسی CentOS
  ۳) آموزش کاربردی تست‌نفوذ شبکه با کالی لینوکس (Kali-Linux)
  ٤) آموزش کاربردی برنامه‌نویسی به زبان پایتون
  ٥) آموزش کاربردی تست‌نفوذ وب (با رویکرد کشف باگ و درآمدزایی ارزی-ریالی)
- ✓ تست نفوذ، همکاری و دریافت پاداش (Bounty)، تقدیرنامه (Ack/Cert) و HoF (Hall of Fame) از دپارتمان امنیت شرکت‌های مشهور خارجی و داخلی:

BMW	Siemens	Qualcomm	Symantec	Synology
ABN-Amro	RighTel	AloPeyk	Bazaar	PhonePay

و تعدادی شرکت خارجی و داخلی دیگر ....

Penetration-Testing: PWK, CEH, WiFu

Linux: RHCE, RHCSA, LPIC-2, LPIC-1

Network: CCNA, CWNA, CWSP

Programming: Python, Django

لطفاً بخوانید .... ❤

اگر این کتاب مورد پسند شما واقع شد و توانست گره‌ای از کارتان بگشاید، از انجام دو کار دریغ نفرمایید: نخست، دعا و انرژی مثبت برای این حقیر و نویسندگان اصلی کتبی که اینجانب آن‌ها را مطالعه کرده‌ام و نتیجه آن مطالعات و تجارب شخصی، تبدیل به این کتاب کاربردی شده است. دوم، معرفی این کتاب و دیگر کتاب‌های انتشارات پندارپارس به دوستان و اساتید خود در گروه‌ها و کانال‌های شبکه‌های اجتماعی. چرا که تنها این حمایت‌هاست که موجب دلگرمی ما به تولید آثار به‌روز این حوزه، برای شما علاقه‌مندان است.

در این کتاب سعی شده است تا شما خوانندگان با بیشتر اَبعاد فریم‌ورک جنگو آشنا شوید و در پایان به‌عنوان یک متخصص جنگو شناخته شوید. با توجه به حجم مطالب و از آنجا که هیچ اثری بی‌اشکال نیست، از شما دوستان گرامی تقاضا دارم نظرها و پیشنهادهای خود را به آدرس ایمیل Ali.Azimzadeh70@Gmail.com با عنوان "کتاب جنگو" یا در صفحه اختصاصی این کتاب در سایت پندارپارس (PendarePars.com) با اینجانب در میان بگذارید. ❤❤

در آخر، از حمایت مادی و معنوی شما خوانندگان عزیز و گران‌قدر که از کپی‌برداری، اسکن و الکترونیکی کردن کتاب پرهیز می‌نمایید و مؤلف و ناشر را برای تألیف و نشر کتاب‌های بعدی یاری می‌رسانید صمیمانه سپاسگزاری می‌نمایم.

این کتاب به چه افرادی پیشنهاد می‌شود !؟

اگر مصمم به یادگیری اصولی فریم‌ورک جَـنگو به‌منظور ساخت وب‌سایت و وب اپلیکیشن هستید، و با زبان برنامه‌نویسی پایتون (در سطح پیش‌متوسط)، با مفاهیم مرتبط به وب سرور (در سطح پایه) و با زبان انگلیسی (در سطح سال آخر دبیرستان) آشنایی خوبی دارید، مطالعه این کتاب به شما پیشنهاد می‌شود.

مخاطبان این کتاب، افراد علاقه‌مند به ساخت و توسعه وب اپلیکیشن در وب سرورهای ویندوز و لینوکس (سمت Back-End)، مدرسان، دانشجویان و دانش‌آموزان مقاطع تحصیلی کار و دانش، دبیرستان و هنرستان در رشته‌های نرم‌افزار، سخت‌افزار، فناوری‌اطلاعات و الکترونیک می‌باشند.

دقت داشته باشید که این کتاب یک مرجع تئوری دانشگاهی نیست؛ بلکه بیش‌از ۲۵۰ مثال کاربردی دارد و تنها کاری که شما باید در راستای یادگیری هرچه بهتر جَـنگو در این کتاب انجام دهید، داشتن پشتکار، تمرکز در مطالعه، صبر و حوصله، آزمون و خطا و تمرین زیاد است (نه copy-paste). موارد ذکر شده، رمز پیروزی و موفقیت شما در یادگیری این حجم از مطالب است.

نیاز به راهنمایی دارم. روش‌های ارتباط با نویسنده کتاب !؟

برای ارتباط با نویسنده و دریافت آخرین اخبار مرتبط با کتاب، پیرامون جلد دوم کتاب، دانلود سورس کد مثال‌ها، پرسش و پاسخ، دوره‌های آموزشی، کلاس‌های آنلاین رفع اشکال و سایر موضوعات، می‌توانید از طریق مسیرهای ارتباطی زیر اقدام کنید:

Telegram Channel (Public): http://t.me/django_persian	Telegram Group (Private): http://t.me/joinchat/msVarKgCNDMyMDY0
E-Mail: ali.azimzadeh70@gmail.com	PendarePars: PendarePars.com/blog/alireza-azimzadeh
Linked-in: www.linkedin.com/in/alireza-azimzadeh-34874079/	

آیا محتوای این کتاب در نسخه‌های آتی 4.x و 5.x جنگو، به‌روز و قابل استفاده است!؟ واقعاً مطالب آموزشی کتاب تا چه مدت زمانی اعتبار دارد !؟

در ابتدا، این کتاب صرفاً یک Handbook شخصی برای مطالعه و ارجاعات روزانه بود ((چون نظرم بر این است که مستندات جنگو بسیار کامل، اما کمی نامنظم در سایت docs.djangoproject.com ارائه شده‌اند)).

از آنجایی‌که چندین کتاب انگلیسی نام آشنا و چندین ویدئوی آموزشی خارجی و ایرانی برای یادگیری جنگو (از نسخه 1.11 گرفته تا 3.0) مطالعه کرده و دیده بودم، و هر کدام یک دردسر آموزشی و ناسازگاری نسخه‌ای داشتند، چنین مواردی باعث می‌شد به دفعات به سایت رسمی مستندات جنگو رجوع کنم که واقعاً خسته‌کننده بود. پس از گذشت مدتی، تصمیم گرفتم کتاب را به‌گونه‌ای ویرایش و نگارش کنم که بدانم هر یک از مطالب آموزشی در کدام یک از نسخه‌ها بطور کامل منقضی/ منسوخ (deprecated) می‌شوند و در کدام یک از نسخه‌ها بطور کامل حذف (removed) خواهند شد.

خوشبختانه، نتیجه جست‌وجوها در انجمن‌های خارجی و مستندات جنگو، رسیدن به بخش‌های مهمی مانند "Internals"، "Releases"، "Wiki" و "Contributing" بود.

نکته: جنگو از ساختار نسخه‌بندی سِمَنتیک (semantic versioning) استفاده می‌کند:

docs.djangoproject.com/en/dev/internals/release-process/#release-cadence

در ابتدا به بررسی تک‌تک مستندات بخش Releases پرداختم؛ که چه تغییراتی در نسخه‌ها ایجاد شده است. به‌عنوان مثال، معرفی نوع‌داده BigAutoField به‌جای AutoField در نسخه 3.2 جنگو با حفظ سازگاری (compatibility).

https://docs.djangoproject.com/en/dev/releases/

https://docs.djangoproject.com/en/dev/releases/3.2/#customizing-type-of-auto-created-primary-keys

و همین فرآیند بررسی را برای سایر مطالب آموزشی کتاب پیش بردم که واقعاً وقت‌گیر بود؛ ولی ارزش داشت.

به‌عنوان مثال، در یادداشت‌های **نسخه 4.0 جنگو** اشاره شده که قرار است چه ویژگی‌هایی در نسخه 4.0 منقضی شوند؛ یا چه ویژگی‌هایی قرار است در این نسخه حذف شوند و دیگر وجود نخواهند داشت:

https://docs.djangoproject.com/en/dev/releases/4.0/

در ادامه به بررسی زیربخش‌های Internals (release-process و deprecation-timeline) پرداختم. در این زیربخش‌ها، ویژگی‌هایی که قرار است در هر یک از نسخه‌های جنگو منقضی و حذف شوند را تک‌تک مورد بررسی قرار دادم. به‌عنوان مثال: **نسخه 5.0 جنگو**:

https://docs.djangoproject.com/en/dev/internals/deprecation/#deprecation-removed-in-5-0

https://docs.djangoproject.com/en/dev/internals/release-process/#deprecation-policy

https://docs.djangoproject.com/en/dev/internals/contributing/writing-code/submitting-patches/#deprecating-a-feature

https://www.djangoproject.com/weblog/2015/jun/25/roadmap/

آخرین بخش مهم، بررسی سیاست چگونه منسوخ/حذف کردن ویژگی‌ها بود. با مطالعه این بخش (deprecation-policy) می‌توان نتیجه‌گیری کرد که محتوای آموزشی این کتاب کاربردی، قطعاً تا پایان عمر نسخه 5.0 جنگو اعتبار دارد (یعنی تا پایان سال 2026 میلادی).

نکته: از آنجایی که موضوعات اصلی و هسته‌ی (core) جنگو در این کتاب مورد بحث و بررسی قرار گرفته است، نظر بر این است که مطالب کتاب تا پایان عمر **نسخه 5.2 LTS جنگو** نیز اعتبار خواهند داشت (یعنی تا پایان سال **2027** میلادی).

این حجم از مطالب و نکات آموزشی را چگونه بیاموزم !؟

اگر به تازگی شروع به یادگیری جنگو کرده‌اید یا آشنایی نسبی با این فریم‌ورک دارید، دقت داشته باشید که در شروع کار قرار نیست تمامی مطالب کتاب را مطالعه و تمرین کنید. چون برخی از عناوین آموزشی برای سطوح متوسط و پیشرفته هستند و پیش‌نیاز مطالعه این موضوعات، یادگیری کامل و با دقت مطالب سطح مقدماتی است ((به‌عنوان نمونه: "گزینه‌های اختیاری در ایجاد وراثت (proxy و abstract)"، "تراکنش‌ها در مدل"، "استفاده از چندین دیتابیس در مدل"، "سیگنال‌ها در مدل"، "نماهای کلاسی و میکسین"، "برچسب‌های حلقه‌ها"، "مدیریت پیام‌های خطا در فرم"، "سرویس‌های REST" و "عناوین حاوی کلمه سفارشی‌سازی" و...)). پیشنهاد می‌شود محتوای هر فصل را یک‌بار تا پایان روخوانی کنید تا یک دید کلی نسبت به موضوعات به‌دست آورید و سپس بهترین روش را برای یادگیری انتخاب نمایید.

موضوعات جلد دوم کتاب و نحوه تهیه آن

عناوین آموزشی که در جلد دوم کتاب مورد بحث و بررسی قرار خواهند گرفت در جدول زیر لیست شده‌اند؛ و ممکن است محتوای آموزشی برخی عنوان‌ها کمتر یا بیشتر شود. همچنین لازم به ذکر است که چاپ جلد دوم کتاب منوط به حمایت مادی و معنوی شما خوانندگان عزیز و گران‌قدر از کتاب کنونی است. ❤️❤️

کانال‌ها (Channels): مدیریت وب سوکت‌ها، پروتکل‌های چت، پروتکل‌های اینترنت اشیاء و...	امنیت (Security & Hardening): امن‌سازی اپلیکیشن‌ها، تنظیمات پروژه و وب‌سرورهای لینوکسی.
کش (Cache): آشنایی با انواع روش‌های Caching	ذخیره‌ساز (Storage): آشنایی با انواع روش‌های ذخیره‌سازی
ایجاد پروژه در محیط عملیاتی: هاست پایتون/جنگو، سرور مجازی لینوکسی (VPS) و پلتفرم‌های ابری خارجی	آشنایی با مباحث تکمیلی ORM جنگو: کوئری‌های خام (RAW Queries)، ادغام کوئری‌ها و...
آشنایی با ابزارها و روش‌های کنترل کارائی و بهینه‌سازی (Performance & Optimizations)	آشنایی با ابزارهای ثبت وقایع و رهگیری درخواست‌ها و پاسخ‌ها (Issue/User Tracking & Analytics & Logging)

آشنایی با توابع و نماهای Async (Asynchronous Views & Functions)	آشنایی با ابزارهای مدیریت صف‌ها و زمان‌بندها (Queues & Schedulers)
آشنایی با روش‌های توزیع‌باز (Load Balancing + CDNs)	آشنایی با روش‌های مقایس‌پذیری (Vertical & Horizontal Scaling)
آشنایی با درگاه‌های پرداخت اینترنتی	آشنایی با فریم‌ورک Fast-API
معرفی روش‌های ایده‌آل (Best Practices) برای بهبود عملکرد پروژه‌های جنگو	

چگونه با کانال‌ها و انجمن‌های مرتبط با جنگو آشنا شوم !؟

هر وقت احساس کردید که نیاز به طرح پرسش و رفع اشکال، یافتن دوست و هم‌تیمی، آموختن مطالب آموزش تکمیلی، دریافت راهنمایی و مشاوره در زمینه‌های پایتون، جنگو و لینوکس داشتید، در شبکه‌های اجتماعی زیر عضو شوید:

Persian/Farsi language:

https://t.me/django_community	https://youtube.com/c/ShahriarShariati
https://t.me/django2	https://youtube.com/c/Bobycloud
https://t.me/djangoir	https://youtube.com/c/Silicium7
https://t.me/django	https://t.me/djangolearn_lr

English language:

https://youtube.com/c/ParwizForogh	https://youtube.com/c/ReversePython
https://youtube.com/c/VitorFreitas	https://youtube.com/c/VeryAcademy
https://youtube.com/c/CryceTruly	https://youtube.com/c/Pyplane

اگر مصمم به یادگیری اصولی ساخت و توسعه برنامه‌های تحت وب هستید، حتماً بخوانید:

خلاصه و بی‌تعارف، اگر یک برنامه‌نویس جوان(Junior) یا ارشد(Senior) هستید و می‌خواهید برنامه‌های تحت وب امن توسعه دهید که تا حد بسیار قابل قبولی نسبت به حملات سایبری مصون باشند، در سری کتاب‌های فارسی، تندخوانی کتاب "آموزش کاربردی تست نفوذ وب" به شدت توصیه می‌شود (نه به این‌خاطر که نویسنده این کتاب هستم). چون باید بدانید که هکرها از طریق چه حفره‌ها و روش‌های جدیدی اقدام به نفوذ در سایت و سرور شما می‌کنند. در آخر، شاید برایتان جالب باشد که بدانید با توجه به اینکه کتاب در پاییز ۱۳۹۵ به چاپ رسیده است، محتوای آن همچنان به‌روز و قابل استفاده است. و برای این موضوع که توانستم چنین کتابی برای علاقه‌مندان نگارش کنم، قدردان و سپاسگزار خداوند متعال هستم. ❤❤❤

مسئله این نیست که خرید کتاب چقدر گران تمام می‌شود،

مسئله این است که اگر کتاب نخوانی چقدر برایت گران تمام می‌شود !

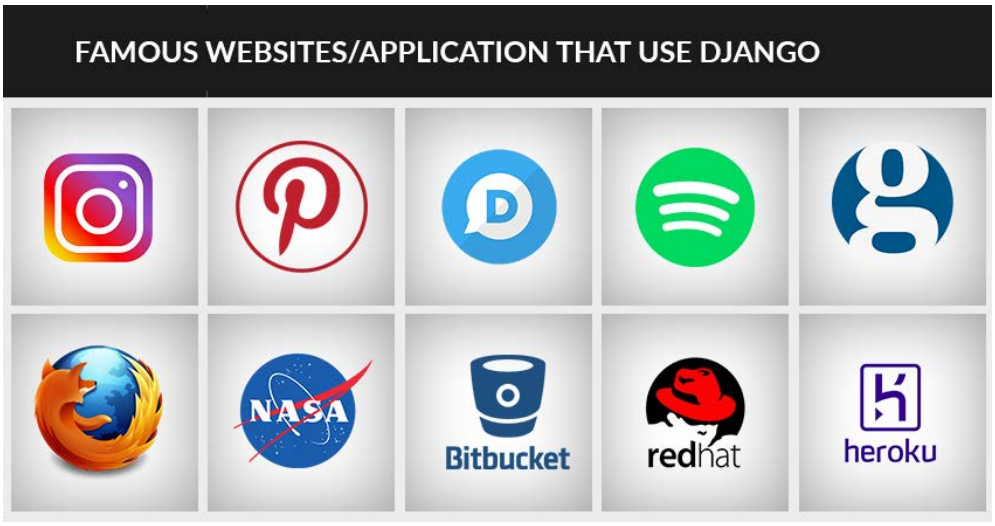
فصل نخست

مقدمه‌ای بر فریم‌ورک جنگو

۱-۱ مقدمه

پایتون یکی از محبوب‌ترین زبان‌های برنامه‌نویسی جهان است و به‌عنوان سومین زبان پر استفاده پس از Java و C در پروژه‌های سایت گیت‌هاب (Github.com) در سال 2019 معرفی شده است. بنابراین اگر شما به پایتون علاقه‌مند هستید، قطعاً درباره فریم‌ورک (framework) قدرتمند جَـنگو (Django) که برپایه پایتون است مطالبی دیده‌اید یا شنیده‌اید. به‌وسیله فریم‌ورک جنگو می‌توانید اپلیکیشن‌های تحت‌وب بسیار قدرتمند، پیچیده و حرفه‌ای در سریع‌ترین زمان و با امنیت بالا بسازید؛ و حتی از قابلیت‌های گسترده زبان پایتون (مانند مصورسازی داده، تحلیل‌داده، پردازش متن و تصویر، یادگیری ماشین و...) در این فریم‌ورک بهره‌مند شوید.

شاید برایتان جالب باشد که بدانید چه شرکت‌های برجسته‌ای درحال استفاده از فریم‌ورک جنگو هستند:



https://octoverse.github.com/

https://www.tiobe.com/tiobe-index/

https://insights.stackoverflow.com/survey/

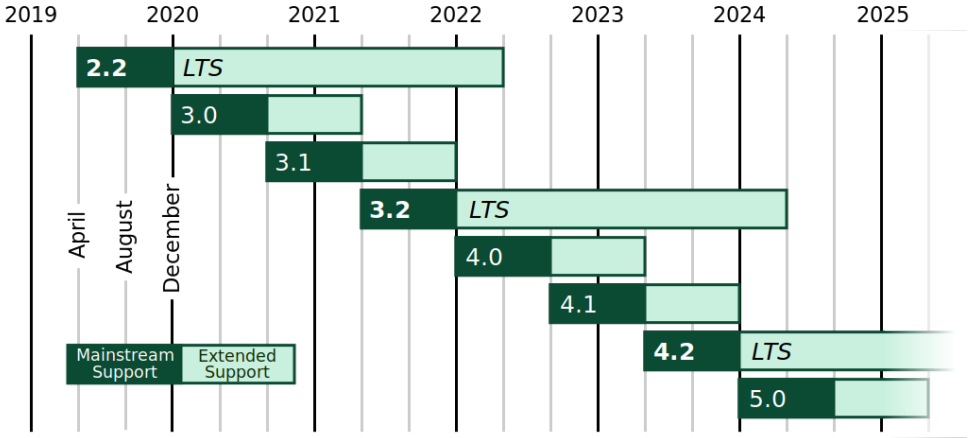
https://stackify.com/popular-programming-languages-2018/

ساخت پروژه جنگو در سال 2003 توسط آدریان‌هولواتی (Adrian Holovaty) و سیمون‌ویلسون (Simon Willison) در نشریه Journal-World به‌هنگام ساخت برنامه‌های پایتونی آغاز شد. در سال 2005 با پیوستن توسعه‌دهندگان حرفه‌ای به تیم آنها، تصمیم گرفته شد این فریم‌ورک به‌شکل متن‌باز(open source) و عمومی منتشر شود. آنها فریم‌ورک خود را در جولای سال 2005 به یادبود گیتاریست سبک جاز "جنگو رینهارت" به‌نام جنگو(Django) منتشر کردند. درنهایت در ژوئن سال 2008، بنیاد نرم‌افزاری DSF (Django Software Foundation) برای توسعه پایدار و نگهداری از فریم‌ورک جنگو در سطح بین‌المللی شکل گرفت.

www.djangoproject.com/weblog/2008/jun/17/foundation/

برابر آمار رسمی منتشر شده در دسامبر سال 2017، بیش‌از هـزار نفر شرکت‌کننده (contributor) روی هسته این فریم‌ورک درحال فعالیت بوده‌اند و بیش‌از چـهار هزار بسته (package) برای کار با

جنگو منتشر شده است. و تاکنون این فریم‌ورک بیش‌از ۱۵ نسخه(version)، انتشار(release) داشته است. در دو تصویر زیر می‌توانید بخشی از نقشه انتشار نسخه‌های جنگو را مشاهده کنید:



Release Series	Release Date	End of mainstream support[1]	End of extended support[2]
3.2 LTS	April 2021	December 2021	April 2024
4.0	December 2021	August 2022	April 2023
4.1	August 2022	April 2023	December 2023
4.2 LTS	April 2023	December 2023	April 2026

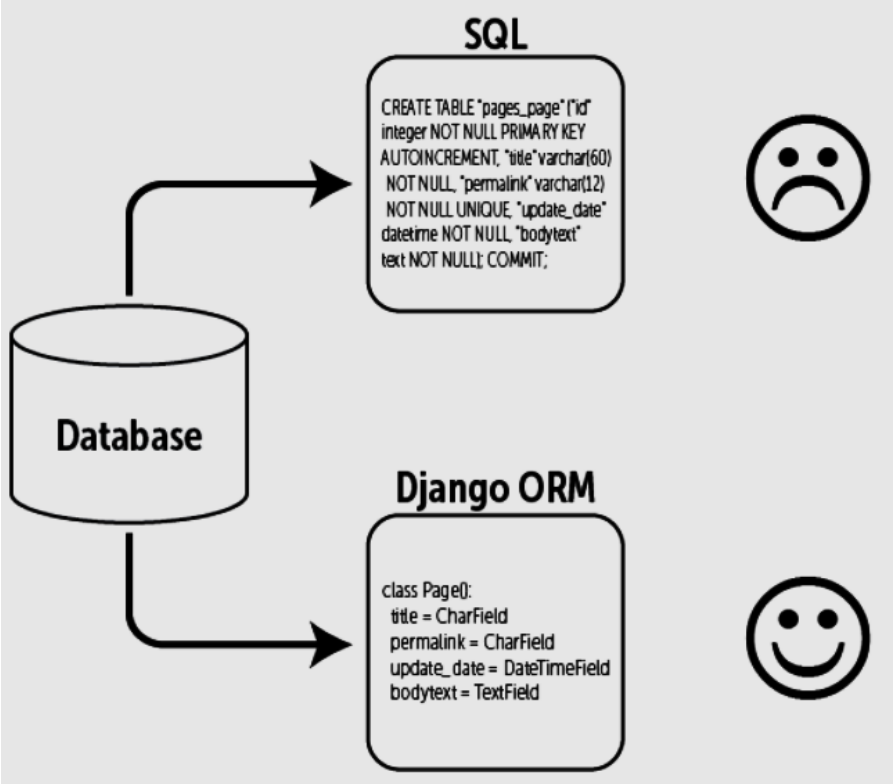
بنیاد نرم‌افزاری DSF جنگو پیوسته در تلاش است تا قابلیت‌های کاربردی به فریم‌ورک جنگو اضافه شوند؛ که تا حد زیادی نیازها و نگرانی‌های برنامه‌نویسان را در نظر گرفته و به رفع آن‌ها کمک نماید. این رویکرد باعث‌شده که جنگو به یکی از برترین فریم‌ورک‌های پایتون در سال‌های اخیر تبدیل شود. در ادامه، به‌شرح تعدادی از مزایای مهم جنگو می‌پردازیم:

- ✓ **کامل بودن**: بنیاد جنگو تلاش کرده به همه جوانب فرآیند توسعه یک برنامه تحت‌وب توجه کند و با ارائه مستندات غنی و استانداردهای طراحی اصولی، به‌بهترین شکل ممکن از برنامه‌نویسان حمایت و پشتیبانی نماید.
- ✓ **امن بودن**: جنگو در حوزه امنیت، شما را از بسیاری از اشتباه‌های رایج امنیتی مانند HostHeader-Injection، CSRF/XSS-Attacks، SQL-Injection، Clickjacking و...

مصون می‌دارد؛ و حتی به پژوهشگران امنیت و شکارچیان باگ که موفق به کشف باگ امنیتی شوند، پاداش نقدی در قالب برنامه "Bug Bounty" پرداخت می‌کند.

- ✓ **قابلیت حمل و توسعه در بین سیستم‌ها:** جنگو برپایه زبان پایتون نوشته شده است و به همین دلیل می‌تواند روی سیستم‌عامل‌های مختلفی مانند ویندوز، لینوکس، مک و... براحتی اجرا گردد. همچنین این فریم‌ورک، به‌خوبی توسط شرکت‌های ارائه‌دهنده خدمات هاستینگ و سرویس‌های ابری (PaaS) پشتیبانی می‌شود.
- ✓ **پر کاربرد بودن:** برای ساخت انواع سایت در زمینه‌های تولید محتوا، خبری، فروشگاهی، شبکه‌اجتماعی و... می‌توان از جنگو استفاده کرد. این فریم‌ورک با فریم‌ورک‌های سمت کاربر مانند Vue.js، React و... تعامل بسیار خوبی دارد و قادر به ارائه محتوا در فرمت‌های مختلفی مانند HTML، JSON، XML و... می‌باشد.
- ✓ **قابلیت نگهداری و استفاده مجدد:** جنگو قابلیتی برای استفاده چندباره از کدها فراهم می‌کند و با جلوگیری از نوشتن کدهای تکراری، به میزان قابل‌توجهی حجم کدها را کاهش می‌دهد. همچنین قابلیت گروه‌بندی کدهای مرتبط در یک ماژول را برای استفاده مجدد دارد.
- ✓ **مقیاس‌پذیری:** بزرگ یا کوچک بودن پروژه برای جنگو تفاوت چندانی ندارد. این فریم‌ورک قدرتمند توانایی پشتیبانی و توسعه پروژه‌های کوچک و تبدیل آن‌ها به پروژه‌هایی با ترافیک و حجم زیادی از اطلاعات را دارد.
- ✓ **انجمن‌های پشتیبانی (Community):** برنامه‌نویسان زیادی در سراسر دنیا از جنگو استفاده می‌کنند. بنابراین در صورت بروز مشکل، سایت‌ها و کانال‌های ارتباطی زیادی وجود دارند تا بتوانید با تجربه‌ی سایر کاربران، سطح پروژه خود را ارتقاء داده و مشکلتان را برطرف نمایید.
- ✓ **پشتیبانی از انواع دیتابیس و سئو (SEO):** جنگو از دیتابیس‌های مختلفی مانند MySQL، MariaDB، Oracle، PostgreSQL، MongoDB و... پشتیبانی می‌کند؛ و با توجه به نوع پروژه، امکان استفاده از چندین دیتابیس را به‌طور همزمان در اختیارتان می‌گذارد. همچنین جنگو در بهبود و بهینه‌سازی سئو سایت کمک شایانی می‌کند.

جنگو از موتور ORM (Object Relation Mapper) برای ارتباط با دیتابیس(ها) استفاده می‌کند. مزیت این موتور، جلوگیری نسبی از وقوع باگ‌های امنیتی مانند SQL-Injection در سمت دیتابیس است. در واقع، ارتباط بین داده‌های مدل و کوئری‌های SQL را موتور ORM تأمین می‌کند. حتی اگر در حین کار، نوع دیتابیس را هم تغییر دهیم هیچ نگرانی وجود ندارد و خود ORM این مسئله را مدیریت می‌کند. ♥♥



همان‌طور که می‌دانید زبان برنامه‌نویسی پایتون در دو نسخه ۲ و ۳ منتشر شده است و برای کار با فریم‌ورک جنگو باید از یک نسخه پایتون مناسب استفاده کنید تا کدهای هسته جنگو به‌درستی کار کنند. چون این کدها وابسته به کتابخانه‌ها، توابع و کلاس‌های پایتون هستند، و استفاده از نسخه نامناسب پایتون موجب برهم زدن عملکرد جنگو می‌شود.

باتوجه به تصویر صفحه بعد و از آنجایی‌که محتوای آموزشی این کتاب بر پایه نسخه‌های 3.2.x، 4.x و 5.x جنگو است، استفاده از نسخه 3.8.x، 3.9.x یا 3.10.x پایتون پیشنهاد می‌شود.

یادآوری: LTS (Long Term Service) چیست؟ گونه‌ای از نسخه‌های خاص از یک محصول نرم‌افزاری است که برای مدت زمان طولانی‌تری نسبت به سایر نسخه‌های ساده پشتیبانی می‌شود. ارائه نسخه LTS برای پروژه‌ها و نرم‌افزارهای متن‌باز بسیار پرکاربرد و مرسوم است.

What Python version can I use with Django?

Django version	Python versions
1.11	2.7, 3.4, 3.5, 3.6, 3.7 (added in 1.11.17)
2.0	3.4, 3.5, 3.6, 3.7
2.1	3.5, 3.6, 3.7
2.2	3.5, 3.6, 3.7, 3.8 (added in 2.2.8), 3.9 (added in 2.2.17)
3.0	3.6, 3.7, 3.8, 3.9 (added in 3.0.11)
3.1	3.6, 3.7, 3.8, 3.9 (added in 3.1.3)
3.2	3.6, 3.7, 3.8, 3.9
4.0	3.8, 3.9, 3.10

دقت داشته باشید اگر توسعه برنامه‌های تحت‌وب به‌درستی انجام نپذیرد می‌تواند تکراری، خسته‌کننده و آزاردهنده باشد. فریم‌ورک جنگو تلاش کرده برای شما شرایطی فراهم کند تا روی موضوعاتی خارج از دوباره‌نویسی بخش‌های تکراری تمرکز کنید (DRY + Out-of-Box). در فصل‌های دوم تا نهم کتاب تلاش شده با مفاهیم قاعده DRY و Out-of-Box جنگو بیشتر آشنا شوید. ♥♥

۱-۲ معماری فریم‌ورک جنگو

فریم‌ورک جنگو برپایه معماری MVC (Model-View-Controller) پیاده‌سازی شده است. MVC یک روش برای توسعه نرم‌افزار است که در آن نحوه‌ی تعریف‌کردن و دسترسی به داده‌ها (Model) از منطق برنامه یا همان کنترل‌کننده (Controller) و آن نیز از چگونگی ایجاد ارتباط (View)، به‌صورت کاملاً مجزا پیاده‌سازی می‌شود. در معماری MVC هر یک از بخش‌های برنامه بطور مستقل کار می‌کنند و در صورت نیاز به اعمال تغییر، هر یک از این بخش‌ها بدون تأثیرگذاری روی سایر قسمت‌ها براحتی تغییر می‌یابند.

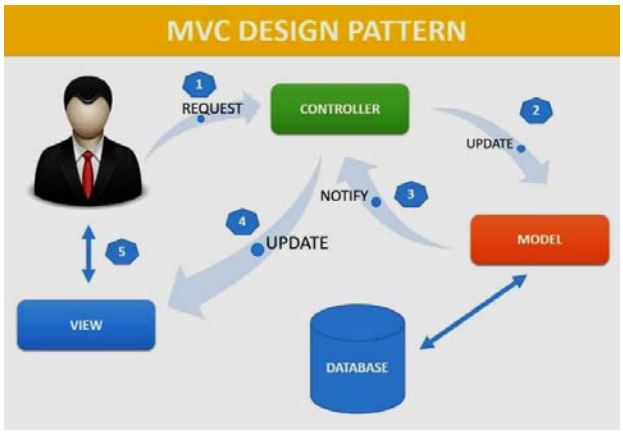
الگوی معماری MVC اختصاصی جنگو، MVT (Model-View-Template) نام دارد و به سه مؤلفه (Component) زیر تقسیم می‌شود:

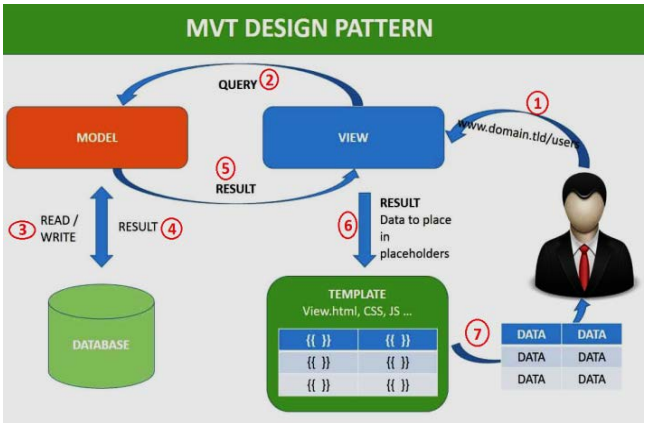
الف- Model: از مدل‌ها در جنگو برای ایجاد ارتباط با دیتابیس استفاده می‌شود. زمانی‌که یک مدل در محیط جنگو تعریف می‌کنید، جنگو بطور خودکار در دیتابیس، یک جدول برای آن مدل می‌سازد.

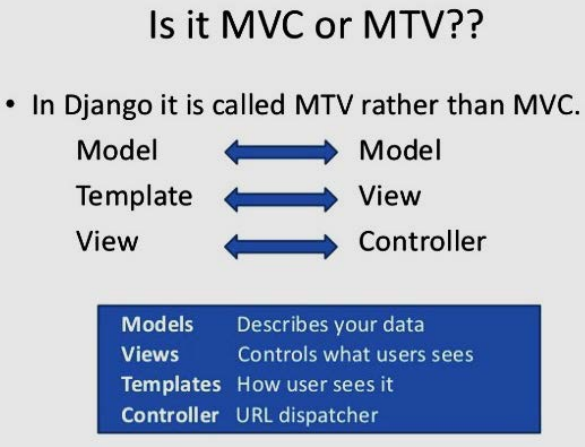
ب- Template: یا همان قالب، معمولاً یک فایل HTML است که برای نمایش داده‌ها در مرورگر کاربر استفاده می‌شود. جنگو یک *قالب زبان* بسیار قوی به‌نام DTL (Django Template Language) دارد که در فصل‌های بعدی با مفهوم و چگونگی کار با آن بیشتر آشنا خواهید شد.

پ- View: وظیفه نماها (views) ایجاد ارتباط میان قالب‌ها (templates) و مدل‌ها (models) است. یک نما در یک اپلیکیشن جنگو تصمیم می‌گیرد که چه داده‌هایی باید از دیتابیس به قالب تحویل داده شوند یا چه کارهایی نیاز است روی داده‌های ورودی سمت کاربر انجام شود. نماهای جنگو به دو شکل تابعی (Function-based) و کلاسی (Class-based) تعریف می‌شوند؛ و در صورت نیاز پروژه می‌توانند با قالب‌های جنگو نیز در ارتباط باشند.

تفاوت اصلی میان این دو معماری این است که در نوع MVT، خود جنگو مدیریت مؤلفه Controller را برعهده می‌گیرد (کنترل‌کننده یک کد نرم‌افزاری است که تعامل میان مؤلفه‌های Model و View را مدیریت می‌کند). برای درک بهتر مفاهیم این بخش، بادقت به تصاویر زیر توجه کنید:



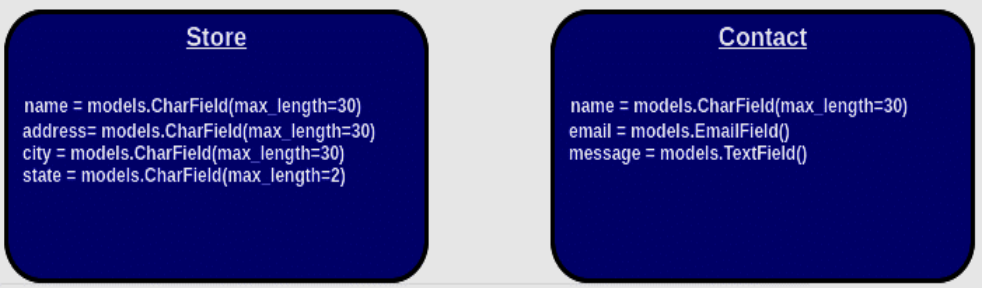




۱-۲-۱ قاعده "خودت را تکرار نکن" (Don't Repeat Yourself)

قاعده DRY به‌طور واضح به موضوعات ارث‌بری، کاهش کارهای تکراری و صرفه‌جویی در زمان کدنویسی در جنگو اشاره دارد. برای درک بهتر قاعده DRY به این‌مثال توجه کنید: فرض کنید می‌خواهیم یک وب اپلیکیشن فروشگاهی با موضوع "خانه قهوه" بسازیم که تعدادی شعبه در سطح کشور دارد و یک فرم "تماس با ما" برای مشاهده مشخصات تک‌تک شعبه‌ها و ارسال پیام مستقیم به مدیریت هر شعبه، در سایت coffee-house.com دارد.

در ابتدا باید بررسی کنیم چه داده‌هایی قرار است در دیتابیس ذخیره شوند و چه اطلاعاتی قرار است در صفحه HTML به مشتری نمایش داده شوند. فرم "تماس با ما" نیاز به داشتن تعدادی تکست‌باکس برای نام مشتری، آدرس ایمیل، متن پیام و... دارد. فرم "نمایش اطلاعات شعبه" هم نیاز به داشتن تعدادی تکست‌باکس برای کلید یکتا به‌منظور تفکیک فروشگاه‌ها از یکدیگر در دیتایس، نام فروشگاه، نام شهر، نام استان، آدرس و... دارد. پس برای این‌منظور، دو مدل زیر باید در فایل models.py تعریف شوند:



در هر یک از مدل‌های Store و Contact سـه متغیر وجود دارد که همان فیلدهای دو جدول Store و Contact در دیتابیس هستند. برای فیلد name یک نوع‌داده(data-type) از جنس Char و حداکثر کارکتر مجاز قابل ذخیره‌سازی در این فیلد را برابر 30 تعریف کردیم. برای فیلد email هم یک نوع‌داده از جنس Email تعریف کردیم تا جنگو به‌طور خودکار بحث اعتبارسنجی(Validation) ایمیل بودن یا نبودن مقدار وارده را انجام دهد (اگر نوع‌داده Char تعریف می‌شد باید خودمان بصورت دستی یک اعتبارسنج برای ممیزی داده ورودی می‌نوشتیم که زمان‌بر بود[۱]). سپس در ادامه، فیلدهای address، city، state، message و... را برای این دو مدل تعریف کردیم.

اگر از مفهوم قاعده DRY جنگو استفاده نمی‌شد، می‌بایست گام‌های زیر را خودمان انجام می‌دادیم و کدنویسی موارد زیر ممکن بود با خطای انسانی و اتلاف وقت در فرآیند دیباگ همراه باشد:

- ایجاد یک دیتابیس رابطه‌ای به همراه دو جدول برای ذخیره‌سازی اطلاعات.
- ایجاد یک لایه Business-Logic (برای پردازش و اعتبارسنجی داده‌ها).
- ایجاد فرم‌های HTML متعدد برای ثبت، ارسال و نمایش داده‌های فرم.
- ایجاد یک رابط کاربری برای مدیران سایت، جهت دسترسی به داده‌های درون دیتابیس.

موارد اشاره شده در متن بالا، از نوشتن کدهای HTML برای نمایش اطلاعات در مرورگر گرفته تا طراحی جداول دیتابیس، با نوشتن چند خط کد ساده، براحتی و به‌صورت خودکار توسط جنگو انجام می‌شود و در فصل‌های بعدی کتاب با قدرت قاعده DRY بیشتر آشنا خواهید شد.

---

[۱] سود نبردن از قاعده DRY جنگو

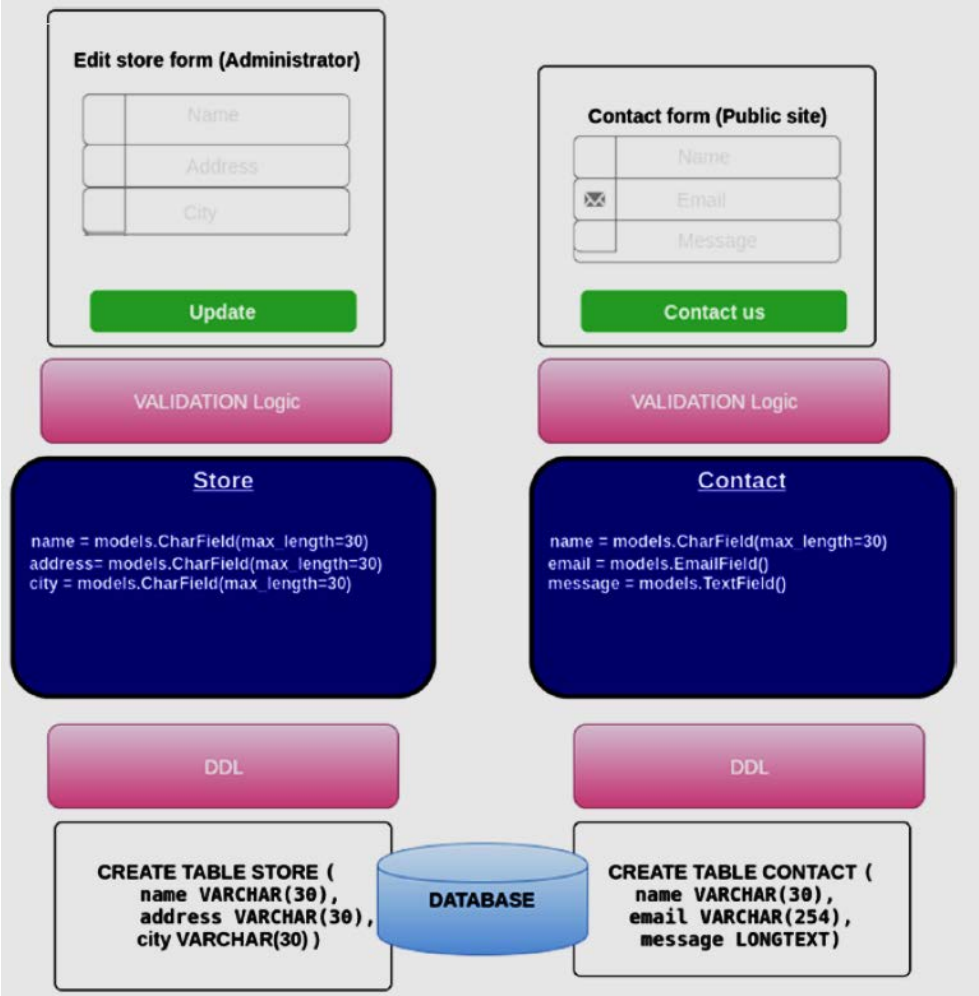


Figure 1-2. *Django models create separate constructs based on DRY principle*

۱-۳ نصب و راه‌اندازی جنگو در لینوکس و ویندوز

روش‌های گوناگونی برای نصب جنگو در سیستم‌عامل‌های لینوکس و ویندوز وجود دارد:

- ✓ دانلود و نصب فریم‌ورک جنگو از طریق سایت‌رسمی جنگو به‌نشانی زیر:

https://www.djangoproject.com/download/

- ✓ دانلود و نصب در محیط‌های لینوکسی با کمک ابزار apt-get.

- ✓ دانلود و نصب با کمک ابزار کنترل‌گر بسته‌های پایتون به‌نام pip.
- ✓ دریافت به‌روز آخرین تغییرات و نسخه‌ها (مانند آلفا و بتا) از طریق مخزن گیت‌هاب پروژه به‌نشانی: https://github.com/django/django/.

۱-۳-۱ به‌روز رسانی و نصب pip

باتوجه به اینکه محتوای آموزشی این‌کتاب برپایه پایتون۳ است و ابزار pip به‌طور پیش‌گزیده در فایل نصبی این‌نسخه گنجانده شده است، تنها کاری که باید انجام داد ارتقاء ابزار pip است:

Windows (Command Prompt or PowerShell):

```
C:\Users\linux10> pip install pip --upgrade
```

Linux Terminal:

```
[root@milani]$ pip install pip --upgrade
```

Output:

```
Collecting pip
      Downloading pip-21.0.1-py2.py3-none-any.whl (5.3MB)
Installing collected packages: pip
      Found existing installation: pip 20.1.1
Successfully installed pip-21.0.1
Requirement already satisfied: pip in c:\python38\lib\site-packages (21.0.1)
```

Windows (Command Prompt or PowerShell):

```
C:\Users\linux10> python -V      ## Capital Viii
```

Linux Terminal:

```
[root@milani]$ python -V
```

Output:

```
Python 3.8.6
```

۱-۳-۲ نصب کتابخانه جنگو

یکی از روش‌های آسان نصب جنگو در سیستم، استفاده از ابزار **pip** است. مثال:

Windows (Command Prompt or PowerShell):

```
C:\Users\linux10> pip install django==3.2.5
```

Linux Terminal:

```
[root@milani]$ pip install django==3.2.5
```

Output:

```
Collecting django==3.2.5
        Using cached Django-3.2.5-py3-none-any.whl (7.8 MB)
Requirement already satisfied: sqlparse>=0.2.2 in c:\python38\lib\site-packages (from django==3.2.5) (0.4.1)
Requirement already satisfied: asgiref<4,>=3.2.10 in c:\python38\lib\site-packages (from django==3.2.5) (3.3.1)
Requirement already satisfied: pytz in c:\python38\lib\site-packages (from django==3.2.5) (2021.5)
Installing collected packages: django
        Attempting uninstall: django
                Found existing installation: Django 3.1.6
                Uninstalling Django-3.1.6:
                        Successfully uninstalled Django-3.1.6
Successfully installed django-3.2.5
```

برای ارتقاء و نصب آخرین نسخه ارائه شده هر کتابخانه پایتونی می‌توانید از پارامتر upgrade-- در دستور pip استفاده کنید:

Windows (Command Prompt or PowerShell):

```
C:\Users\linux10> pip install django --upgrade
```

Linux Terminal:

```
[root@milani]$ pip install django --upgrade
```

اگر تمایل به نصب جنگو از طریق سایت‌رسمی جنگو به نشانی DjangoProject.com دارید، پس‌از دانلود فایل با پسوند tar.gz. می‌توانید به‌صورت زیر اقدام به نصب جنگو نمایید:

Windows (Command Prompt or PowerShell):

```
C:\Users\linux10\Desktop> pip install Django-3.2.5.tar.gz
```

Linux Terminal:

```
[root@milani]$ pip install /root/Downloads/Django-3.2.5.tar.gz
```

Output:

```
Processing /root/Downloads/Django-3.2.5.tar.gz
Collecting pytz (from Django==3.2.5)
        Using cached pytz-2021.2-py3-none-any.whl
Building wheels for collected packages: Django
        Running setup.py bdist_wheel for Django ... done
        Stored in directory: /home/debian/.cache/pip/wheels/56/bf/24/
f44162e115f4fe0cfeb4b0ae99b570fb55a741a8d090c9894d
Successfully built Django
Installing collected packages: pytz, Django
Successfully installed Django-3.2.5pytz-2021.2
```

برای نصب جنگو از طریق سایت گیت‌هاب(Github.com) به دو روش می‌توان عمل کرد: دانلود و نصب خودکار جنگو به‌صورت از راه‌دور؛ و یا کپی کامل مخزن پروژه روی سیستم و نصب در زمان دلخواه.

نکته: پیش‌از اجرای دستورات زیر، حتماً از نصب‌بودن برنامه Git در سیستم‌عامل اطمنیان حاصل کنید. برای دانلود این نرم‌افزار می‌توانید به‌نشانی https://git-scm رجوع نمایید.

Method 1)

Linux Terminal:

```
[root@milani]$ pip install git+https://github.com/django/django.git
```

Output

```
Collecting git+https://github.com/django/django.git
        Cloning https://github.com/django/django.git to ./pip-31j_bcqa-build
Requirement already satisfied: pytz in /python/mydjangosandbox/lib/python3.8/site-
packages
(from Django==3.0.dev20210408112615)
Installing collected packages: Django
                Successfully uninstalled Django-3.1.6
        Running setup.py install for Django ... done
        Successfully installed Django-3.2.5.dev20210408112615
```

Method 2)

Linux Terminal (Part 1):

```
[root@milani]$ git clone https://github.com/django/django.git
```

Output:

```
Cloning into django...
remote: Counting objects: 388550, done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 388550 (delta 5), reused 0 (delta 0), pack-reused 388531
Receiving objects: 100% (388550/388550), 158.63 MiB | 968.00 KiB/s, done.
Resolving deltas: 100% (281856/281856), done.
Checking connectivity... done.
```

Linux Terminal (Part 2):

```
## Assuming Django Git download made to /root/Downloads/django/

[root@milani]$ cd /root/Downloads/

[root@milani]$ pip install /django/
```

Output:

```
Processing /root/Downloads/django
Collecting pytz (from Django==3.0.dev20210408112615)
        Using cached pytz-3.2.5-py3-none-any.whl
Installing collected packages: pytz, Django
        Running setup.py install for Django ... done
Successfully installed Django-3.2.5.dev20210408112615 pytz-2021.2
```

۱-۳-۳ نصب محیط مجازی (Virtual Environment)

استفاده از محیط‌های_مجازی در توسعه برنامه‌های جنگو الزامی نیست. اما استفاده از این قابلیت باعث می‌شود برنامه شما در یک محیط کاملاً ایزوله و امن، تست و توسعه داده شود. نصب کتابخانه‌ها در این‌محیط هیچ تأثیری روی کتابخانه‌های نصب شده روی سیستم و بالعکس ندارد. به کمک محیط‌های_مجازی می‌توان چندین نسخه از جنگو روی سیستم‌عامل نصب داشت و بدون هیچ مشکلی روی پروژه‌های جدید و قدیمی کار کرد.

از پرکاربردترین روش‌های ساخت محیط_مجازی در پایتون، کتابخانه **virtualenv** نام دارد و به‌صورت زیر نصب و استفاده می‌گردد:

Windows (Command Prompt or PowerShell):

```
C:\Users\linux10\> pip install virtualenv
```

Linux Terminal:

```
[root@milani]$ pip install virtualenv
# OR:
[root@milani]$ pip install virtualenv --upgrade
```

Output:

```
Collecting virtualenv
        Downloading virtualenv-20.4.2-py2.py3-none-any.whl (7.2 MB)
        |……..| 7.2 MB 298 kB/s
Collecting appdirs<2,>=1.4.3
        Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Collecting distlib<1,>=0.3.1
        Downloading distlib-0.3.1-py2.py3-none-any.whl (335 kB)
        |……..| 335 kB 192 kB/s
Collecting filelock<4,>=3.0.0
        Downloading filelock-3.0.12-py3-none-any.whl (7.6 kB)
Requirement already satisfied: six<2,>=1.9.0 in c:\python38\lib\site-packages (from
virtualenv) (1.15.0)
Installing collected packages: filelock, distlib, appdirs, virtualenv
Successfully installed appdirs-1.4.4 distlib-0.3.1 filelock-3.0.12 virtualenv-20.4.2
```

به‌عنوان مثال: در سیستم‌عامل ویندوز، وارد پوشه Desktop می‌شویم و یک‌پوشه با نام **my_all_envs** می‌سازیم و درون این پوشه اقدام به ایجاد تعدادی محیط_مجازی می‌کنیم:

Syntax:

virtualenv --python=python3 **YourName**

Windows (Command Prompt or PowerShell):

C:\....\Desktop\my_all_envs> virtualenv --python=python3 **azimzadeh_env1**

C:\....\Desktop\my_all_envs> virtualenv --python=python3 **django3.2.5_env_coffe**

C:\....\Desktop\my_all_envs> virtualenv --python=python3 **pendareparsENV**

Linux Terminal:

[root@milani]$ virtualenv --python=python3 **Django_Sandbox_env**

Output:

```
created virtual environment CPython3.8.6.final.0-64 in 672ms
creator
CPython3Windows(dest=C:\Users\linux10\Desktop\my_all_envs\azimzadeh_env1,
clear=False, no_vcs_ignore=False, global=False)
seeder FromAppData(download=False, pip=bundle, setuptools=bundle,
wheel=bundle, via=copy,
app_data_dir=C:\Users\linux10\AppData\Local\pypa\virtualenv)
added seed packages: pip==21.0.1, setuptools==52.0.0, wheel==0.36.2
activators
BashActivator,BatchActivator,FishActivator,PowerShellActivator,PythonActivator,X
onshActivator
```

ابزار دوم، **venv** نام دارد که یک‌ماژول از پیش نصب‌شده در پایتون۳ است. نحوه‌ی کار با این ابزار به‌شکل زیر است:

Syntax:

python -m venv **YourName**

Windows (Command Prompt or PowerShell):

C:\.....\Desktop\my_all_envs> python -m venv **Azimzadeh**

Linux Terminal:

[root@milani]$ python -m venv **Django_Sandbox_env**

هر دو ابزار عملکرد پایه یکسانی دارند و تفاوت‌شان در پارامترهای تنظیمی تکمیلی است. چون در این‌کتاب نیازی به اعمال این تنظیمات تکمیلی نداریم، به‌شرح آن‌ها نمی‌پردازیم.

۱-۳-۳-۱ فعال‌سازی محیط مجازی در سیستم‌عامل ویندوز

ساختار درختی یک محیط_مجازی پایتون در سیستم‌عامل ویندوز به‌شکل زیر است:

```
C:\Users\linux10\Desktop\my_all_envs\azimzadeh_env1> tree
## OR:
C:\Users\linux10\Desktop\my_all_envs\azimzadeh_env1> tree /f
Folder PATH listing
├───Lib
│   └───site-packages
│       ├───pip
│       ├───pip-21.0.1.dist-info
│       ├───pkg_resources
│       ├───setuptools
│       ├───setuptools-52.0.0.dist-info
│       ├───wheel
│       └───_distutils_hack
└───Scripts
        activate
        activate.bat
        activate_this.py
        deactivate.bat
        easy_install-3.8.exe
        easy_install.exe
        pip-3.8.exe
        pip.exe
        pip3.8.exe
        pip3.exe
        python.exe
```

برای مدیریت محیط_مجازی از دو دستور **activate** و **deactivate** استفاده می‌شود. این دو ابزار در پوشه **Scripts** محیط_مجازی قرار دارند. زمانی‌که دستور activate به‌درستی اجرا شود، نامی که پیش‌تر برای محیط_مجازی انتخاب کرده‌اید، در ابتدای تمامی خطوط ترمینال ظاهر می‌شود. مثال:

```
C:\....\my_all_envs\azimzadeh_env1\Scripts> activate
(azimzadeh_env1) C:\....\my_all_envs\azimzadeh_env1\Scripts>
```

برای خروج از محیط_مجازی باید دستور deactivate را اجرا کرد. مثال:

(azimzadeh_env1) C:\....\my_all_envs\azimzadeh_env1\Scripts> **deactivate**
C:\....\my_all_envs\azimzadeh_env1\Scripts>

با آنکه ابزارهای محیط_مجازی در پوشه Scripts قرار دارند، اما می‌توان آن‌ها را در دیگر مسیرها نیز فراخوانی کرد. برای درک بهتر موضوع "فعال‌سازی"، یک مثال کاربردی در بخش "ساخت یک پروژه" خواهید دید.

۱-۳-۳-۲ فعال‌سازی محیط مجازی در سیستم‌عامل لینوکس

ساختار درختی یک محیط_مجازی پایتون در سیستم‌عامل لینوکس به‌شکل زیر است:

```
[root@milani]$ tree
├── bin
│   ├── activate
│   ├── activate.csh
│   ├── activate.fish
│   ├── Activate.ps1
│   ├── easy_install
│   ├── easy_install-3.8
│   ├── pip
│   ├── pip3
│   ├── pip3.8
│   ├── python -> python3
│   └── python3 -> /usr/bin/python3
├── include
├── lib
│   └── python3.8
│       └── site-packages
│           ├── easy_install.py
│           ├── pip
│           ├── pip-20.0.2.dist-info
│           ├── pkg_resources
│           ├── pkg_resources-0.0.0.dist-info
│           ├── setuptools
├── lib64 -> lib
├── pyvenv.cfg
└── share
```

برای مدیریت محیط_مجازی، از دو دستور **activate** و **deactivate** استفاده می‌شود. این دو ابزار در دایرکتوری **bin** محیط_مجازی قرار دارند. زمانی‌که دستور activate به‌درستی اجرا شود، نامی که پیش‌تر برای محیط_مجازی انتخاب کرده‌اید، در ابتدای تمامی خطوط ترمینال ظاهر می‌شود. به‌خاطر داشته باشید برای فراخوانی activate در شِل/پوسته جاری لینوکس از دستور **source** استفاده می‌شود. مثال:

```
milani@milani:~/Desktop$  python3 -m venv django_book
milani@milani:~/Desktop/django_book$ source bin/activate
(django_book) milani@milani:~/Desktop/django_book$
```

برای خروج از محیط_مجازی باید دستور deactivate را اجرا کرد. مثال:

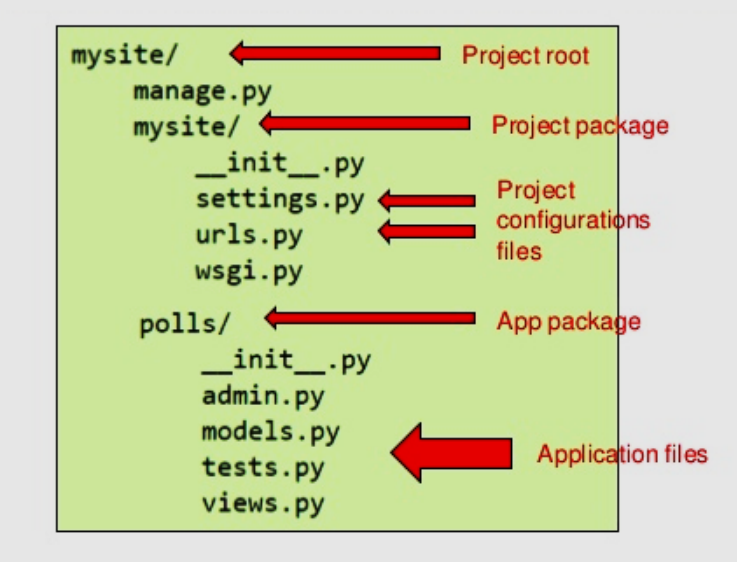
```
(django_book) milani@milani:~/Desktop/django_book$ deactivate
milani@milani:~/Desktop/django_book$
```

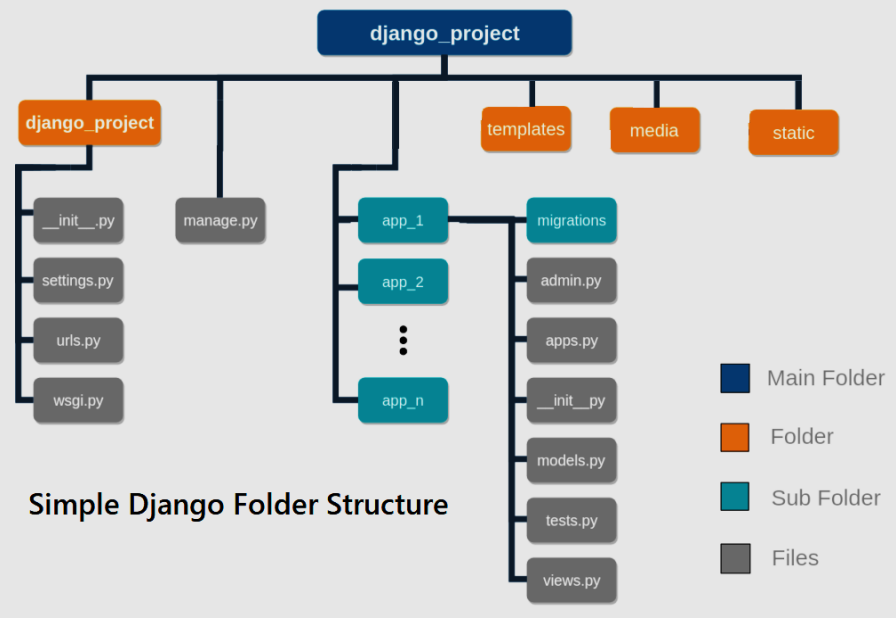
نکته مهم: به‌یاد داشته باشید که پیش‌نیاز تست و تمرین مثال‌های این کتاب، قرار داشتن در محیط_مجازی پایتون است.

۱-۴ شروع و ساخت یک پروژه

برای ساخت یک پروژه جدید باید از ابزار **django-admin** که در کتابخانه django قرار دارد استفاده کنیم. فایل‌های اجرایی(.exe) و اسکریپتی(.py) این ابزار که هر دو عملکرد یکسانی دارند، در مسیرهای سیستمی /usr/bin و /usr/local/bin؛ یا در مسیرهای /bin و /Scripts در محیط_مجازی در دسترس هستند. در فصل‌های بعد با دیگر آرگومان‌های ابزار django-admin به‌طور کامل آشنا خواهید شد. ساختار کلی پروژه‌های جنگو به‌شکل زیر است:

```
+<BASE_DIR_project_name>
|
+── manage.py
|
+──+<PROJECT_DIR_project_name>
     |
     __init__.py
     settings.py
     urls.py
     wsgi.py
```





در پوشه پروژه (Proj_Dir)، چهار فایل مهم وجود دارند که در ادامه به شرح آن‌ها می‌پردازیم:

- ✓ **فایل __init__.py**: این فایل توسط جنگو تولید می‌شود تا اطمینان حاصل شود مفسر پایتون، این پوشه را به عنوان یک بسته (Package) معتبر در نظر گرفته و برای مقاصد بسته‌سازی (packaging) در پروژه مورد استفاده قرار می‌دهد.

- ✓ **فایل settings.py**: تنظیمات کلان پروژه و اپلیکیشن‌ها در این فایل مدیریت می‌شوند.
- ✓ **فایل ursl.py**: مسیر/آدرس URL ریشه (root URL-Paths) تمامی اپلیکیشن‌های پروژه، به‌منظور فراخوانی مسیرها/آدرس‌های URL خود اپلیکیشن‌ها، در این فایل قرار می‌گیرند.
- ✓ **فایل wsgi.py**: اطلاعات لازم برای راه‌اندازی پروژه روی " Web Server Gateway Interface" در این فایل قرار دارد.

برای ساخت یک پروژه جنگو نیاز به نوشتن آرگومان **startproject** در ابزار django-admin داریم. پس در ابتدا وارد محیط_مجازی پایتون شده و با کمک دستور django-admin یک پروژه با نام coffehouse ایجاد می‌کنیم:

Windows (Part 1):

C:\...\my_all_envs\azimzadeh_env1\Scripts> **activate**

(azimzadeh_env1) C:\...\my_all_envs\azimzadeh_env1\Scripts> **pip install django==3.2.5**

(azimzadeh_env1) C:\...\my_all_envs\azimzadeh_env1\Scripts> **cd ..**

(azimzadeh_env1) C:\...\my_all_envs\azimzadeh_env1>

Linux (part 1):

milani@milani:~/Desktop/django_book$ **source bin/activate**

(django_book) milani@milani:~/Desktop/django_book$ **pip install django==3.2.5**

Syntax:

django-admin startproject **YourName**

Windows (Part 2):

C:\...\my_all_envs\azimzadeh_env1> django-admin startproject **coffehouse**

Linux (Part 2):

milani@milani:~/Desktop/django_book$ django-admin startproject **coffehouse**

با بررسی پوشه‌های ایجاد شده می‌توان مشاهده کرد که دو پوشه دقیقاً با نام coffehouse ساخته شده است. اولین پوشه، پوشه_پایه (Base Directory) نام دارد که حاوی فایل manage.py است و برای مدیریت کلان پروژه استفاده می‌شود (اپلیکیشن‌ها در همین پوشه_پایه باید ایجاد شوند). البته نام این پوشه را می‌توان تغییر داد. از پوشه دوم برای مدیریت اپلیکیشن‌های پروژه استفاده می‌گردد و به‌هیچ‌وجه نباید نام این پوشه را تغییر داد.

خوشبختانه فریم‌ورک جنگو برای توسعه، تست و مشاهده خروجی اپلیکیشن‌ها در مرورگر، به یک وب‌سرور سبک و ساده‌ی توکار(built-in) مجهز است. به‌منظور اجرای این وب‌سرور توکار باید از آرگومان **runserver** در اسکریپت manage.py استفاده کرد. مقدار پیش‌گزیده این آرگومان برابر 127.0.0.1:8000 است. مثال:

Syntax:

```
python manage.py runserver
## OR:
python manage.py runserver IP:Port
python manage.py runserver 192.168.1.2:8008
```

Windows (Command Prompt or PowerShell):

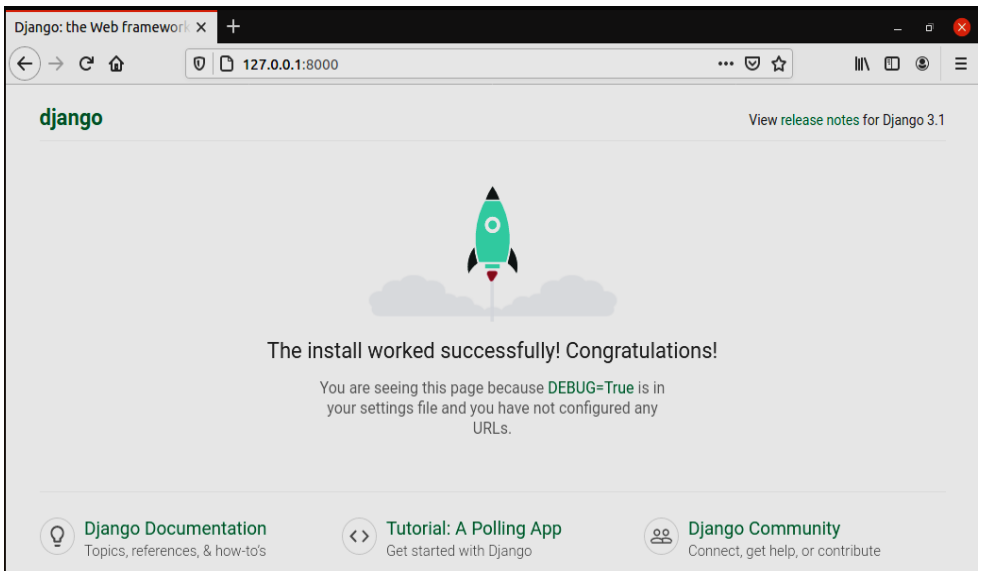
```
(azimzadeh_env1) C:\....\coffehouse>python manage.py runserver
```

Linux Terminal:

```
(django_book) milani@milani:~/..../coffehouse$ python manage.py  runserver
```

Output:

```
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).  ←←←
Run 'python manage.py migrate' to apply them.
March 13, 2021 - 05:57:13
Django version 3.2.5, using settings 'coffehouse.settings'
Starting development server at http://127.0.0.1:8000/  ←←←
Quit the server with CONTROL-C (Ctrl+C).  ←←←
```



۱-۵ نصب و راه‌اندازی دیتابیس برای پروژه

جنگو به‌طور پیش‌گزیده از دیتابیس فایلی SQLite استفاده می‌کند و برای کار با آن نیاز به نصب برنامه جدیدی نیست. همچنین جنگو از موتورهای دیتابیس (Database Engine) گوناگونی پشتیبانی می‌کند که مهم‌ترین آن‌ها عبارتند از:

- ✓ دیتابیس MySQL
- ✓ دیتابیس MariaDB
- ✓ دیتابیس PostgreSQL
- ✓ دیتابیس SQLite3
- ✓ دیتابیس Oracle
- ✓ دیتابیس MongoDB

تنظیمات مربوط به دیتابیس(ها)، در یک متغیر از نوع دیکشنری به‌نام **DATABASES** در فایل settings.py ذخیره شده است. درحال‌حاضر نیازی به اعمال تغییرات در متغیر DATABASES نداریم و از قابلیت‌های از پیش تنظیم‌شده آن استفاده خواهیم کرد. اما در ادامه این فصل به توضیح کامل آن خواهیم پرداخت. ساختار کلی این متغیر به‌شکل زیر است:

```
DATABASES = {
        'default': {
                'ENGINE': ' ',
                'NAME': ' ',
                'USER': ' ',
                'PASSWORD': ' ',
                'HOST': ' ',
                'PORT': ' ',
                'ATOMIC_REQUESTS': False,
                'AUTOCOMMIT': True,
                'CONN_MAX_AGE': 0,
                'CHARSET': ' ',
                'OPTIONS': {},
        }
}
```

برای کلید ENGINE در این متغیر، تنها می‌توان یکی از مقادیر زیر را با توجه به نوع دیتابیس درج نمود:

- ✓ برای دیتابیس MySQL: مقدار django.db.backends.mysql
- ✓ برای دیتابیس MariaDB: مقدار django.db.backends.mysql
- ✓ برای دیتابیس PostgreSQL: مقدار django.db.backends.postgresql_psycopg2
- ✓ برای دیتابیس SQLite3: مقدار django.db.backends.sqlite3
- ✓ برای دیتابیس Oracle: مقدار django.db.backends.oracle

در ادامه به‌شرح کلیدهای مهم متغیر DATABASES می‌پردازیم:

- **ATOMIC_REQUESTS**: مقدار پیش‌گزیده این کلید False است. اگر True تنظیم شود، جنگو هر درخواست را مجبور می‌کند در قالب یک تراکنش اجرا گردد (یعنی از لحظه فراخوانی نما در فایل views.py تا لحظه به اتمام رسیدن نما). اگر درحین پردازش درخواست، خطا یا استثنائی رخ ندهد، جنگو آن تراکنش را commit می‌کند؛ وگرنه آن تراکنش را rollback می‌کند. دقت داشته باشید که فعال‌بودن این کلید روی کارائی و راندمان پروژه، سربار می‌افزاید.
- **AUTOCOMMIT**: مقدار پیش‌گزیده این کلید True است تا کنترل commit شدن یا نشدن درخواست توسط دیتابیس به‌طور خودکار مدیریت گردد. اگر False تنظیم شود، مدیریت commit تراکنش‌ها برعهده شما خواهد بود.

برای آشنایی بیشتر با موضوع تراکنش‌ها در جنگو، به نشانی docs.djangoproject.com/en/3.2/topics/db/transactions/ رجوع نمایید.

- **CONN_MAX_AGE**: از این کلید برای مشخص‌کردن مدت‌زمان برقرار بودن لینک‌ارتباطی (connection) با دیتابیس استفاده می‌شود. مقدار پیش‌گزیده این کلید، عدد صفر است که به‌معنای خاتمه سریع ارتباط، پس‌از پردازش هر درخواست است. برای ایجاد لینک‌ارتباطی نامحدود و دائمی می‌توان از مقدار None استفاده کرد.
- **HOST**: اگر مقدار این کلید خالی باشد، جنگو مقدار پیش‌گزیده‌ی localhost را درج می‌کند. اگر می‌خواهید نوع لینک‌ارتباطی با دیتابیس به‌شکل سوکت باشد (مانند MySQL) باید مقدار '/var/run/mysql' درج شود.
- **NAME**: این کلید حاوی نام دیتابیس است. برای SQLite باید مسیر کامل فایل درج شود. مثال: 'NAME': 'C:/www/store/db.sqlite3'
- **OPTIONS**: از این کلید برای تنظیم و ارسال پارامترهای تکمیلی در زمان اتصال به دیتابیس استفاده می‌شود. مقدار پیش‌گزیده این کلید یک دیکشنری خالی است.
- **PASSWORD**: از این کلید به‌منظور تنظیم رمزعبور برای اتصال به دیتابیس استفاده می‌شود.
- **PORT**: از این کلید به‌منظور تنظیم شماره پورت برای اتصال به دیتابیس استفاده می‌شود.
- **USER**: از این کلید به‌منظور تنظیم نام کاربری برای اتصال به دیتابیس استفاده می‌شود.
- **CHARSET**: از این کلید برای انتخاب نوع رمزگذار کارکترهای (CharacterSet Encoding) دیتابیس استفاده می‌شود. مثال: 'CHARSET':'utf8mb4'.

در سایت زیر می‌توانید Charsetهای تحت پوشش دیتابیس MySQL را مشاهده کنید:

https://dev.mysql.com/doc/refman/5.7/en/charset-charsets.html

برای مشاهده لیست کامل کلیدها و پارامترهای متغیر DATABASES به نشانی زیر رجوع نمایید:

https://docs..com/en/3.2/ref/settings/#std:setting-DATABASES

به‌خاطر داشته باشید جنگو به‌طور پیش‌گزیده، از دیتابیس تعریف‌شده در پارامتر **default** استفاده می‌کند. البته می‌توانید چندین دیتابیس دیگر نیز تعریف کنید. به‌عنوان نمونه:

```
DATABASES = {
        'default': {
```

```
        'NAME': 'app_data_db',
        'ENGINE': 'django.db.backends.postgresql',
        'USER': 'postgres_user',
        'PASSWORD': 'SecretPassWorD',
        'HOST': '127.0.0.1',
        'PORT': '5432',
    },
    'users': {
        'NAME': 'user_data_db',
        'ENGINE': 'django.db.backends.mysql',
        'USER': 'mysql_user',
        'PASSWORD': 'priv4te+98',
        'HOST': '190.150.100.100',
        'PORT': '3306',
    }
}
```

پارامتر default به‌هیچ وجه نباید حذف شود. اما محتوای آن‌را می‌توان خالی تنظیم کرد. مثال:

```
DATABASES = {
    'default': {},
    'auth_db': {
        'NAME': 'auth_db_name',
        'ENGINE': 'django.db.backends.mysql',
        'USER': 'azimzadeh',
        'PASSWORD': 'AzimZ@dehM!L@Ni',
    },
    'primary': {
        'NAME': 'primary_name',
        'ENGINE': 'django.db.backends.mysql',
        'USER': 'mysql_user',
        'PASSWORD': 'spam2021spam',
    },
    'replica1': {
        'NAME': 'replica1_tehran_datacenter',
        'ENGINE': 'django.db.backends.mysql',
        'USER': 'mysql_user',
```

```
        'PASSWORD': 'eggs1111',
    },
    'replica2': {
        'NAME': 'replica2_karaj_datacenter',
        'ENGINE': 'django.db.backends.mysql',
        'USER': 'mysql_user',
        'PASSWORD': 'eggs2222',
    },
}
```

نگران نحوه‌ی استفاده از نمونه تنظیمات بالا نباشید، چون در فصل "مدل‌ها در جنگو"، پیرامون مسیریابی و لینک کردن مدل‌ها به دیتابیس‌ها به‌طور کامل صحبت خواهیم کرد.

پس از انتخاب نوع دیتابیس، باید بسته‌های مورد نیاز آن دیتابیس را در محیط_مجازی هم نصب کرد تا جنگو در پشت صحنه قادر به برقراری لینک ارتباطی با دیتابیس باشد. حتماً از نصب بودن بسته‌های پایتونی زیر در محیط_مجازی اطمینان حاصل کنید:

- ✓ دیتابیس MySQL (mysql-python): pip install PyMySQL و pip install mysqlclient
- ✓ دیتابیس MariaDB (mysql-python): pip install PyMySQL و pip install mysqlclient
- ✓ دیتابیس PostgreSQL (psycopg2): pip install psycopg2
- ✓ دیتابیس Oracle: pip install cx_Oracle

در پایان باید از تنظیمات اعمال‌شده در فایل settings.py اطمینان حاصل کنیم. روش‌های متعددی برای تست برقرار بودن لینک ارتباطی با دیتابیس وجود دارند. اما یکی از بهترین‌ها، استفاده از دستور **migrate** در اسکریپت manage.py در پوشه BASE_DIR پروژه است. حتماً دستور زیر را اجرا کرده تا جداول زیرساختی و مهم پروژه جنگو در دیتابیس ساخته شوند:

در فصل بعدی، با مفهوم مهاجرت (migrations) و دستورات آن در جنگو بیشتر آشنا خواهید شد.

Linux or Windows:

(azimzadeh_env1) C:\....\azimzadeh_env1\coffehouse> **python manage.py makemigrations**

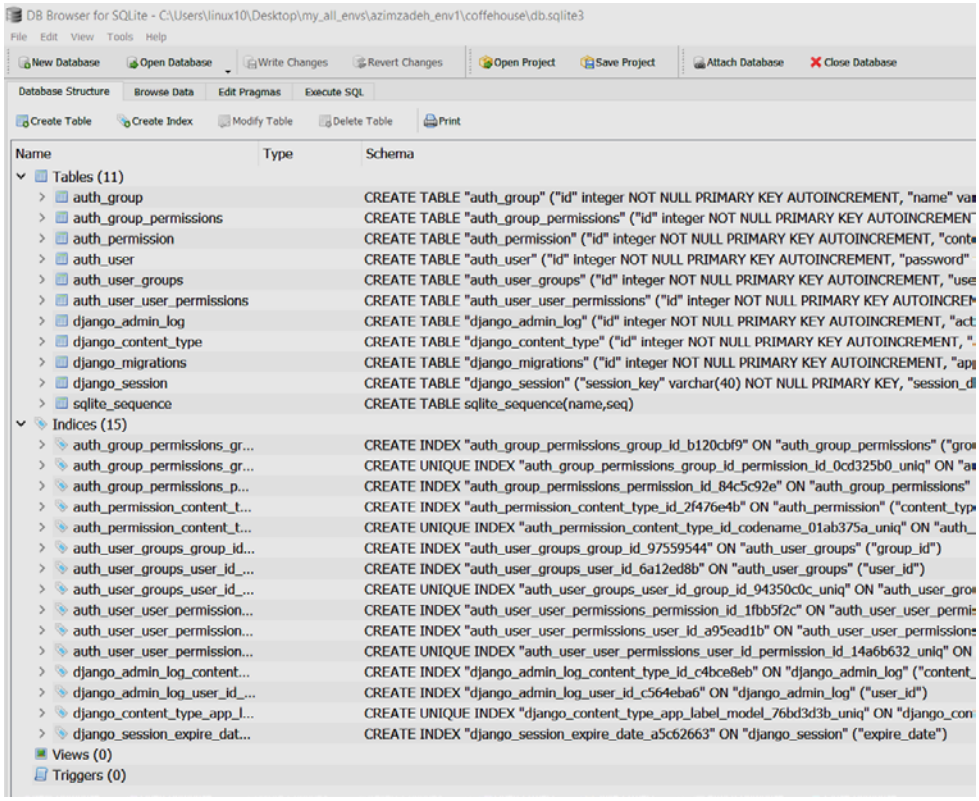
(azimzadeh_env1) C:\....\azimzadeh_env1\coffehouse> **python manage.py migrate**

Output:

```
Operations to perform:
    Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
    Applying contenttypes.0001_initial... OK
    Applying auth.0001_initial... OK
    Applying admin.0001_initial... OK
    Applying admin.0002_logentry_remove_auto_add... OK
    Applying admin.0003_logentry_add_action_flag_choices... OK
    Applying contenttypes.0002_remove_content_type_name... OK
    Applying auth.0002_alter_permission_name_max_length... OK
    Applying auth.0003_alter_user_email_max_length... OK
    Applying auth.0004_alter_user_username_opts... OK
    Applying auth.0005_alter_user_last_login_null... OK
    Applying auth.0006_require_contenttypes_0002... OK
    Applying auth.0007_alter_validators_add_error_messages... OK
    Applying auth.0008_alter_user_username_max_length... OK
    Applying auth.0009_alter_user_last_name_max_length... OK
    Applying auth.0010_alter_group_name_max_length... OK
    Applying auth.0011_update_proxy_permissions... OK
    Applying auth.0012_alter_user_first_name_max_length... OK
    Applying sessions.0001_initial... OK
```

وظیفه دستور migrate تبدیل‌کردن هر آنچه که باید در قالب جدول، فیلد، داده و... در دیتابیس ذخیره شود است. در فصل‌های بعد با کاربردهای عملیاتی این دستور بیشتر آشنا خواهید شد.

از آنجایی‌که جنگو از یکسری جداول و داده‌های از پیش تعریف شده برای مدیریت آسان‌تر پروژه استفاده می‌کند، اجرای دستور "python manage.py migrate" اجباری است. اکنون اگر به مسیر پوشه BASE_DIR پروژه بروید، یک دیتابیس فایلی از نوع SQLite خواهید دید. با کمک نرم‌افزارهای SqliteBrowser و Aryson-SQLite-Viewer و سایت‌های آنلاین مانند dbfopener.com و sqliteonline.com می‌توانید محتوای دیتابیس‌های SQLite را به‌راحتی مشاهده کنید:



۱-۶ تنظیم سریع انتشار یک محتوا (Quick Start)

به‌منظور آشنایی مقدماتی شما عزیزان با مفاهیم پایه جنگو (مدل‌ها، نماها، قالب‌ها، مسیرهای URL و...)، در این بخش می‌خواهیم یک پروژه ساده به‌روش شروع سریع (Quick Start) با فریم‌ورک جنگو، تجربه و پیاده‌سازی کنیم.

اکنون سناریوی ساده زیر را در نظر بگیرید که کاربر یک آدرس URL در مرورگر وارد می‌کند و جنگو باید برای آن مسیر/منبع درخواستی، یک محتوای نمایشی (demo) در یک فایل html، در صفحه مرورگر به کاربر نمایش دهد:

۱. در ابتدا، کاربر آدرس یا منبع درخواستی خود را در مرورگر وارد می‌کند:

http://azimzadeh-milani.com/**homepage**
http://127.0.0.1:8000/**homepage**

۲. در این گام، جنگو منبع درخواست شده در درخواست (request) کاربر را با الگوهای URL / مسیرهای URL که در فایل urls.py پروژه قرار دارند تطبیق می‌دهد (چون هر پروژه می‌تواند شامل چندین اپلیکیشن باشد).

۳. پس‌از مشخص شدن آنکه کدام اپلیکیشن باید فراخوانی شود، آن درخواست به‌سوی اپلیکیشن مربوطه هدایت می‌شود. این‌بار آدرس/منبع درخواست شده، با الگوهای URL / مسیرهای URL که در فایل urls.py اپلیکیشن قرار دارند تطبیق داده می‌شود.

نکته: به مسیرهای URLیی که درون متدهای ()path و ()re_path نوشته می‌شوند را "الگو یا الگوی تطبیق‌دهنده (URL Pattern)" می‌نامند.

٤. از آنجایی‌که هر مسیر URL باید به یک نمای جنگو نگاشت (mapping) شود، اکنون نمای تطبیق داده شده [مثلاً ()show_home_page] در اپلیکیشن مربوطه (مثلاً coffe_app) فراخوانی می‌شود.

٥. منطق (logic code) نمای ()show_home_page چون پیش‌تر توسط شما در فایل views.py اپلیکیشن پیاده‌سازی شده است، این نما پس‌از پردازش درخواست (request)، نتایج را درون قالب (template) جنگو (مثلاً homepage.html) درج می‌کند و سپس محتوای این قالب HTML را در صفحه مرورگر به کاربر نمایش می‌دهد.

برای تجربه شروع سریع (Quick Start) سناریوی بالا، مراحل زیر را با دقت انجام دهید:

الف– ابتدا وارد فایل urls.py پروژه در مسیر زیر شوید:

Note:

ProjectName: coffehouse AppName: coffe_app

Linux or Windows:

(azimzadeh_env1) ...\azimzadeh_env1> python manage.py startproject coffehouse

(azimzadeh_env1) ...\azimzadeh_env1> cd coffehouse\coffehouse

(azimzadeh_env1) ...\coffehouse\coffehouse> **gedit urls.py** ←← **Linux**

(azimzadeh_env1) ...\coffehouse\coffehouse> **notepad urls.py** ←← **Windows**

خط زیر را پس‌از خط **path('admin/', admin.site.urls),** در فایل urls.py پروژه درج کنید:

path ('', include('coffe_app.urls')),

کد نهایی:

azimzadeh_env1\coffehouse\coffehouse\urls.py:

```
from django.contrib import admin
from django.urls import path, re_path, include
urlpatterns = [
        path ('admin/', admin.site.urls),
        path ('', include('coffe_app.urls')),
]
```

پس‌از انجام دستورالعمل گام‌های **الف** تا **ز**، اگر کاربر آدرس http://127.0.0.1:8000/admin/ را در مرورگر وارد کند باید محتوای اپلیکیشن admin فراخوانی شود و اگر کاربر آدرس http://127.0.0.1:8000/homepage را وارد کند باید محتوای اپلیکیشن coffe_app فراخوانی شود.

از متغیر **urlpatterns** که یک نوع‌داده (DataType) از جنس List پایتون در جنگو است، برای تعریف مسیرهای URL (بواسطه متدهای path() یا re_path() در بسته django.urls) و هدایت درخواست‌ها (requests) به‌سمت اپلیکیشن‌ها استفاده می‌گردد. با کمک متد include() در بسته django.urls می‌توان مسیرهای URL یک اپلیکیشن را در فایل urls.py پروژه فراخوانی کرد.

نکته: دقت داشته باشید در انتهای هر خط، به ازای تعریف هر الگوی مسیر (URL Pattern) باید یک کارکتر کاما/ویرگول (,) درج شود.

ب- از آنجایی‌که نام اپلیکیشن را **coffe_app** انتخاب کردیم، اکنون با کمک آرگومان **startapp** در اسکریپت manage.py می‌توانیم این اپلیکیشن را در پروژه coffehouse جنگو ایجاد کنیم:

Linux or Windows:

(azimzadeh_env1) ...\coffehouse> **python manage.py startapp coffe_app**

اکنون در پوشه BASE_DIR پروژه، دو پوشه به نام‌های coffehouse و coffe_app می‌بینید. درون پوشه coffe_app تعدادی فایل مهم وجود دارند که به شرح آن‌ها می‌پردازیم:

- ✓ **فایل __init__.py**: این فایل توسط فریم‌ورک جنگو تولید می‌شود تا اطمینان حاصل شود مفسر پایتون، این پوشه را به عنوان یک بسته (Package) در نظر گرفته و برای مقاصد بسته‌سازی (packaging) در پروژه مورد استفاده قرار می‌دهد.
- ✓ **پوشه migrations**: با اجرای دستور "python manage.py migrate"، تعدادی فایل پایتونی در پوشه migrations ایجاد می‌شوند که از آن‌ها برای تبدیل مدل‌های جنگو (محتوای فایل models.py) به ساختارهای DDL (Database Definition Language) و DML (Data Manipulation Language) دیتابیس استفاده می‌شود.

- ✓ **فایل urls.py**: در این فایل، مسیرهای URL اپلیکیشن به‌منظور فراخوانی نماها یا قالب‌های جنگو درج می‌شوند. دقت داشته باشید که یک فایل urls.py در پوشه پروژه وجود دارد و یک فایل urls.py برای تک‌تک اپلیکیشن‌ها وجود دارد که شما باید این فایل را پس‌از ایجاد اپلیکیشن حتماً بسازید.
- ✓ **فایل admin.py**: از این فایل برای دسترسی و مدیریت تحت وب اشیاء (objects) یا همان نمونه داده‌های (instances) مدل‌ها استفاده می‌شود.
- ✓ **فایل apps.py**: از این فایل به‌منظور تنظیم پارامترهای اختیاری (options) در پیکربندی اپلیکیشن استفاده می‌شود.
- ✓ **فایل models.py**: از این فایل برای تعریف کلاس‌های مدل دیتابیس استفاده می‌شود.
- ✓ **فایل tests.py**: از این فایل برای تعریف تست‌ها (unit test-case) به‌منظور بررسی وضعیت کارکرد/رفتار اپلیکیشن استفاده می‌شود.
- ✓ **فایل views.py**: از این فایل برای تعریف نماها (view methods) یا همان توابع کنترل‌کننده رفتار اپلیکیشن استفاده می‌شود.

پ- اکنون در پوشه coffe_app باید یک فایل با نام urls.py بسازید و محتوای زیر را در آن درج نمایید:

azimzadeh_env1\coffehouse\coffe_app\urls.py:

```
from django.urls import path, re_path
from . import views
urlpatterns = [
        path ('homepage/', views.show_home_page),
]
```

در این فایل مشخص کردیم که اگر کاربری، منبع یا مسیر **homepage** را درخواست کرد، جنگو باید نمای ()show_home_page را از فایل views.py اپلیکیشن صدا زده و در آخر، یک پاسخ بازگرداند (مثلاً: نمایش محتوای قالب homepage.html در صفحه مرورگر).

د- برابر نمونه کد زیر می‌توان درخواست دریافتی را در نمای ()show_home_page پردازش کرد:

azimzadeh_env1\coffehouse\coffe_app\views.py:

```
from django.shortcuts import render
def show_home_page (request):
        ## do something here ....
```

```
    mycontent = {'mykey': 'We Love IRAN'}
    return render (request, "homepage.html", context=mycontent)
```

با کمک متد render() می‌توان برای درخواست (request) دریافتی، یک پاسخ (response) ایجاد کرده و محتوای یک قالب جنگویی (مانند html) را در صفحه مرورگر به کاربر نشان داد.

ت- اکنون زمان ساخت یک قالب HTML و درج محتوای دلخواه در آن است. برای این‌کار، وارد پوشه اپلیکیشن coffe_app شده و یک پوشه دقیقاً به‌نام **templates** ساخته و درون این پوشه، یک فایل html با نام homepage.html بسازید و محتوای زیر را در آن درج نمایید. مثال:

azimzadeh_env1\coffehouse\coffe_app\templates\homepage.html:

```
<html>
<head>
    <title>Django by Azimzadeh</title>
</head>
<body>
    <center>
        <h1>{{ mykey }}</h1>
        <br>
        <h2>Welcome to HomePage of coffe_app :)</h2>
        <br>
        <h3>You launched your first Django project &hearts;</h3>
        <br>
        <h3>Keep Calm and Learn Django &hearts;</h3>
    </center>
</body>
</html>
```

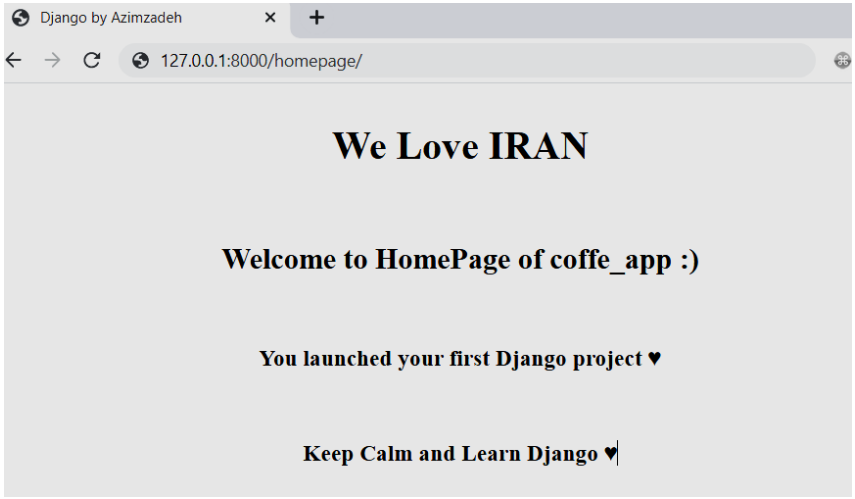
ث- در نهایت برای اینکه اپلیکیشن coffe_app به‌درستی کار کند، باید نام آن‌را در فایل settings.py پروژه، در متغیر INSTALLED_APPS درج کنیم:

```
INSTALLED_APPS = [
    'coffe_app',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
```

آیا به‌خاطر دارید پس‌از اجرای دستور "python manage.py migrate" تعدادی جدول در دیتابیس ساخته شد!؟ دلیلش، از پیش وجود داشتن نام این شش اپلیکیشن در متغیر INSTALLED_APPS است. چون جنگو بیشتر کارها را بدون آگاهی شما برایتان آسان کرده و خودش مدیریت می‌کند. پس هرگاه اپلیکیشنی ایجاد کردید حتماً نام آن‌را درون این متغیر درج کنید.

نکته: در انتهای نام هر اپلیکیشن باید یک کارکتر کاما/ویرگول (,) درج شود.

ج- با اجرای دستور "python manage.py runserver" و وارد کردن آدرس 127.0.0.1:8000/homepage/ در مرورگر، محتوای زیر نمایش داده خواهد شد:



منابع و مأخذ:

Django 3.1 for Beginners / Professionals / APIs
Copyright: © 2020 – LearnPub

Mastering Django (Covers 2 & 3)
Copyright: © 2020 – GNW Publishing

Django 3 By Example (Third Edition)
Copyright: © 2020 – PacktPub

Building APIs with Django and Django Rest Framework (DRF)
Copyright: © 2018 – Release 2.0

Beginning Django (Covers 1.11 LTS)
Copyright: © 2017 – Apress

Mastering Django Core (Covers 1.8 LTS)
Copyright: © 2017– LearnPub

Building RESTful Python Web Services
Copyright: © 2016 – PacktPub

https://docs.djangoproject.com/en/dev/
https://docs.djangoproject.com/en/ ## 3.2.x, 4.x, 5.x
https://learndjango.com/tutorials
https://simpleisbetterthancomplex.com
https://www.fullstackpython.com
https://developer.mozilla.org
https://testdriven.io/blog/
https://studygyaan.com/
https://learnbatta.com/ ## full stack tutorials.
https://dev.to/