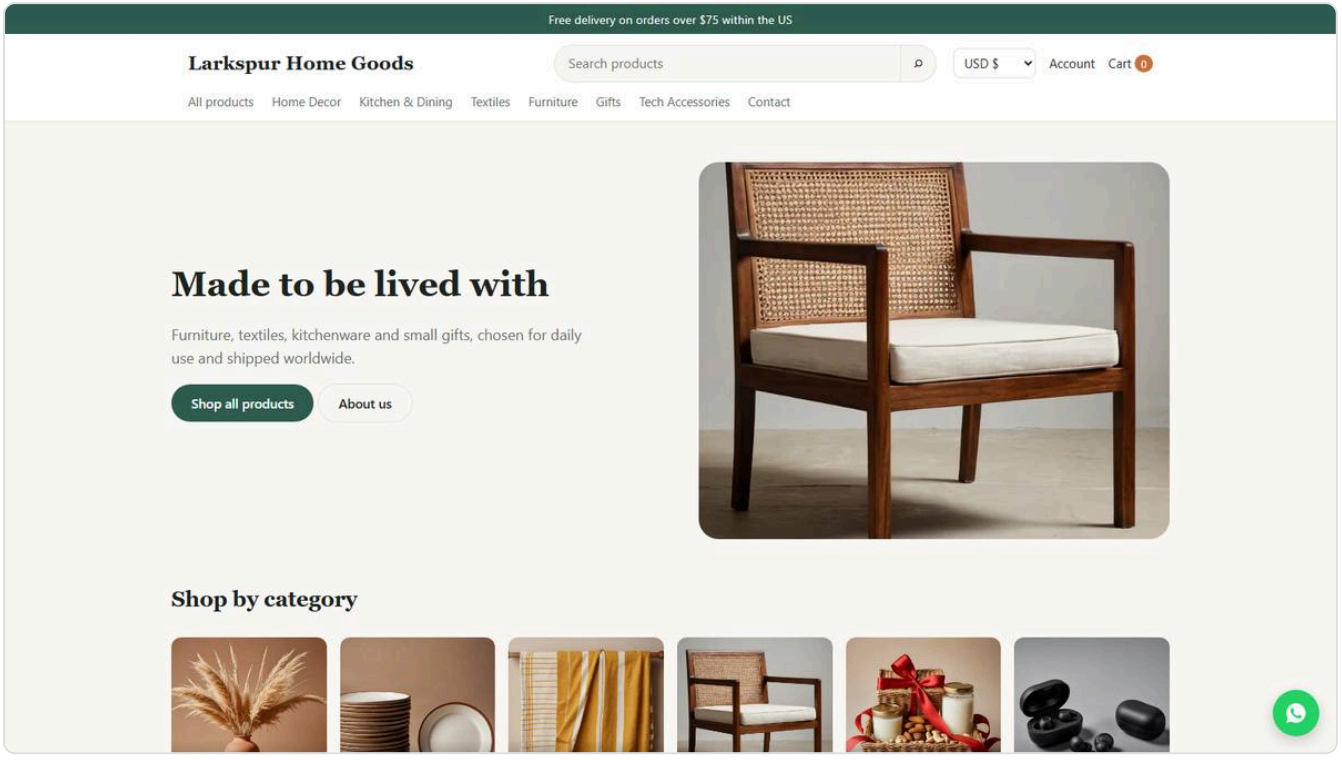
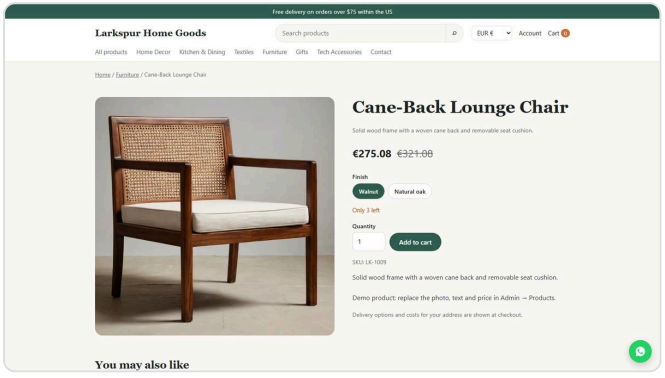
CODE CRAFT · INSTALLATION GUIDE · VERSION 1.0

Django Ecommerce Store

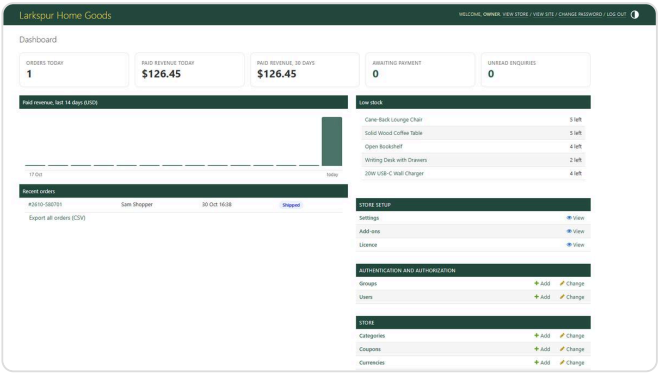
Code Craft Store for Django 5.2 LTS: storefront, owner control centre, variants, coupons, multi-currency, delivery zones, taxes and add-ons. This guide takes you from the download to a live store on your computer, cPanel (Python App / Passenger), a VPS with gunicorn and nginx, or a Hostinger VPS.



The storefront with the demo catalogue (Larkspur Home Goods). Every name, colour, photo and text is set in the admin.



Product with a variant choice; each variant has its own price and stock.



Admin dashboard: orders today, paid revenue, 14-day chart, low stock.

1. What is in the download

Path	What it is
`manage.py`, `config/`	Django project: settings (read from `.env`), URLs, WSGI.
`passenger_wsgi.py`	Entry point for cPanel "Setup Python App" and other Passenger hosts.
`shop/`	The store: models, storefront views and templates, admin control centre, tests.
`shop/demo_media/`	Demo product photos, copied to `media/demo/` by `seed_demo`.
`addons/`	Put add-on folders here (Stripe, Razorpay, PayPal, Auto Currency Rates are sold separately).
`requirements.txt`, `.env.example`	Python packages and the settings template.

No `.env`, database or virtual environment is included: you create them during installation, so nothing from our machine ends up on your server.

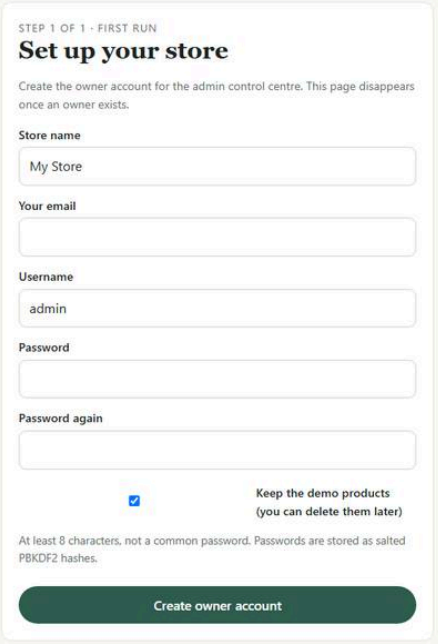
Requirements

- Python 3.10 or later (tested with 3.14). cPanel needs the **Setup Python App** feature (CloudLinux), which most cPanel hosts offer.
- SQLite works with no setup. PostgreSQL or MySQL/MariaDB are optional.
- Packages: Django 5.2 LTS, WhiteNoise (static files), segno (UPI QR codes), gunicorn (Linux servers).

2. Try it on your computer

```
python -m venv .venv
.venv\Scripts\activate            # Windows
source .venv/bin/activate           # macOS / Linux
pip install -r requirements.txt
cp .env.example .env                # Windows: copy .env.example .env
python -c "import secrets; print(secrets.token_urlsafe(50))"   # paste as SECRET_KEY in .env
python manage.py migrate
python manage.py seed_demo          # demo products, currencies, shipping, taxes, coupons
python manage.py collectstatic --noinput
python manage.py runserver          # open http://127.0.0.1:8000
```

The first visit opens **/setup/**. Create the owner account there; the page closes itself once an owner exists. Untick "Keep the demo products" if you want an empty catalogue.



First-run setup: the owner account is created here, with a strong password (8+ characters, not a common password).

3. cPanel: Setup Python App (Passenger)

Upload zip → Create Python App → pip install → .env → migrate → Restart

1. **Upload**: in File Manager, upload the zip to your home folder (not public_html) and extract it, e.g. to `/home/USER/store`.
2. **Create the app**: cPanel → Software → **Setup Python App** → Create Application.

Python version	3.10 or newer (pick the newest offered)
Application root	`store`
Application URL	your domain (or a subdomain)
Application startup file	`passenger_wsgi.py`
Application Entry point	`application`

3. **Install packages**: on the app page, copy the "Enter to the virtual environment" command, open **Terminal** in cPanel (or SSH), paste it, then run the commands below. If there is no terminal, type `requirements.txt` into "Configuration files", click Add, then **Run Pip Install**.

```
source /home/USER/virtualenv/store/3.12/bin/activate && cd /home/USER/store
pip install -r requirements.txt
cp .env.example .env
nano .env        # SECRET_KEY, DEBUG=0, ALLOWED_HOSTS=yourdomain.com,www.yourdomain.com,
                 # CSRF_TRUSTED_ORIGINS=https://yourdomain.com, HTTPS=1
python manage.py migrate
python manage.py seed_demo
python manage.py collectstatic --noinput
```

4. **Restart** the app on the Setup Python App page, then open `https://yourdomain.com/setup/`.
5. **SSL**: cPanel → SSL/TLS Status → Run AutoSSL, so the site works on https.

> No terminal at all? You can set the variables from `.env` as "Environment variables" on the Python App page instead, and run `migrate`, `seed_demo` and `collectstatic` once through cPanel → Cron Jobs (a one-off job such as `cd ~/store && ~/virtualenv/store/3.12/bin/python manage.py migrate`, then delete the job).

Static files are served by WhiteNoise and uploaded photos by Django (`SERVE_MEDIA=1`), so no extra Apache rules are needed. After any code or `.env` change, click **Restart** (or `touch tmp/restart.txt`).

4. VPS: gunicorn + nginx (Ubuntu / Debian)

```
sudo apt update && sudo apt install -y python3-venv nginx
sudo adduser --system --group --home /srv/store store
sudo -u store bash -c 'cd /srv/store && unzip /tmp/django-ecommerce-store.zip && mv Code-Craft-
Django-Store app'
cd /srv/store/app
sudo -u store python3 -m venv .venv
sudo -u store .venv/bin/pip install -r requirements.txt
sudo -u store cp .env.example .env && sudo -u store nano .env   # SECRET_KEY, DEBUG=0,
ALLOWED_HOSTS,
                                      # CSRF_TRUSTED_ORIGINS=https://..., HTTPS=1, SERVE_MEDIA=0
sudo -u store .venv/bin/python manage.py migrate
sudo -u store .venv/bin/python manage.py seed_demo
sudo -u store .venv/bin/python manage.py collectstatic --noinput
```

systemd service: /etc/systemd/system/store.service

```
[Unit]
Description=Code Craft Django store
After=network.target

[Service]
User=store
Group=store
WorkingDirectory=/srv/store/app
ExecStart=/srv/store/app/.venv/bin/gunicorn config.wsgi:application --bind 127.0.0.1:8001 --workers
3 --timeout 60
Restart=always

[Install]
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload && sudo systemctl enable --now store
```

nginx site: /etc/nginx/sites-available/store

```
server {
    server_name yourdomain.com www.yourdomain.com;
    client_max_body_size 20M;
    location /static/ { alias /srv/store/app/staticfiles/; expires 30d; }
    location /media/  { alias /srv/store/app/media/; expires 7d; }
    location / {
        proxy_pass http://127.0.0.1:8001;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

```
sudo ln -s /etc/nginx/sites-available/store /etc/nginx/sites-enabled/
sudo nginx -t && sudo systemctl reload nginx
sudo apt install -y certbot python3-certbot-nginx && sudo certbot --nginx -d yourdomain.com -d
www.yourdomain.com
```

> Make sure nginx can read `staticfiles/` and `media/` (e.g. `sudo chmod o+rx /srv/store /srv/store/app` and the two folders). With `SERVE_MEDIA=0`, nginx serves uploaded photos.

5. Hostinger VPS

1. hPanel → **VPS** → choose your server → **OS & Panel** → install a plain **Ubuntu 24.04** template (or keep the one you have). Note the server IP and the root password or add your SSH key.
2. hPanel → **Domains** → DNS / Nameservers: add an **A record** for `@` and `www` pointing to the VPS IP. DNS can take up to a few hours.
3. Connect with SSH (`ssh root@IP`, or the browser terminal in hPanel) and follow section 4 exactly.
4. If the VPS firewall is on (hPanel → VPS → Security → Firewall), allow ports 22, 80 and 443.
5. Open `https://yourdomain.com/setup/` and create the owner account.

> Hostinger **shared** and **Cloud** web-hosting plans do not run Python web apps; use a Hostinger VPS (or any cPanel host with Setup Python App).

6. PostgreSQL or MySQL instead of SQLite

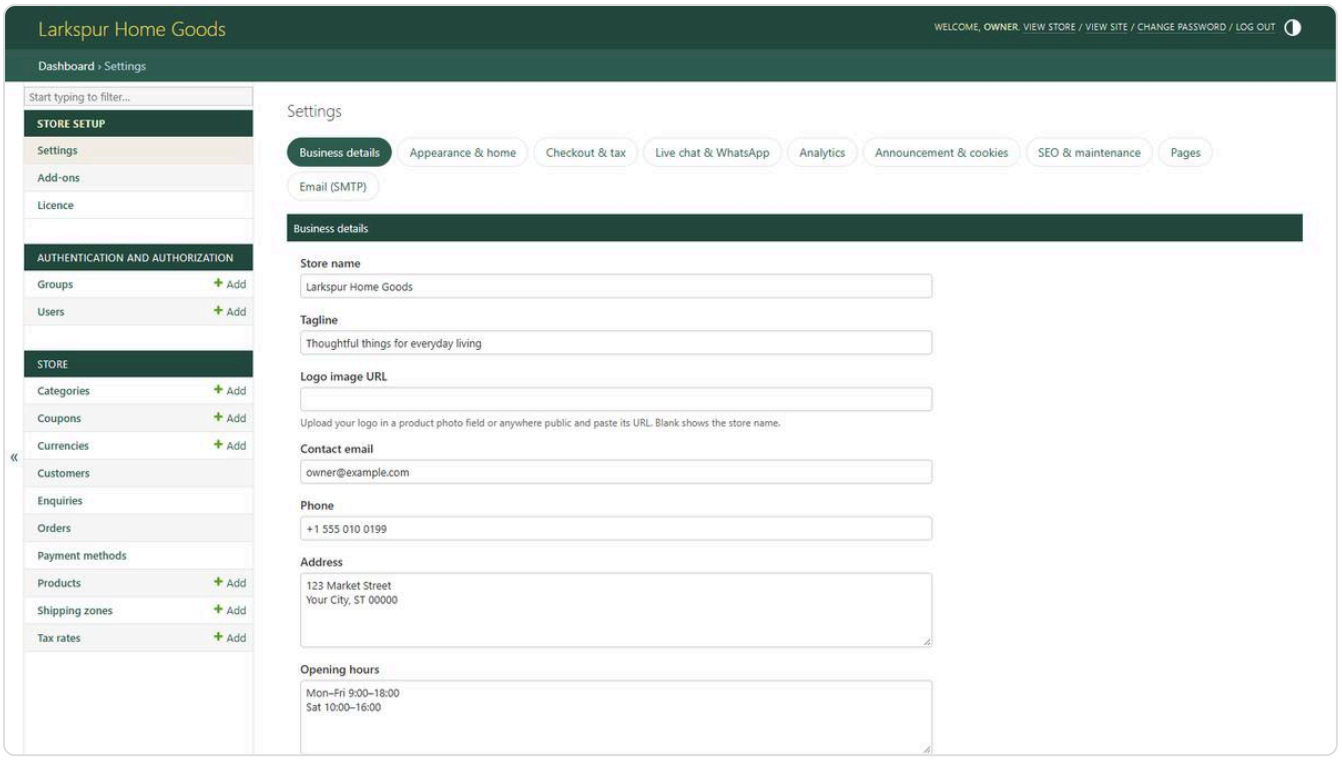
SQLite is fine for most small shops. To use a database server, create an empty database and user, install the driver and set `DATABASE_URL` in `.env` **before** running `migrate`:

```
pip install "psycopg[binary]>=3.2"      # PostgreSQL
DATABASE_URL=postgres://store_user:PASSWORD@127.0.0.1:5432/store_db

pip install "mysqlclient>=2.2"          # MySQL / MariaDB (needs the MySQL client library)
DATABASE_URL=mysql://store_user:PASSWORD@127.0.0.1:3306/store_db
```

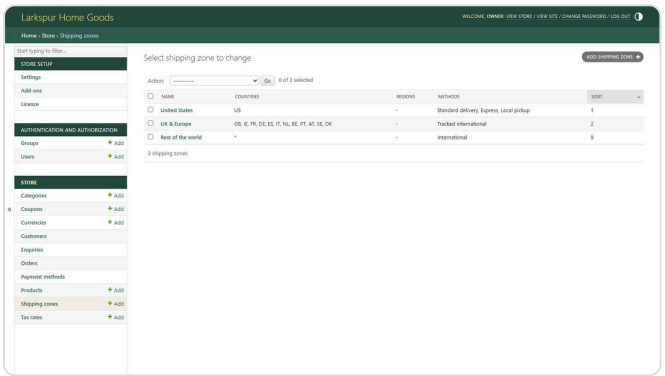
Special characters in the password must be URL-encoded (e.g. `@` becomes `%40`). The store was tested on SQLite; the queries use the Django ORM only, so they are portable.

7. Set up your shop

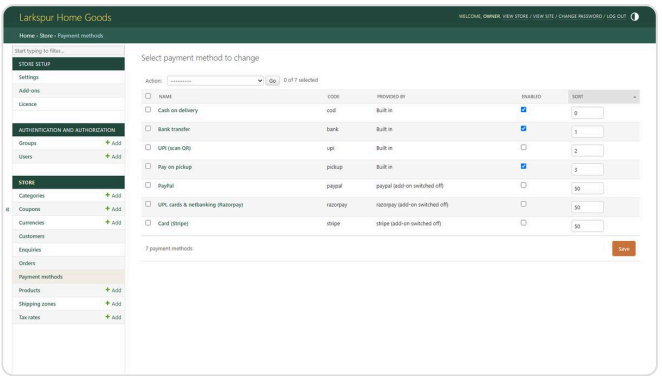


Admin → Settings: business details, appearance, checkout and tax, chat and WhatsApp, analytics, announcement and cookies, SEO and maintenance, pages, SMTP email.

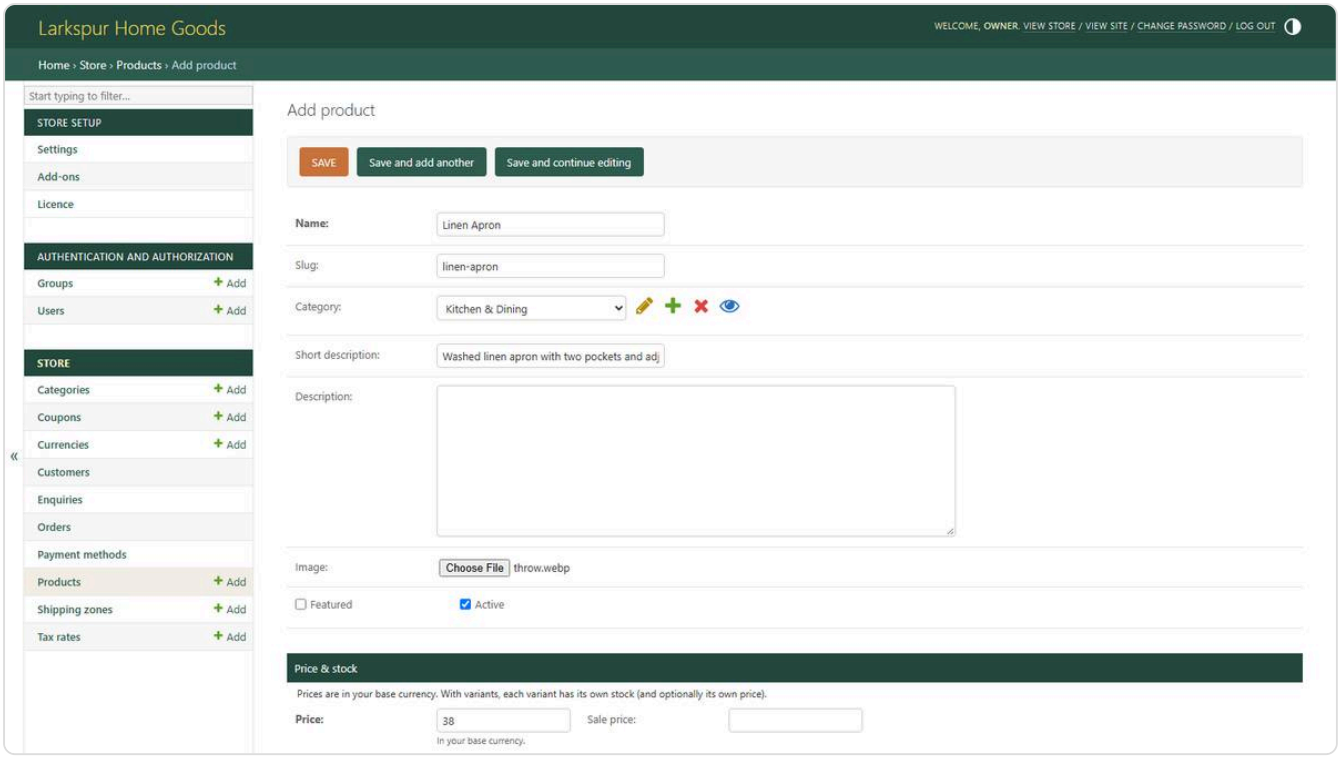
1. **Settings**: name, logo, contact details, address, map, colours, home banner; pages (about, shipping, returns, privacy, terms); SMTP email with a test button.
2. **Currencies**: choose the base currency (the one you type prices in), enable others with a rate, symbol, decimals and symbol position.
3. **Tax rates**: one rate per country, or per region; "Prices I enter include tax" in Settings → Checkout & tax.
4. **Shipping zones**: a list of countries (or `*` for the rest of the world) and optional regions, each with methods: flat rate, free over an amount, weight table, order-value table, local pickup. Table lines are `up to:cost`, e.g. `2:5`.
5. **Payment methods**: tick the ones you use and set the order. Bank details, UPI ID and pickup address are in Settings → Checkout & tax.
6. **Products and categories**: replace the demo catalogue.



Shipping zones with their methods.



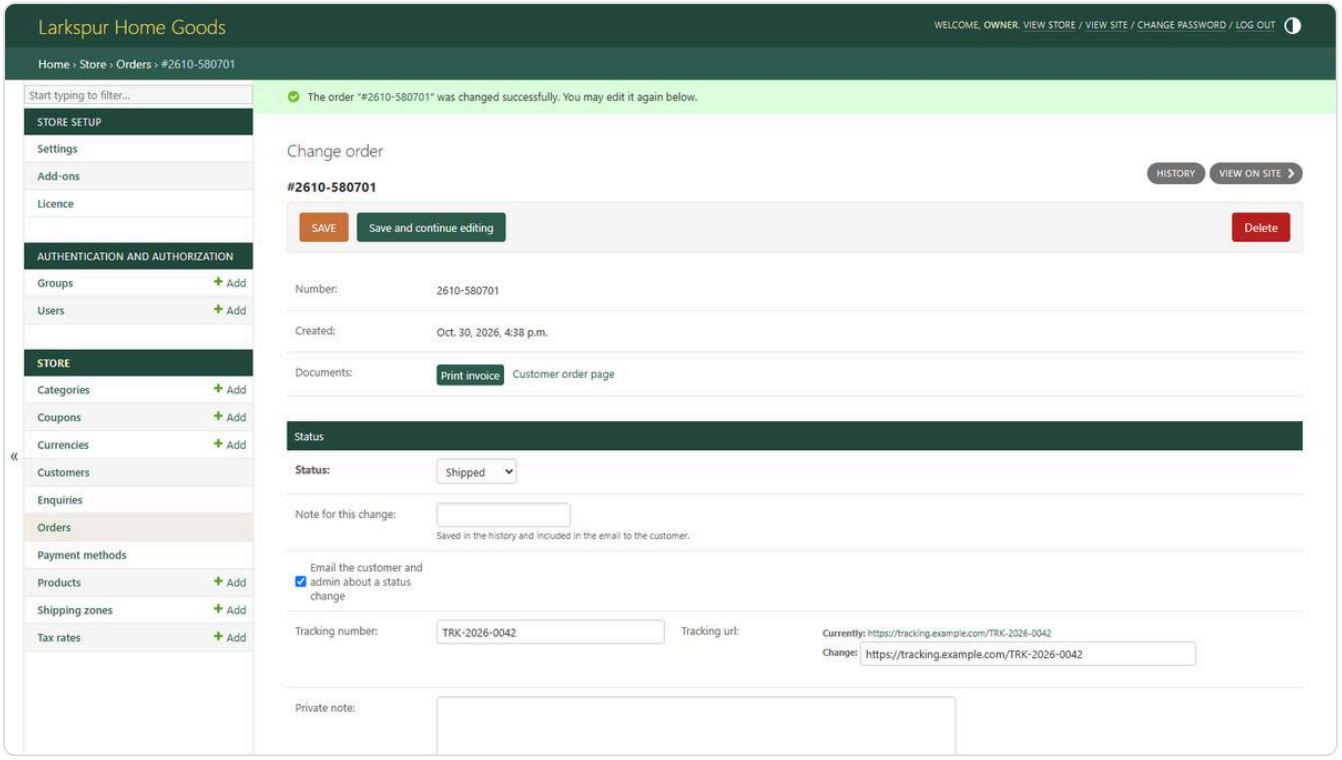
Payment methods: built-in ones and any installed add-ons.



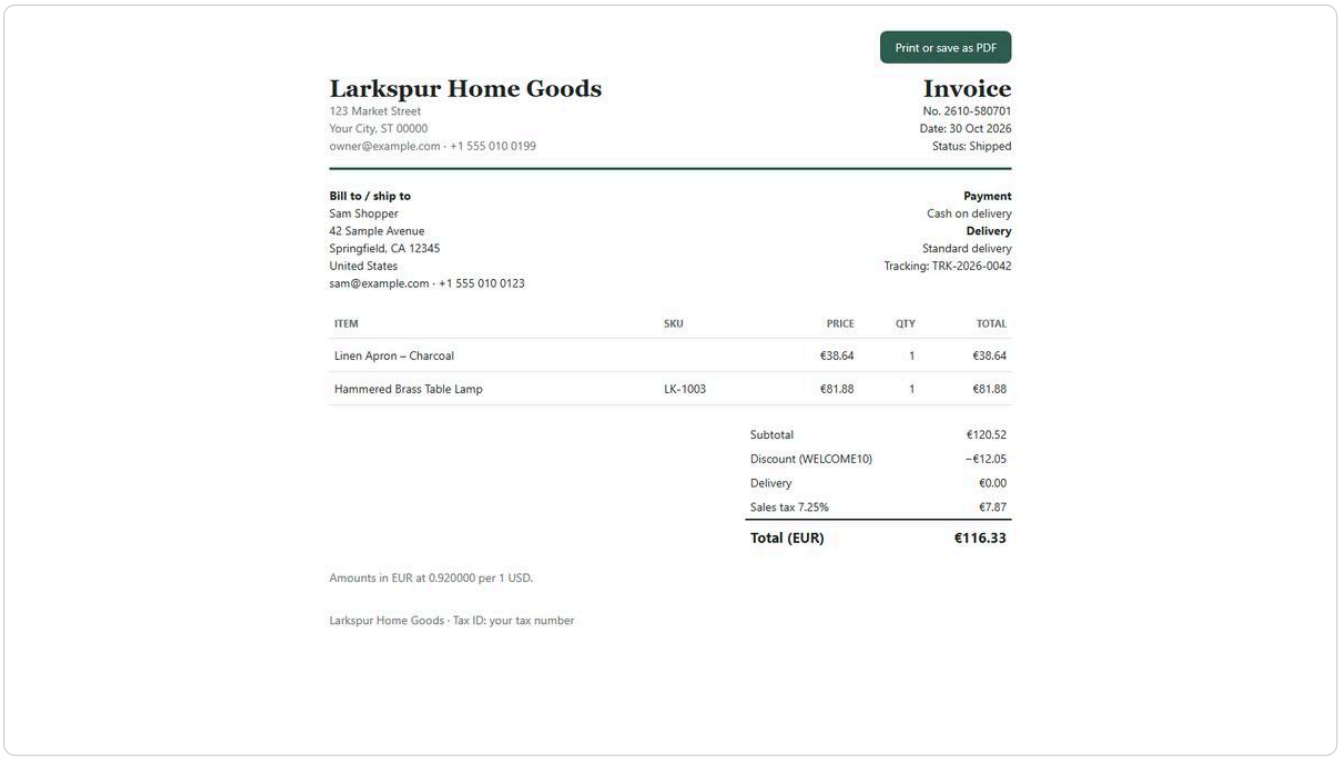
Product editor: price, sale price, stock, low-stock alert, variants with their own price and stock, photos and SEO.

8. Orders

An order starts as **Pending** (or **Paid** when a gateway add-on confirms the payment). Open it in Admin → Orders, change the status, add a tracking number and link and an optional note, and save: the change is logged and emailed to the customer and to you. Cancelling or refunding returns the stock once. "Print invoice" opens a printable invoice (save as PDF from the print dialog); Admin → Orders → select → "Export selected orders to CSV", or the dashboard link, exports orders.



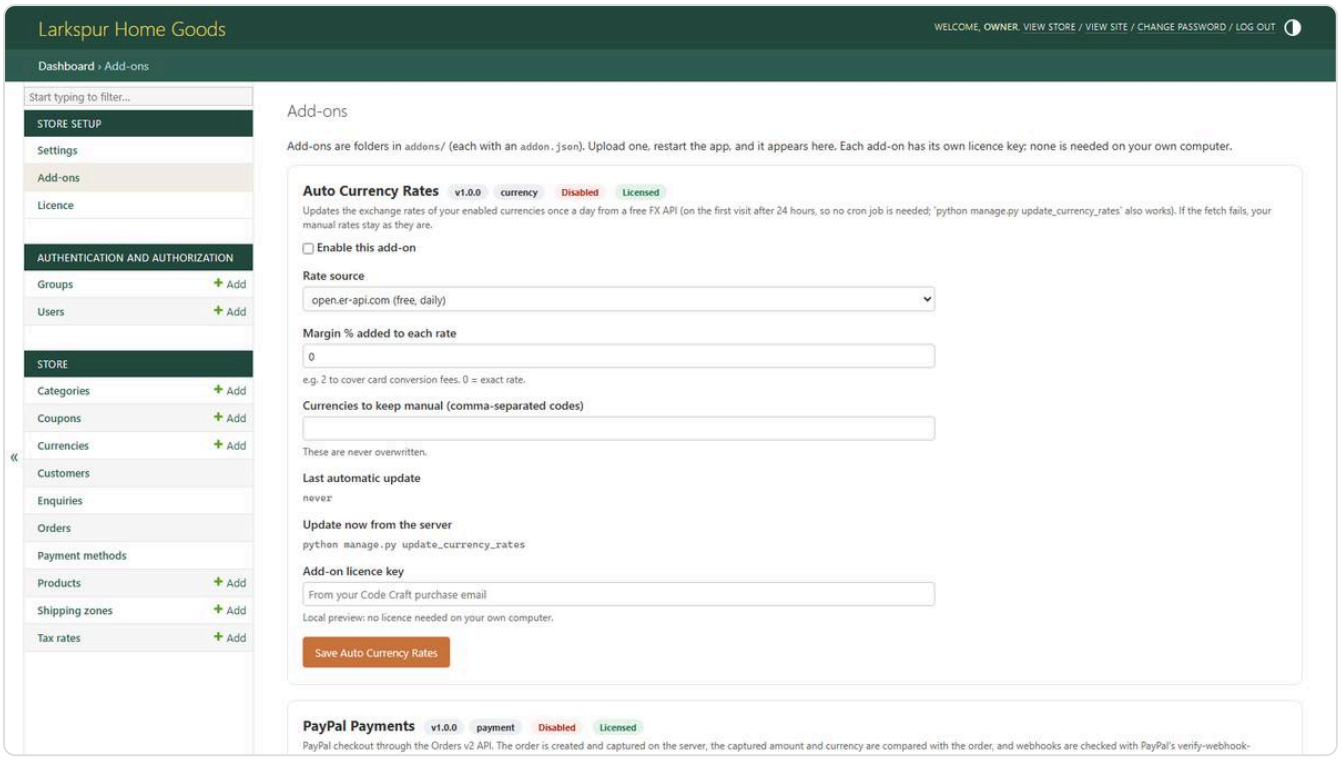
Order screen: status, note, tracking number and link, history, invoice link.



Printable invoice in the order's currency.

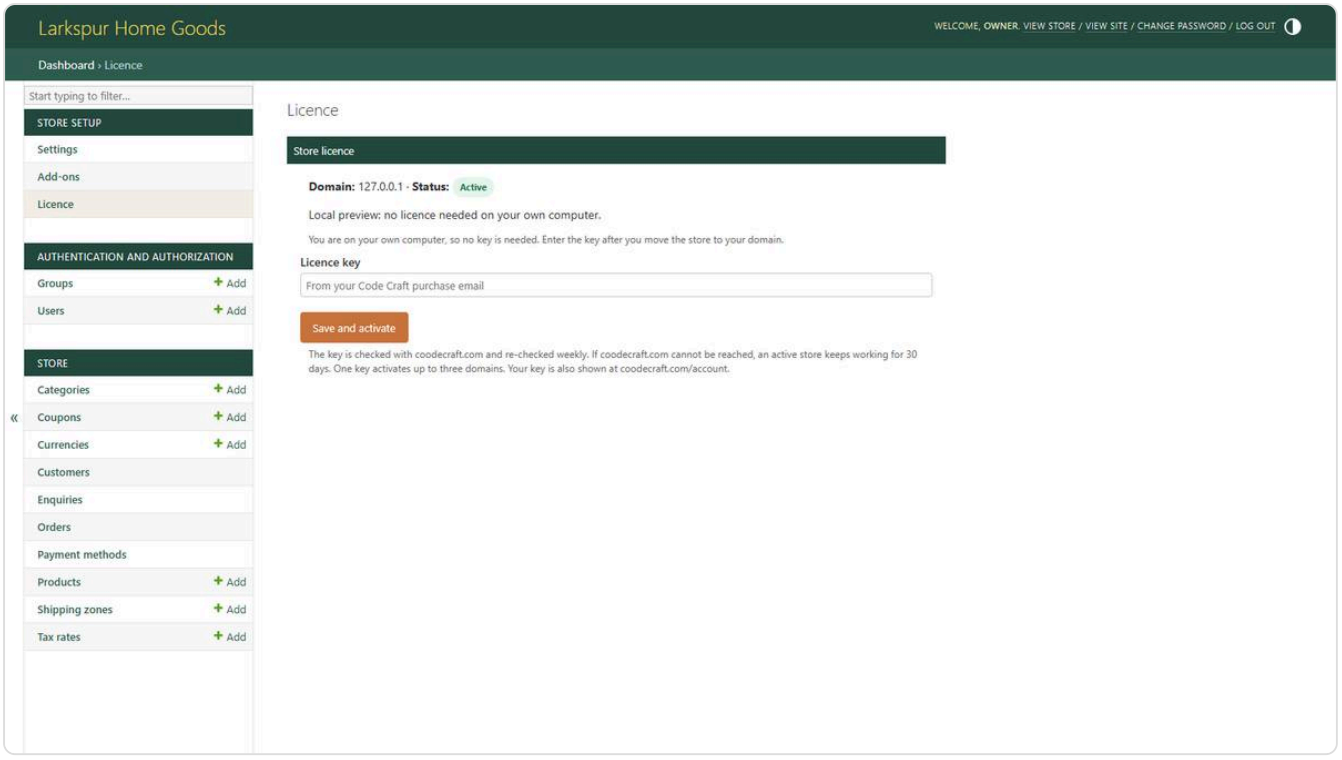
9. Add-ons, live chat, analytics and licence

Add-ons are folders you upload to `addons/`; restart the app and they appear in Admin → Add-ons with their own settings and licence key. Payment add-ons then appear in Payment methods, where you tick them.



Admin → Add-ons: each add-on has its settings, an enable switch and its own licence key.

Live chat (Tawk.to, Crisp, Tidio or your own snippet), the **WhatsApp** button, **GA4**, **Google Tag Manager** and **Meta Pixel** are in Settings. When the cookie notice is on, analytics load only after the visitor accepts.



Admin → Licence. No key is needed on your own computer.

Licence: paste the key from your purchase email in Admin → Licence. It is checked with coodecraft.com and re-checked weekly; if coodecraft.com cannot be reached, an active store keeps working for 30 days. Until the licence is active on a live domain, checkout and the contact form are switched off and a banner shows. One key activates up to three domains.

10. Updates, backups and troubleshooting

Backups

Back up `db.sqlite3` (or your database), the `media/` folder and `.env`. That is everything that is yours.

Updating to a new version

```
# keep .env, db.sqlite3 and media/; replace the other files, then:
pip install -r requirements.txt
python manage.py migrate
python manage.py collectstatic --noinput
# then restart (cPanel: Restart button; VPS: sudo systemctl restart store)
```

Problem	Fix
"SECRET_KEY is not set"	Create `.env` from `.env.example` and set SECRET_KEY, then restart.
Bad Request (400)	Add your domain to `ALLOWED_HOSTS` (with and without www).
CSRF verification failed on https	Set `CSRF_TRUSTED_ORIGINS=https://yourdomain.com` and `HTTPS=1`.
Page has no styling	Run `collectstatic` and restart.
Product photos missing (VPS)	Check the nginx `/media/` alias and folder permissions, or set `SERVE_MEDIA=1`.
Emails not arriving	Fill in Settings → Email (SMTP) and use "Send a test email". Without SMTP, emails are written to the server log.
"Too many attempts" at login	Wait 15 minutes; logins lock after 5 failed tries.
Checkout switched off	Activate your licence in Admin → Licence.

Help with installation: **coodecraft.com/contact**.