

Sviluppatori - Primi passi

Il percorso d'ingresso per chi sviluppa con noi: strumenti, documentazione e prove pratiche con Django.

Plasticjumper Srl · aggiornato il 7 Novembre 2026

Sviluppatori - Primi passi

Di che cosa parla
Il percorso d'ingresso per chi sviluppa con noi: strumenti, documentazione e prove pratiche con Django.

Azienda
Plasticjumper Srl

Ultimo aggiornamento
7 Novembre 2026, 23:51

PDF preparato il
8 Novembre 2026, 06:24

Versione online, sempre aggiornata
https://pandora.plasticjumper.it/manuale/10/sviluppatori-primi-passi/

Formato
PDF/A accessibile: segnalibri, indice e struttura dei titoli per i lettori di schermo; le immagini hanno il testo alternativo.

Pandora, il portale dei manuali di Plastic Jumper · pandora.plasticjumper.it

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it



Figura 1 Copertina del manuale Sviluppatori - Primi passi

Questo manuale ti accompagna nelle prime settimane in Plastic Jumper. Ti dice quali strumenti usiamo, dove studiare e, soprattutto, ti propone una serie di prove pratiche da svolgere in ordine.

Come usarlo:

1. Se vuoi un ambiente pulito, parti dal modulo **Preparare il sistema**. È facoltativo: puoi saltarlo se lavori già su Linux o su macOS.
2. Installa e configura gli **Strumenti**.
3. Tieni a portata di mano la **Documentazione per smanettare**: la consulterai durante le prove.
4. Svolgi le **Prove** una alla volta. Ognuna ha un obiettivo, i passi e un modo per verificare di averla fatta.
5. Chiudi con la **Prova finale** e consegna il lavoro.

Non serve sapere tutto subito. Serve leggere la documentazione, provare, sbagliare e capire perché. Quando ti blocchi per più di mezz'ora, chiedi: è parte del lavoro.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Il manuale è aggiornato a ottobre 2026: Python 3.13, Django 6.1 (la versione a supporto lungo è la 5.2), PostgreSQL 18.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Indice

Preparare il sistema (facoltativo) 9
Prima di cominciare 9
Macchina virtuale o computer di prova 9
Quale distribuzione 9
Installare Ubuntu Server 10
Scaricare l'immagine 10
Creare la macchina virtuale 10
Installazione guidata 10
Primo accesso e aggiornamenti 10
SSH e firewall 11
Pacchetti di base 11
Se usi un Mac 12
Macchina virtuale con UTM 12
Lavorare direttamente su macOS 12
Se usi Windows 12
WSL2 in breve 13

Strumenti 15
Python e ambienti virtuali 15
Quale versione 15
uv 15
python -m venv 16
Django 16
Versioni 16
Come è fatto un progetto 16
Database 16
PostgreSQL in produzione 16
SQLite in locale 17
Editor e qualità del codice 17
Quale editor 17
Ruff 17
Git e GitHub 17
Git 17
GitHub 18
Pubblicazione e assistente 18
Dokku 18
Claude Code 18

Documentazione per smanettare 19
Il tutorial ufficiale di Django 19
Installazione e tutorial 19

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Le pagine da consultare più spesso ... 19
Pacchetti che usiamo oggi ... 20
Accesso, sicurezza e API ... 20
Sviluppo, PDF e interfaccia ... 20
Pacchetti superati ... 20
Accessibilità ... 20
Le regole ... 21
Gli strumenti di verifica ... 21
Altre letture utili ... 21
Per approfondire ... 21

Prove di base ... 23
Come funzionano le prove ... 23
Regole del gioco ... 23
Prova 1 - Ambiente e progetto ... 23
Passi della prova 1 ... 23
Verifica della prova 1 ... 24
Prova 2 - Un'app, un modello, l'admin ... 24
Passi della prova 2 ... 24
Verifica della prova 2 ... 24
Prova 3 - Viste e template ... 24
Passi della prova 3 ... 25
Verifica della prova 3 ... 25
Prova 4 - Un modulo con validazione ... 25
Passi della prova 4 ... 25
Verifica della prova 4 ... 25
Prova 5 - I test ... 26
Passi della prova 5 ... 26
Verifica della prova 5 ... 26
Prova 6 - Rami e richieste di unione ... 26
Passi della prova 6 ... 26
Verifica della prova 6 ... 27

Prove avanzate ... 29
Prova 7 - PostgreSQL e variabili d'ambiente ... 29
Passi della prova 7 ... 29
Verifica della prova 7 ... 29
Prova 8 - Un'API con Django REST Framework ... 30
Passi della prova 8 ... 30
Verifica della prova 8 ... 30
Prova 9 - Accesso con django-allauth ... 30
Passi della prova 9 ... 30
Verifica della prova 9 ... 30
Prova 10 - Una pagina accessibile ... 31
Passi della prova 10 ... 31

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Verifica della prova 10 31
Prova 11 - Trovare e correggere un bug 31
Il codice da correggere 31
Passi della prova 11 32
Verifica della prova 11 32
Prova 12 - README e CHANGELOG 33
Passi della prova 12 33
Verifica della prova 12 33
Prove in più 33
Un PDF con WeasyPrint 33
Un comando di gestione 33
Proteggere l'accesso 34
Esportare i dati 34

Prova finale 35
Il progetto: registro delle letture 35
Cosa deve fare 35
Requisiti tecnici 35
Facoltativo 36
Come lavorare 36
Organizzazione 36
Cosa consegnare 36
Elenco della consegna 36
Come sarà valutata 36

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

MODULO 1

Preparare il sistema (facoltativo)

Questo modulo è facoltativo. Ti serve se vuoi un ambiente Linux pulito, simile ai server su cui pubblichiamo, senza toccare il sistema che usi ogni giorno. Se lavori già su Linux o preferisci macOS, passa pure al modulo **Strumenti**.

1.1 Prima di cominciare

Decidi dove installare e quale distribuzione usare. Bastano pochi minuti, ma evita di dover ricominciare.

Macchina virtuale o computer di prova

Una **macchina virtuale** è un computer simulato dentro il tuo. È la scelta più comoda: se sbagli, la cancelli e ne crei un'altra. Ti servono almeno 2 processori, 4 GB di memoria e 25 GB di disco liberi.

Programmi per creare macchine virtuali:

- VirtualBox, gratuito, per Linux, Windows e Mac con processore Intel.
- virt-manager, su Linux, basato su KVM: più leggero e veloce.
- UTM, per Mac, soprattutto con processore Apple Silicon (M1, M2, M3, M4…).

Un **computer di prova** (un vecchio portatile, un mini PC) va bene se vuoi l'esperienza completa. Ricorda che l'installazione cancella il disco.

Quale distribuzione

Ti consigliamo **Ubuntu Server 24.04 LTS**. LTS vuol dire supporto lungo: aggiornamenti di sicurezza per anni. Alternative valide:

- **Debian 13** («trixie»): stabile e sobria, molto simile a Ubuntu nei comandi.
- **Fedora Server** o altre distribuzioni: vanno bene, ma i comandi dei pacchetti cambiano (`dnf` al posto di `apt`). Cerca l'equivalente quando serve.

La versione «Server» non ha interfaccia grafica: lavori dal terminale. È proprio quello che vogliamo imparare.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

1.2 Installare Ubuntu Server

I passi essenziali, dall'immagine al primo accesso remoto.

Scaricare l'immagine

Scarica l'immagine ISO dalla pagina di Ubuntu Server. Per un Mac con Apple Silicon usa invece la versione per ARM.

Per Debian, l'immagine «netinst» è sulla pagina di download di Debian.

Creare la macchina virtuale

Nel programma di virtualizzazione:

1. Crea una nuova macchina di tipo Linux, Ubuntu a 64 bit.
2. Assegna 2 processori, 4 GB di memoria, un disco da 25 GB.
3. Collega la ISO come disco di avvio.
4. Per la rete scegli «NAT» (più semplice) oppure «bridge» se vuoi raggiungerla da altri computer.

Se usi un computer di prova, copia la ISO su una chiavetta con balenaEtcher o con `dd`, e avvia da lì.

Installazione guidata

L'installatore fa quasi tutto da solo. Fai attenzione a questi passi:

- **Lingua e tastiera**: scegli italiano o inglese per la lingua, tastiera italiana.
- **Disco**: usa l'intero disco (nella macchina virtuale è il disco virtuale).
- **Utente**: inserisci nome, nome del computer, nome utente e una password robusta. Annotala in un gestore di password.
- **SSH**: spunta «Install OpenSSH server». Ti servirà per collegarti dal tuo computer.
- **Pacchetti aggiuntivi** (snap): non scegliere nulla.

Al termine, riavvia e accedi con il tuo utente.

Primo accesso e aggiornamenti

Aggiorna subito il sistema:

```
sudo apt update
sudo apt full-upgrade -y
sudo reboot
```

Imposta il fuso orario, se non è già giusto:

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

```
sudo timedatectl set-timezone Europe/Rome
```

SSH e firewall

Dal tuo computer collegati alla macchina con SSH invece di usare la finestra della macchina virtuale. Trova l'indirizzo con `ip a`, poi:

```
ssh tuonome@indirizzo-della-macchina
```

Meglio ancora con una chiave al posto della password. Sul tuo computer:

```
ssh-keygen -t ed25519
ssh-copy-id tuonome@indirizzo-della-macchina
```

Poi attiva il firewall **ufw**, lasciando aperto SSH. L'ordine conta: se attivi il firewall prima di permettere SSH, resti chiuso fuori.

```
sudo ufw allow OpenSSH
sudo ufw enable
sudo ufw status
```

Quando proverai il server di sviluppo di Django da un altro computer, apri anche la porta 8000 con `sudo ufw allow 8000/tcp`, e richiudila quando hai finito.

Pacchetti di base

Installa gli strumenti che userai nelle prove:

```
sudo apt install -y git build-essential curl python3 python3-venv python3-dev
```

Ubuntu 24.04 porta Python 3.12; per la 3.13 usa `uv` (vedi il modulo **Strumenti**). Debian 13 ha già Python 3.13.

PostgreSQL: la versione dei depositi della distribuzione va bene per esercitarti. Per avere l'ultima (la 18) usa il deposito ufficiale di PostgreSQL:

```
sudo apt install -y postgresql-common
sudo /usr/share/postgresql-common/pgdg/apt.postgresql.org.sh
sudo apt install -y postgresql-18 libpq-dev
```

Controlla che il servizio sia attivo con `sudo systemctl status postgresql`.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

1.3 Se usi un Mac

Su Mac hai due strade. Puoi creare una macchina virtuale Linux, oppure lavorare direttamente su macOS. Per le prove di questo manuale vanno bene entrambe.

Macchina virtuale con UTM

Sui Mac con Apple Silicon VirtualBox non è la scelta giusta. Usa UTM:

1. Scarica l'immagine di Ubuntu Server per ARM (ARM64, non AMD64).
2. In UTM scegli «Virtualize», poi «Linux», e seleziona la ISO.
3. Assegna 4 GB di memoria, 2 processori, 25 GB di disco.
4. Avvia e segui l'installazione guidata, come sopra.
5. Dopo l'installazione, togli la ISO dal lettore virtuale prima di riavviare, altrimenti ricomincia l'installazione.

Da qui in poi i passi sono gli stessi di Ubuntu.

Lavorare direttamente su macOS

È la strada più rapida. Installa prima gli strumenti da riga di comando di Apple:

```
xcode-select --install
```

Poi installa Homebrew, seguendo il comando indicato sul sito, e con Homebrew gli strumenti:

```
brew install python@3.13 git postgresql@18 uv
brew services start postgresql@18
```

Differenze da ricordare rispetto a Linux:

- Il terminale predefinito è **zsh**: le impostazioni vanno in `~/.zshrc`, non in `~/.bashrc`.
- PostgreSQL di Homebrew crea un utente con il tuo nome, senza password: non serve `sudo -u postgres`.
- Se `postgresql@18` non è nel percorso dei comandi, Homebrew ti dice quale riga aggiungere a `~/.zshrc`.
- Il file system di macOS di solito non distingue maiuscole e minuscole nei nomi dei file; i server Linux sì. Scrivi i nomi sempre uguali.

1.4 Se usi Windows

Non sviluppiamo direttamente su Windows: usa **WSL2**, che fa girare un vero Ubuntu dentro Windows.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

WSL2 in breve

Apri PowerShell come amministratore e lancia:

```
wsl --install -d Ubuntu-24.04
```

Riavvia, apri «Ubuntu» dal menu Start e crea il tuo utente. Da lì segui i passi di Ubuntu (aggiornamenti e pacchetti di base). Non servono SSH e firewall.

Tieni i progetti dentro Linux (per esempio in `~/progetti`), non in `C:\`: è molto più veloce. VS Code si collega a WSL con l'estensione «WSL». La guida completa è nella documentazione Microsoft su WSL.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

MODULO 2

Strumenti

Gli strumenti con cui lavoriamo ogni giorno: linguaggio, framework, database, editor, controllo delle versioni e pubblicazione. Installali e verifica che funzionino prima di passare alle prove.

2.1 Python e ambienti virtuali

Python è il linguaggio su cui si basa tutto il nostro lavoro.

Quale versione

Usiamo **Python 3.13**. Controlla quale hai con:

```
python3 --version
```

Ogni progetto ha il suo **ambiente virtuale**: una cartella con Python e i pacchetti di quel progetto, separati dal resto del sistema. Non installare mai pacchetti nel Python di sistema.

uv

uv gestisce versioni di Python, ambienti e pacchetti, ed è molto veloce. Su Linux si installa con il comando indicato nella sua documentazione:

```
curl -LsSf https://astral.sh/uv/install.sh | sh
```

Su Mac: `brew install uv`.

Comandi essenziali:

```
uv python install 3.13
uv init mioprogetto
cd mioprogetto
uv add django
uv run django-admin startproject config .
uv run python manage.py runserver
```

`uv add` aggiorna `pyproject.toml` e il file di blocco `uv.lock`: vanno entrambi nel deposito git.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

python -m venv

In alcuni progetti usiamo ancora il metodo classico, con `requirements.txt`:

```
python3.13 -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
```

Ricorda di attivare l'ambiente (`source .venv/bin/activate`) a ogni nuova sessione del terminale, oppure lascia che lo faccia l'editor. Segui il metodo che trovi nel progetto: non mescolarli.

2.2 Django

Django è il framework web che usiamo per tutti i portali.

Versioni

- **Django 6.1** è la versione corrente: la usiamo nei progetti nuovi.
- **Django 5.2** è la versione a supporto lungo (LTS): la trovi nei progetti che devono restare stabili a lungo.

Controlla la versione installata con `python -m django --version`. Le date di supporto di ogni versione sono nella pagina Download di Django.

Come è fatto un progetto

Un **progetto** Django contiene impostazioni, indirizzi principali e una o più **app**. Ogni app ha i suoi modelli (i dati), le viste (la logica), i template (le pagine) e i test. Lo vedrai bene nel tutorial ufficiale e nelle prove.

2.3 Database

I dati dei nostri portali stanno in un database relazionale.

PostgreSQL in produzione

PostgreSQL è il database di tutti i progetti nuovi; la versione corrente è la 18. In qualche progetto più vecchio trovi ancora MySQL.

Su Linux crea un utente e un database per le prove:

```
sudo -u postgres createuser --pwprompt tirocinio
sudo -u postgres createdb --owner tirocinio tirocinio
```

Su Mac, con PostgreSQL di Homebrew, ometti `sudo -u postgres`.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Django parla con PostgreSQL attraverso il pacchetto `psycopg` (la versione 3): `uv add "psycopg[binary]"`.

SQLite in locale

Per sviluppare in locale va bene SQLite, un database in un solo file, già incluso in Python. È l'impostazione predefinita di un progetto Django nuovo. Prima di consegnare, però, prova sempre anche con PostgreSQL: i due non si comportano in modo identico.

2.4 Editor e qualità del codice

Un buon editor ti fa risparmiare tempo e ti segnala gli errori mentre scrivi.

Quale editor

- Visual Studio Code, il più usato da noi. Installa l'estensione Python di Microsoft.
- VSCodium, lo stesso editor senza la telemetria di Microsoft.
- PyCharm, di JetBrains, con una versione gratuita.

Su Ubuntu senza interfaccia grafica lavori con VS Code dal tuo computer, collegandoti via SSH con l'estensione «Remote - SSH».

Ruff

Al posto di pylint e flake8 usiamo Ruff: controlla lo stile e formatta il codice, in una frazione di secondo.

```
uv add --dev ruff
uv run ruff check .
uv run ruff format .
```

In VS Code installa l'estensione «Ruff». Le regole di stile seguono la PEP 8.

2.5 Git e GitHub

Ogni riga di codice che scrivi finisce in un deposito git.

Git

Git tiene la storia delle modifiche. Prima di tutto presentati:

```
git config --global user.name "Nome Cognome"
git config --global user.email "nome@esempio.it"
git config --global init.defaultBranch main
```


Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Il libro Pro Git è gratuito e in parte tradotto in italiano: leggi almeno i primi tre capitoli.

GitHub

I nostri depositi sono su GitHub, privati. Crea un account, attiva l'autenticazione a due fattori e aggiungi la tua chiave SSH pubblica nelle impostazioni (`cat ~/.ssh/id_ed25519.pub`). Poi verifica il collegamento:

```
ssh -T git@github.com
```

Se vuoi, installa anche GitHub CLI (`gh`): crea depositi e richieste di unione dal terminale.

2.6 Pubblicazione e assistente

Due strumenti che vedrai usare fin dai primi giorni.

Dokku

Pubblichiamo i portali su un nostro server con Dokku, una piattaforma che costruisce e avvia l'applicazione a ogni `git push`. Prima di pubblicare tutti i test devono passare: se uno fallisce, non si pubblica. Durante il tirocinio non pubblichi da solo, ma è utile che tu sappia come funziona.

Claude Code

Usiamo Claude Code, un assistente di programmazione che lavora nel terminale. È utile, ma durante le prove scrivi tu il codice: l'obiettivo è che tu capisca. Puoi usarlo per farti spiegare un errore o un pezzo di documentazione; se ti aiuta a scrivere qualcosa, devi saperlo spiegare riga per riga.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

MODULO 3

Documentazione per smanettare

Le letture da tenere aperte durante le prove: il tutorial ufficiale di Django, i pacchetti che usiamo davvero e le regole di accessibilità. Quando una prova ti chiede di «consultare» qualcosa, lo trovi qui.

3.1 Il tutorial ufficiale di Django

È il punto di partenza di tutti. Fallo per intero, prima o insieme alle prime prove.

Installazione e tutorial

La documentazione di Django è tradotta in italiano. Usa sempre quella della versione che stai usando:

- Guida rapida all'installazione
- Tutorial, parte 1: la prima app
- Le parti dalla 2 alla 8 si raggiungono dal fondo di ogni pagina: database e admin, viste, moduli, test, file statici, admin su misura, pacchetti riutilizzabili.

La traduzione a volte è incompleta: se una frase è poco chiara, passa alla versione inglese del tutorial. Il tutorial usa `pip` e `venv`; puoi seguirlo anche con `uv`.

Le pagine da consultare più spesso

- Modelli
- Viste basate su classi
- Template
- Moduli (form)
- Test
- Impostazioni
- Elenco di controllo per la messa in produzione

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

3.2 Pacchetti che usiamo oggi

Oltre a Django, questi pacchetti tornano in quasi tutti i nostri progetti. Leggi almeno la pagina introduttiva di ciascuno.

Accesso, sicurezza e API

- django-allauth: registrazione e accesso degli utenti, anche con passkey e doppio fattore.
- django-axes: blocca i tentativi ripetuti di indovinare una password.
- Django REST Framework: le API REST dei portali.
- django-hijack: permette all'amministratore di vedere il sito come lo vede un utente. Oggi è un solo pacchetto, già integrato con l'admin.

Sviluppo, PDF e interfaccia

- django-debug-toolbar: mostra query, template e tempi di ogni pagina, solo in sviluppo.
- WeasyPrint: trasforma HTML e CSS in PDF (ricevute, schede, stampe).
- Bootstrap 5: lo trovi in alcuni progetti. Nei progetti nuovi usiamo invece un pacchetto comune interno, con fogli di stile e componenti nostri.

Caratteri, fogli di stile e script sono sempre serviti dal nostro sito: niente CDN, niente Google Fonts. Così il sito non passa dati dei visitatori a terzi e funziona anche se un servizio esterno si ferma.

Pacchetti superati

Nei progetti vecchi potresti trovare questi pacchetti. Non usarli nei progetti nuovi:

- **django-bootstrap4**: Bootstrap 4 non è più aggiornato.
- **django-bootstrap-datepicker-plus**: per le date basta il campo HTML `<input type="date">`, accessibile e senza script.
- **django-admin-tools**: non segue più le versioni recenti di Django; l'admin di Django basta.
- **django-hijack-admin**: incluso in django-hijack.
- **pylint** e **flake8**: sostituiti da Ruff.
- **Bitbucket**: i depositi sono passati a GitHub.

3.3 Accessibilità

I nostri portali devono essere usabili da tutti, anche da chi usa solo la tastiera o un lettore di schermo. Per le pubbliche amministrazioni è un obbligo di legge.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Le regole

L'obiettivo è il livello AA delle WCAG 2.1, richiesto dalle linee guida di AgID e dalla norma europea EN 301 549. In pratica:

- contrasto almeno 4,5:1 per il testo normale, 3:1 per il testo grande e per i bordi dei campi;
- ogni campo di un modulo ha la sua etichetta;
- tutto si usa con la tastiera, e si vede sempre dove si trova il cursore;
- a 320 pixel di larghezza nulla esce dallo schermo;
- titoli in ordine: un solo h1, il titolo della pagina.

Una buona introduzione è WCAG in breve del W3C.

Gli strumenti di verifica

- axe-core: il motore di controllo che usiamo nei test, in Chrome senza finestra.
- L'estensione **axe DevTools** per Chrome e Firefox: lo stesso controllo, a mano, su una pagina.
- La tastiera: Tab, Maiusc+Tab, Invio, Spazio, Esc. Nessuno strumento automatico sostituisce questa prova.

3.4 Altre letture utili

Per quando avrai finito le prove, o vorrai capire meglio.

Per approfondire

- La documentazione di Python 3.13, in inglese, con il tutorial del linguaggio.
- Pro Git, per git.
- Mantieni un changelog e il versionamento semantico: come scriviamo il CHANGELOG e numeriamo le versioni.
- Django Forum, dove chiedere quando la documentazione non basta.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

MODULO 4

Prove di base

Il cuore del manuale. Le prove sono graduate: falle in ordine, ognuna parte da dove è finita la precedente. Tutte costruiscono lo stesso piccolo progetto, un **calendario di eventi** con iscrizioni.

4.1 Come funzionano le prove

Ogni prova ha un obiettivo, cosa consultare, i passi e un modo per verificare di averla fatta.

Regole del gioco

- Lavora in un solo deposito git, `prove-tirocinio`, privato su GitHub. Fai un commit alla fine di ogni prova, con un messaggio chiaro in italiano.
- Scrivi codice, commenti e interfaccia in italiano.
- Quando un passo non è spiegato nei dettagli, è voluto: cerca nella documentazione indicata.
- Se la verifica non torna, non passare alla prova successiva. Capisci prima cosa non va.
- Annota in un file `diario.md` i problemi incontrati e come li hai risolti. Lo guarderemo insieme.

4.2 Prova 1 - Ambiente e progetto

Obiettivo: avere un progetto Django 6.1 che si avvia, in un ambiente virtuale, sotto controllo di git.

Consulta: il modulo **Strumenti** e la guida all'installazione di Django.

Passi della prova 1

1. Crea la cartella `prove-tirocinio` e, dentro, l'ambiente con Python 3.13 (`uv init` oppure `python3.13 -m venv .venv`).
2. Installa Django e crea il progetto nella cartella corrente: `django-admin startproject config .`
3. In `config/settings.py` imposta `LANGUAGE_CODE = "it"` e `TIME_ZONE = "Europe/Rome"`.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

4. Applica le migrazioni iniziali e avvia il server di sviluppo.
5. Crea un file `.gitignore` che escluda almeno `.venv/`, `__pycache__/`, `db.sqlite3` e `.env`.
6. Inizializza git, fai il primo commit, crea il deposito privato su GitHub e invia il codice.

Verifica della prova 1

- `python -m django --version` risponde 6.1 o successiva.
- Su `http://127.0.0.1:8000/` vedi la pagina di benvenuto di Django, in italiano.
- `git status` non mostra né l'ambiente virtuale né il database.
- Il deposito su GitHub è privato e contiene il tuo primo commit.

4.3 Prova 2 - Un'app, un modello, l'admin

Obiettivo: creare l'app `eventi` con il suo modello e gestirla dall'admin di Django.

Consulta: tutorial, parte 2 e l'admin di Django.

Passi della prova 2

1. Crea l'app `eventi` e aggiungila a `INSTALLED_APPS`.
2. Scrivi il modello `Evento` con: titolo, descrizione (facoltativa), data, luogo, posti disponibili (intero positivo), creato il (impostato in automatico).
3. Dai a ogni campo un nome leggibile in italiano (`verbose_name`) e al modello un ordinamento per data.
4. Aggiungi `__str__`, che restituisce il titolo.
5. Crea e applica le migrazioni.
6. Registra il modello nell'admin con colonne (titolo, data, luogo, posti), ricerca per titolo e filtro per data.
7. Crea un superutente e inserisci almeno cinque eventi.

Verifica della prova 2

- `python manage.py makemigrations --check` non trova modifiche in sospeso.
- Nell'admin vedi l'elenco con le colonne, la ricerca e il filtro funzionanti.
- I nomi dei campi nel modulo dell'admin sono in italiano.
- Riesci a spiegare a voce cosa c'è dentro il file di migrazione.

4.4 Prova 3 - Viste e template

Obiettivo: mostrare gli eventi ai visitatori con pagine che condividono la stessa struttura.

Consulta: tutorial, parti 3 e 4, viste basate su classi e ereditarietà dei template.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Passi della prova 3

1. Crea un template `base.html` con: lingua `it`, titolo della pagina, testata con il nome del sito come collegamento alla home, menu in una lista dentro `<nav>`, contenuto in `<main>`, piede.
2. Crea la vista elenco dei prossimi eventi (dalla data di oggi in poi) e la vista dettaglio. Prova una delle due come funzione e l'altra come classe (`ListView`, `DetailView`).
3. I template delle due pagine estendono `base.html` e riempiono i suoi blocchi.
4. Usa indirizzi con nome (`app_name` e `{% url %}`), mai indirizzi scritti a mano.
5. Aggiungi al modello `get_absolute_url` e usalo nell'elenco.
6. Mostra le date con `<time datetime="…">`.

Verifica della prova 3

- Un evento passato non compare nell'elenco.
- Un indirizzo inesistente, come `/eventi/9999/`, dà una pagina 404, non un errore 500.
- Ogni pagina ha un solo `<h1>`, con il titolo della pagina.
- Se cambi il menu in `base.html`, la modifica si vede in tutte le pagine.

4.5 Prova 4 - Un modulo con validazione

Obiettivo: permettere ai visitatori di iscriversi a un evento, controllando i dati inseriti.

Consulta: moduli, moduli dai modelli e validazione.

Passi della prova 4

1. Crea il modello `Iscrizione`, collegato a `Evento` con `related_name="iscrizioni"`, con nome, email e data di iscrizione.
2. Crea un `ModelForm` per l'iscrizione e mostralo nella pagina di dettaglio dell'evento.
3. Aggiungi le regole: niente iscrizioni a eventi passati; niente iscrizioni se i posti sono finiti; la stessa email non si iscrive due volte allo stesso evento.
4. Dopo l'invio riuscito, reindirizza alla pagina dell'evento e mostra un messaggio di conferma con il framework dei messaggi.
5. Gli errori compaiono accanto al campo sbagliato e sono collegati al campo (`aria-describedby`), non solo colorati di rosso.

Verifica della prova 4

- Prova a iscriverti con email vuota, email sbagliata, due volte con la stessa email, a un evento pieno: ogni volta vedi un messaggio chiaro, in italiano.
- Se ricarichi la pagina dopo un'iscrizione riuscita, il browser non invia di nuovo i dati.
- Ogni campo ha un'etichetta `<label>` collegata.
- Hai lasciato attiva la protezione CSRF: nel modulo c'è `{% csrf_token %}`.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

4.6 Prova 5 - I test

Obiettivo: scrivere test automatici che proteggono quello che hai fatto finora.

Consulta: tutorial, parte 5 e i test in Django.

Passi della prova 5

1. Scrivi i test del modello: `__str__`, `get_absolute_url`, il calcolo dei posti liberi.
2. Scrivi i test delle viste con il client di prova: l'elenco esclude gli eventi passati; il dettaglio di un evento inesistente dà 404.
3. Scrivi i test del modulo: un caso valido e almeno un caso per ogni regola della prova 4.
4. Organizza i test in una cartella `eventi/tests/` con un file per argomento.
5. Lancia la suite con `python manage.py test`.

Verifica della prova 5

- La suite finisce con `OK`, senza test saltati.
- Se togli di proposito una regola di validazione, almeno un test fallisce. Poi rimettila.
- Lancia anche `ruff check .`: nessun errore.
- Da qui in poi, a ogni prova la suite deve restare verde prima del commit.

4.7 Prova 6 - Rami e richieste di unione

Obiettivo: lavorare come in squadra, con un ramo per ogni modifica e una revisione prima di unirla.

Consulta: Pro Git, capitolo 3 sui rami e la guida di GitHub sulle richieste di unione (pull request), in inglese.

Passi della prova 6

1. Aggiungi il tuo tutor come collaboratore del deposito.
2. Crea un ramo `aggiunta-luogo-online` partendo da `main` aggiornato.
3. Sul ramo, aggiungi a `Evento` un campo «online» (sì o no) e mostralo nelle pagine, con test.
4. Fai commit piccoli e con messaggi chiari, poi invia il ramo su GitHub.
5. Apri una richiesta di unione verso `main`, con una descrizione: cosa cambia, perché, come provarlo.
6. Rispondi ai commenti del tutor con nuovi commit sullo stesso ramo.
7. Dopo l'approvazione, unisci, cancella il ramo e aggiorna `main` in locale.
8. Prova extra: crea apposta un conflitto (modifica la stessa riga su due rami) e risolvilo.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Verifica della prova 6

- Su GitHub la richiesta di unione risulta unita, con almeno un commento di revisione.
- `git log --oneline --graph` mostra il ramo e la sua unione.
- Sai spiegare la differenza tra `git merge` e `git rebase`, e quando non si usa `git push --force`.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

MODULO 5

Prove avanzate

Queste prove ti portano più vicino a un progetto vero: database di produzione, API, accesso degli utenti, accessibilità, correzione di errori e documentazione. Continua sullo stesso deposito, con un ramo e una richiesta di unione per ogni prova.

5.1 Prova 7 - PostgreSQL e variabili d'ambiente

Obiettivo: usare PostgreSQL al posto di SQLite, senza scrivere password nel codice.

Consulta: impostazioni dei database, note su PostgreSQL e la sezione **Database** del modulo Strumenti.

Passi della prova 7

1. Crea in PostgreSQL l'utente e il database `tirocinio`. Dai all'utente il permesso di creare database, che serve ai test: `ALTER USER tirocinio CREATEDB;`
2. Installa `psycopg[binary]`.
3. In `settings.py` leggi da variabili d'ambiente `SECRET_KEY`, `DEBUG` e i dati del database, con `os.environ`. Se manca il nome del database, il progetto usa SQLite.
4. Metti i valori in un file `.env`, escluso da git. Prepara un `.env.esempio`, con gli stessi nomi e valori finti, da mettere nel deposito.
5. Carica il file all'avvio: con uv, `uv run --env-file .env python manage.py runserver`; con venv, `set -a; source .env; set +a`.
6. Applica le migrazioni sul nuovo database e reinserisci qualche evento.

Verifica della prova 7

- `python manage.py dbshell` apre la console di PostgreSQL; il comando `\dt` mostra le tabelle di `eventi`.
- La suite dei test passa anche su PostgreSQL.
- `git grep -i password` non trova nessuna password vera.
- Senza `.env` il progetto riparte con SQLite.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

5.2 Prova 8 - Un'API con Django REST Framework

Obiettivo: offrire l'elenco degli eventi ad altri programmi, in JSON.

Consulta: il tutorial di Django REST Framework.

Passi della prova 8

1. Installa `djangorestframework` e aggiungilo alle app.
2. Scrivi un `ModelSerializer` per `Evento`, con i posti liberi come campo in sola lettura.
3. Scrivi un `ReadOnlyModelViewSet` e registralo con un router sotto `/api/eventi/`.
4. Lettura libera per tutti; aggiungi poi una seconda vista, solo per utenti autenticati, che elenca le iscrizioni di un evento.
5. Scrivi i test con `APIClient`: risposta 200 e contenuto dell'elenco; risposta 401 o 403 per le iscrizioni senza accesso.

Verifica della prova 8

- Nel browser, `/api/eventi/` mostra la pagina navigabile di DRF.
- Da terminale `curl http://127.0.0.1:8000/api/eventi/` restituisce JSON.
- Gli eventi passati non compaiono, come nella pagina HTML.
- I test dell'API sono verdi.

5.3 Prova 9 - Accesso con django-allauth

Obiettivo: far registrare e accedere gli utenti, e riservare loro una parte del sito.

Consulta: la guida di installazione di django-allauth e la sezione sull'autenticazione di Django.

Passi della prova 9

1. Installa e configura django-allauth seguendo la guida: app, middleware, backend di autenticazione, indirizzi sotto `/accounts/`.
2. Accesso con email al posto del nome utente.
3. In sviluppo, le email vanno nel terminale: `EMAIL_BACKEND = "django.core.mail.backends.console.EmailBackend"`.
4. Crea una pagina «Le mie iscrizioni», visibile solo a chi ha fatto l'accesso; chi non l'ha fatto finisce alla pagina di accesso.
5. Personalizza il template di accesso perché estenda il tuo `base.html`.
6. Facoltativo: attiva l'app `allauth.mfa` per il doppio fattore (codici TOTP) e le passkey, seguendo la sua documentazione.

Verifica della prova 9

- Ti registri, ricevi nel terminale l'email di conferma, confermi e accedi.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

- Uscito dal sito, `/le-mie-iscrizioni/` ti porta alla pagina di accesso.
- Un test verifica entrambi i casi: con e senza accesso.
- Il modulo di accesso ha le etichette e si usa da tastiera.

5.4 Prova 10 - Una pagina accessibile

Obiettivo: rendere la pagina di un evento, con il suo modulo, conforme al livello AA delle WCAG 2.1.

Consulta: la sezione **Accessibilità** del modulo Documentazione.

Passi della prova 10

1. Apri la pagina di un evento in Chrome e lancia l'estensione axe DevTools.
2. Correggi ogni problema segnalato: contrasti, etichette, titoli, lingua, collegamenti senza testo.
3. Definisci i colori una volta sola, come variabili CSS (`--colore-testo`, `--colore-sfondo`…), e usa solo quelle. Niente `opacity` per schiarire un testo.
4. Rendi visibile il cursore della tastiera (`:focus-visible`) su collegamenti, pulsanti e campi.
5. Aggiungi in cima un collegamento «Vai al contenuto» che porta al `<main>`.
6. Prova la pagina larga 320 pixel (strumenti per sviluppatori, modalità dispositivo) e con lo zoom al 200%.
7. Caratteri e fogli di stile stanno nella cartella `static` del progetto: niente CDN.

Verifica della prova 10

- axe non segnala nessuna violazione.
- Con il solo Tab raggiungi, nell'ordine logico, ogni collegamento e ogni campo; invii il modulo con Invio; vedi sempre dove sei.
- A 320 pixel non serve lo scorrimento orizzontale.
- Facoltativo: ascolta la pagina con un lettore di schermo (Orca su Linux, VoiceOver su Mac con Cmd+F5) e verifica che campi ed errori siano letti.

5.5 Prova 11 - Trovare e correggere un bug

Obiettivo: leggere codice scritto da altri, capire perché non funziona e correggerlo con un test.

Consulta: la barra di debug e la documentazione delle query.

Il codice da correggere

Un collega ha scritto questa pagina dei prossimi eventi. La pagina è sempre vuota e, quando mostra qualcosa, va in errore. Ci sono almeno tre errori.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

```
from datetime import date

from django.db import models
from django.shortcuts import render


class Evento(models.Model):
    titolo = models.CharField("titolo", max_length=200)
    data = models.DateField("data")
    posti = models.PositiveIntegerField("posti disponibili")

    def __str__(self):
        return self.titolo

    def posti_liberi(self):
        return self.posti - self.iscrizioni.count


def prossimi(request):
    oggi = date.today()
    eventi = Evento.objects.filter(data__lt=oggi).order_by("data")
    return render(request, "eventi/prossimi.html", {"evento": eventi})
```

E il template:

```
<ul>
{% for evento in eventi %}
  <li><a href="{{ evento.get_absolute_url }}">{{ evento }}</a>:
{{ evento.posti_liberi }} posti liberi</li>
{% empty %}
  <li>Nessun evento in programma.</li>
{% endfor %}
</ul>
```

Passi della prova 11

1. Copia il codice in un ramo nuovo, adattandolo alla tua app.
2. Prima di correggere, scrivi per ogni errore un test che fallisce.
3. Correggi un errore alla volta e lancia i test dopo ogni correzione.
4. Cerca anche i problemi meno evidenti: che data è «oggi» a mezzanotte e mezza, con il fuso di Roma? (Suggerimento: `django.utils.timezone.localdate`.) Quante query fa la pagina con cento eventi? Guardalo con la barra di debug.
5. Nella richiesta di unione descrivi ogni errore: sintomo, causa, correzione.

Verifica della prova 11

- I test scritti al passo 2 prima fallivano, ora passano.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

- La pagina mostra solo gli eventi futuri, con i posti liberi giusti.
- La barra di debug mostra un numero di query che non cresce con il numero di eventi (prova `annotate` con `Count`).

5.6 Prova 12 - README e CHANGELOG

Obiettivo: lasciare il progetto pronto per chi arriva dopo di te.

Consulta: Mantieni un changelog e il versionamento semantico.

Passi della prova 12

1. Scrivi `README.md`: cosa fa il progetto; requisiti; come installarlo, configurarlo (`.env`), avviarlo e lanciare i test; come è organizzato il codice.
2. Scrivi `CHANGELOG.md`, con una voce per ogni versione, dalla `0.1.0` alla versione attuale. Raggruppa le voci in «Aggiunto», «Modificato», «Corretto».
3. Dai a ogni versione un'etichetta git (`git tag v0.1.0`) e inviala su GitHub.
4. Chiedi a un collega di installare il progetto seguendo solo il tuo README.

Verifica della prova 12

- Il collega avvia il progetto e lancia i test senza chiederti nulla.
- La prima voce del CHANGELOG corrisponde all'ultima etichetta git.
- Il README non contiene password né dati personali.

5.7 Prove in più

Se hai finito e vuoi metterti alla prova, scegline una o due. Ognuna in un ramo, con test e richiesta di unione.

Un PDF con WeasyPrint

Genera il biglietto di un'iscrizione in PDF da un template HTML, con WeasyPrint. Verifica: il PDF si apre, ha il titolo dell'evento e il nome dell'iscritto, e il carattere usato è quello del sito, non uno di sistema.

Un comando di gestione

Scrivi `python manage.py promemoria`, che invia un'email agli iscritti degli eventi di domani. Verifica: un test con `call_command` controlla, tramite `mail.outbox`, che le email partano solo per gli eventi giusti.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Proteggere l'accesso

Aggiungi django-axes e verifica, con un test, che dopo cinque tentativi sbagliati l’accesso si blocchi.

Esportare i dati

Aggiungi all’admin un’azione che esporta in CSV gli iscritti degli eventi selezionati. Verifica: il file si apre correttamente nel foglio di calcolo, accenti compresi.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

MODULO 6

Prova finale

La prova finale mette insieme tutto quello che hai fatto. Questa volta parti da zero, in un deposito nuovo, e decidi tu come organizzare il lavoro. Hai a disposizione circa due settimane.

6.1 Il progetto: registro delle letture

Costruisci un piccolo portale in cui ogni utente tiene il registro dei libri che legge.

Cosa deve fare

- Ogni utente si registra e accede con email (django-allauth) e vede solo i propri dati.
- Un libro ha titolo, autore, anno, genere (da un elenco gestito nell'admin) e copertina facoltativa.
- Una lettura collega un utente a un libro, con stato (da leggere, in lettura, letto), data di inizio, data di fine, voto da 1 a 5 e note.
- La data di fine non può precedere quella di inizio; il voto si dà solo ai libri letti.
- Pagine: elenco delle letture con filtro per stato, dettaglio, inserimento e modifica, statistiche dell'anno (libri letti, voto medio, genere preferito).
- Un'API in sola lettura con DRF, solo per l'utente autenticato, che restituisce le sue letture.
- Admin di Django curato: colonne, ricerche, filtri.

Requisiti tecnici

- Python 3.13 e Django 6.1, con uv oppure venv.
- PostgreSQL con le impostazioni da variabili d'ambiente; SQLite ammesso solo come ripiego in locale.
- Test con `python manage.py test` per modelli, moduli, viste e API: la suite finisce con `OK`.
- Ruff senza errori.
- Pagine conformi al livello AA delle WCAG 2.1: nessuna violazione con axe, uso completo da tastiera, nessuno scorrimento orizzontale a 320 pixel.
- Nessuna risorsa da CDN o da Google Fonts: tutto servito dal sito.
- Interfaccia, commenti e messaggi di commit in italiano.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

Facoltativo

Scegli uno o più punti, se hai tempo:

- esportazione del registro dell'anno in PDF con WeasyPrint;
- doppio fattore o passkey con allauth;
- importazione dei libri da un file CSV con un comando di gestione;
- un tema scuro che rispetta i contrasti.

6.2 Come lavorare

Il modo in cui ci arrivi conta quanto il risultato.

Organizzazione

1. Prima di scrivere codice, prepara in `docs/progetto.md` un breve piano: modelli, pagine, indirizzi. Fallo vedere al tutor.
2. Apri una richiesta di unione per ogni funzione (modelli, letture, statistiche, API…), non una sola alla fine.
3. Fai commit piccoli; ognuno lascia la suite verde.
4. Aggiorna il `CHANGELOG.md` a ogni richiesta di unione.
5. Tieni il `diario.md` con problemi, scelte e soluzioni.

6.3 Cosa consegnare

La consegna è il deposito su GitHub, con accesso al tutor.

Elenco della consegna

- Il deposito privato, con `main` aggiornato e le richieste di unione chiuse.
- `README.md` per installare, configurare, avviare e provare il progetto.
- `CHANGELOG.md` con almeno la versione `1.0.0` ed etichetta git corrispondente.
- `.env.esempio` con tutte le variabili necessarie.
- Un file di dati di prova (fixture) da caricare con `python manage.py loaddata`.
- `diario.md` e `docs/progetto.md`.
- Una nota breve con l'esito dei controlli: test, Ruff, axe sulle pagine principali, prova con la tastiera.

Come sarà valutata

Ne parleremo insieme, in un incontro di circa un'ora in cui mostri il progetto e rispondi alle domande. Guardiamo:

- **funziona**: si installa dal README e fa ciò che è richiesto;

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it

- **è affidabile**: test significativi, validazioni, nessun dato di un utente visibile a un altro;
- **è accessibile**: lo proviamo con tastiera e axe;
- **è leggibile**: codice ordinato, nomi chiari, commit e richieste di unione comprensibili;
- **lo capisci**: sai spiegare ogni parte, anche quelle scritte con l'aiuto di un assistente.

Non serve la perfezione. Serve un lavoro onesto, ordinato e che sai difendere.

Plastic Jumper srl · Sede legale: Via G. L. Lagrange 16 — 00197 Roma · Sedi operative: Via Boccaleone 12 — 44121 Ferrara (FE), Via Gramicia 59 — 44123 Ferrara (FE) · contact@plasticjumper.it · PEC plasticjumper@ftpec.it · P.IVA / C.F. 14107281009 — REA RM-1497703 — Codice destinatario 5YOORSD · www.plasticjumper.it