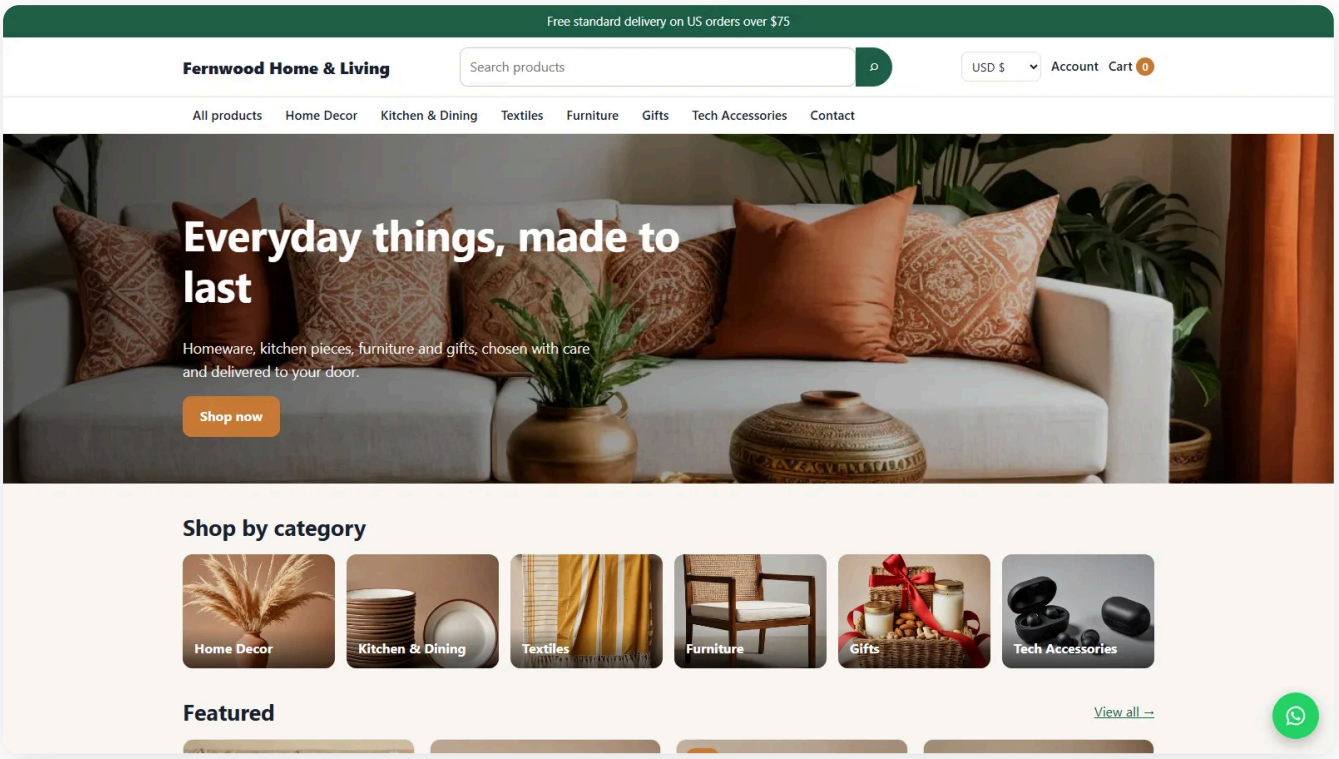
CODE CRAFT · COODECRAFT.COM

Laravel Ecommerce Store

Installation and setup guide: your own computer, cPanel / shared hosting, Hostinger and a VPS. Version 1.0.0.



Contents

1. What you receive and what you need
2. Try it on your own computer
3. Install on cPanel / shared hosting
4. Install on Hostinger
5. Install on a VPS (Ubuntu, Nginx)
6. First-run setup wizard
7. Activate your licence
8. Set up your store: products, payments, shipping, taxes, currencies
9. Orders, emails and invoices
10. Chat, WhatsApp, analytics and SEO
11. Add-ons
12. Going live checklist
13. Troubleshooting

1. What you receive and what you need

`laravel-ecommerce-store.zip` contains the complete Laravel application **including the `vendor/` folder**, so you do not need Composer or SSH. There is no `.env` file and no database in the zip: you create them during installation from `.env.example`.

Requirement	Details
PHP	8.3 or newer (Laravel 13)
PHP extensions	pdo_sqlite or pdo_mysql, mbstring, openssl, fileinfo, curl, ctype, tokenizer, xml (standard on cPanel and Hostinger)
Database	SQLite (built in, nothing to set up) or MySQL / MariaDB
Web server	Apache (cPanel, Hostinger) or Nginx

Folders you will see: `app/` (code), `public/` (the only folder the web should see), `resources/views/` (page templates), `database/`, `storage/` (logs, cache, sessions), `addons/` (add-ons), `vendor/` (Laravel).

2. Try it on your own computer

```
cd laravel-ecommerce-store
cp .env.example .env          (Windows: copy .env.example .env)
php artisan key:generate
php artisan migrate --seed    (creates database/database.sqlite with 24 demo products)
php artisan serve
```

Open `http://localhost:8000/setup`, create your admin account, then browse the store. On your own computer no licence key is needed.

3. Install on cPanel / shared hosting

A Laravel site must be served from its `public/` folder. cPanel gives you three ways to do that; option A is the cleanest.

Step 1 — Upload and extract

1. cPanel → **File Manager**. Open your home folder (`/home/USER` , one level *above* public_html).
2. Upload `laravel-ecommerce-store.zip` , right-click → **Extract**. Rename the folder to `store` .
3. cPanel → **Select PHP Version** (or MultiPHP Manager): choose **PHP 8.3** for your domain and make sure pdo_sqlite (or pdo_mysql), mbstring, fileinfo, openssl and curl are ticked.

Step 2 — Point the website at `store/public`

Option A (recommended): change the document root. cPanel → **Domains** → Manage your domain (or add a subdomain/addon domain) and set the document root to `store/public` . Done — nothing else to move.

Option B: keep public_html. If your main domain's document root cannot be changed:

1. Move everything inside `store/public/` (including the hidden `.htaccess` ; tick "Show hidden files" in File Manager settings) into `public_html/` .
2. Edit `public_html/index.php` and change `$storeRoot = __DIR__.'/..';` to `$storeRoot = __DIR__.'/../store';`

Option C: extract into public_html. If you extracted the whole store into `public_html` , the included root `.htaccess` forwards every request into `public/` and blocks `.env` , `storage` , `vendor` and the other private folders. It works, but A or B is safer.

Step 3 — Create the .env file

1. In `store/` copy `.env.example` to `.env` and edit it: `APP_ENV=production` , `APP_DEBUG=false` , `APP_URL=https://your-domain.com` .
2. **SQLite (simplest):** keep `DB_CONNECTION=sqlite` . **MySQL:** cPanel → MySQL Database Wizard: create a database and user with all privileges, then set `DB_CONNECTION=mysql` , `DB_HOST=localhost` , `DB_DATABASE` , `DB_USERNAME` , `DB_PASSWORD` .

Step 4 — Create the database tables

With Terminal (cPanel → Terminal, if your plan has it):

```
cd ~/store
php artisan key:generate --force
php artisan migrate --seed --force
php artisan config:cache
```

Without Terminal: cPanel → **Cron Jobs** → add a one-off job set to run every minute with the command below, wait two minutes, check that `store/database/database.sqlite` (or your MySQL tables) exists, then **delete the cron job**.

```
cd ~/store && php artisan key:generate --force && php artisan migrate --seed --force
```

> Folders `storage/` and `bootstrap/cache/` must be writable (permission 755 is normal on cPanel), and with SQLite so must `database/`. Run AutoSSL (cPanel → SSL/TLS Status) so the store opens on https://.

4. Install on Hostinger

Shared / Cloud web hosting (hPanel)

1. hPanel → **Websites** → **Manage** → **Advanced** → **PHP Configuration**: choose PHP 8.3. Under PHP extensions keep pdo_sqlite, pdo_mysql, mbstring, fileinfo, curl and openssl on.
2. hPanel → **Files** → **File Manager**. Go to `domains/your-domain.com/` (the folder that contains `public_html`), upload the zip and extract it there; rename the folder to `store`.
3. Hostinger's document root is `public_html`, so use **Option B** from section 3: move the contents of `store/public/` into `public_html/` and set `$storeRoot = __DIR__.'/../store';` in `public_html/index.php`.
4. For MySQL: hPanel → **Databases** → **Management** → create a database and user. In `.env` use `DB_HOST=127.0.0.1` (or `localhost`).
5. Create `store/.env` from `.env.example` as in section 3, step 3.
6. hPanel → **Advanced** → **SSH Access** (Premium plans and up), connect and run the commands from section 3, step 4. Without SSH use **Advanced** → **Cron Jobs** with the same one-off command.
7. hPanel → **Security** → **SSL**: install the free SSL certificate.

Hostinger VPS

Follow section 5.

5. Install on a VPS (Ubuntu 22.04/24.04, Nginx)

```
sudo apt update
sudo apt install nginx php8.3-fpm php8.3-sqlite3 php8.3-mysql php8.3-mbstring php8.3-xml php8.3-
curl php8.3-zip unzip
sudo mkdir -p /var/www/store && sudo unzip laravel-ecommerce-store.zip -d /var/www/store
cd /var/www/store && sudo cp .env.example .env    # edit APP_URL, APP_ENV=production,
APP_DEBUG=false
sudo php artisan key:generate --force && sudo php artisan migrate --seed --force
sudo chown -R www-data:www-data storage bootstrap/cache database public/uploads
```

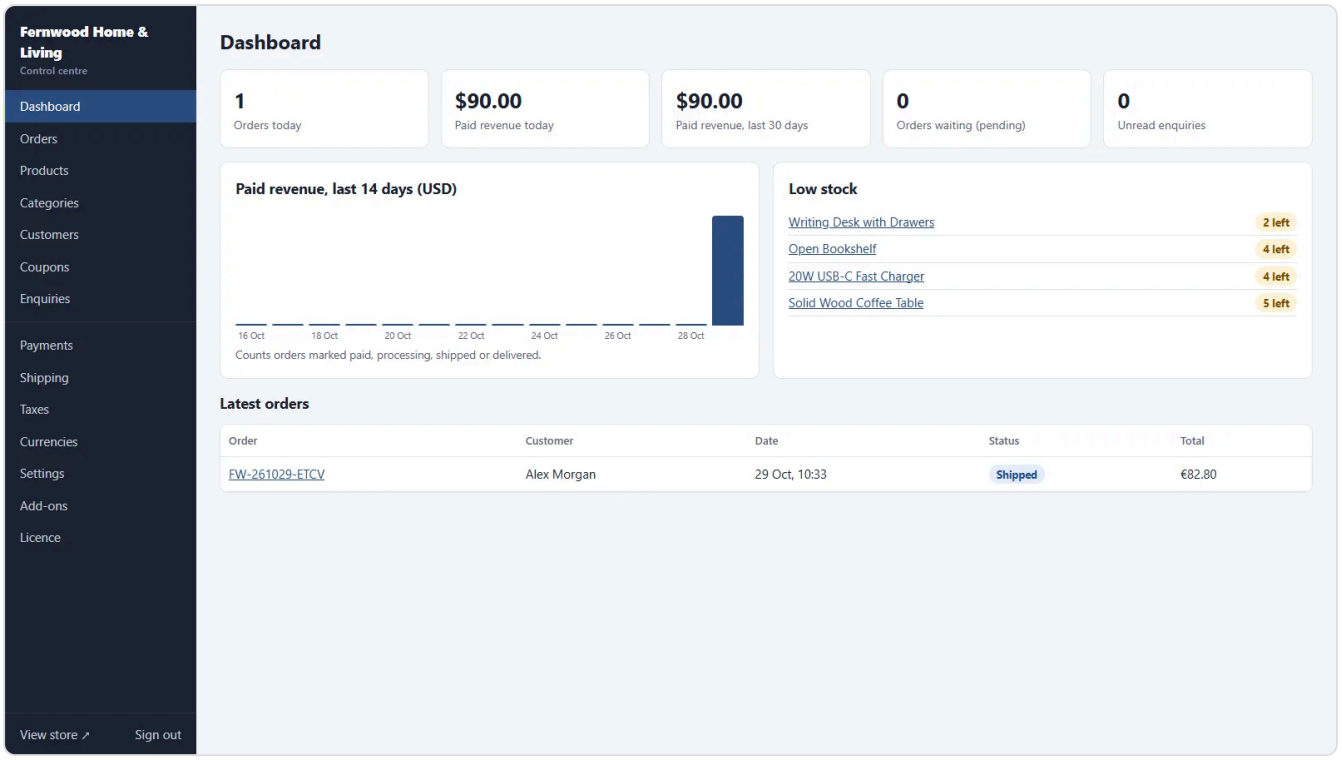
Nginx site (`/etc/nginx/sites-available/store`):

```
server {
    listen 80;
    server_name your-domain.com www.your-domain.com;
    root /var/www/store/public;
    index index.php;
    client_max_body_size 20M;
    location / { try_files $uri $uri/ /index.php?$query_string; }
    location ~ \.php$ { include snippets/fastcgi-php.conf; fastcgi_pass unix:/run/php/php8.3-
fpm.sock; }
    location ~ /\.(?!well-known) { deny all; }
}
```

```
sudo ln -s /etc/nginx/sites-available/store /etc/nginx/sites-enabled/ && sudo nginx -t && sudo
systemctl reload nginx
sudo apt install certbot python3-certbot-nginx && sudo certbot --nginx -d your-domain.com -d
www.your-domain.com
# optional scheduler (used by add-ons): crontab -e -u www-data
* * * * * php /var/www/store/artisan schedule:run >> /dev/null 2>&1
```

6. First-run setup wizard

Open `https://your-domain.com/setup`. Enter your store name and email, choose the base currency (the currency you type prices in) and your main country, and create the admin login. The password is stored hashed (bcrypt). The wizard closes for good once an admin exists; from then on sign in at `/admin`. Five wrong passwords in a row lock the login for a minute.



The control centre dashboard: orders today, paid revenue, low stock and the latest orders.

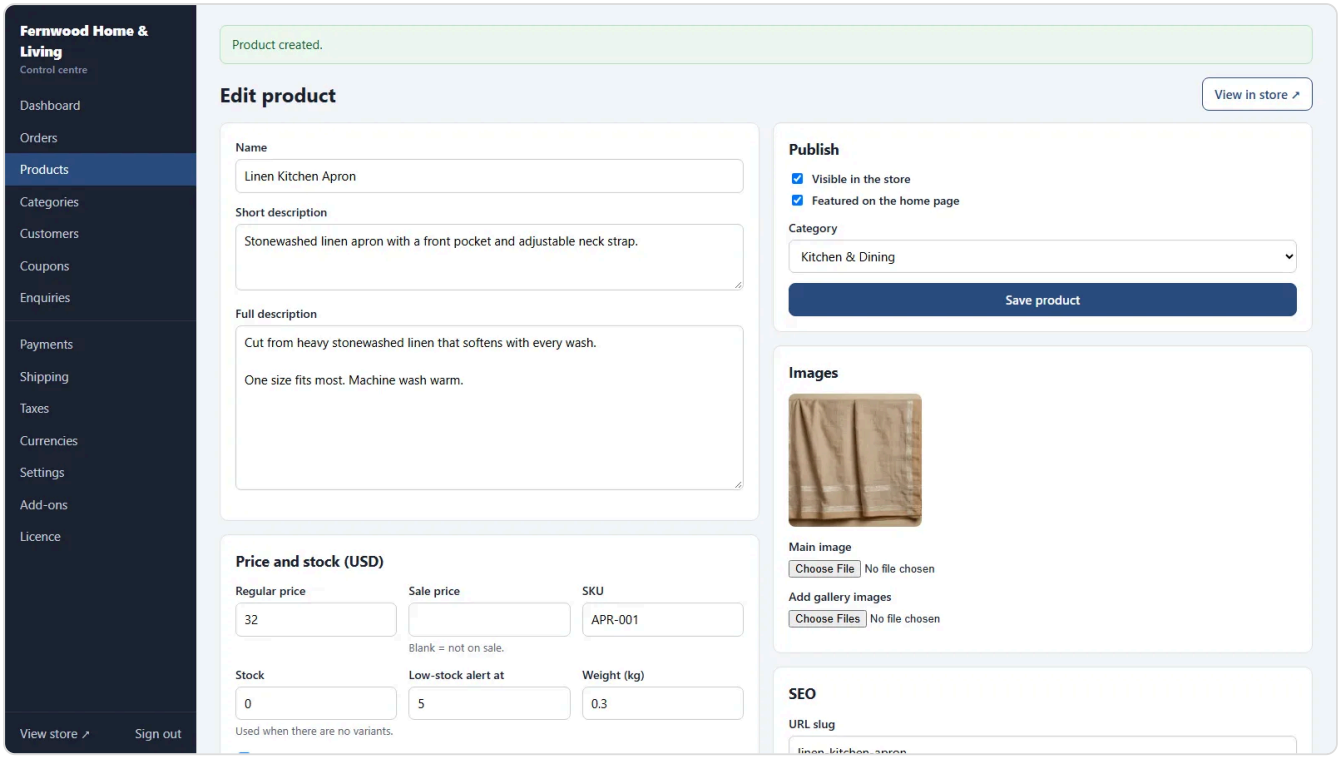
7. Activate your licence

Your licence key is in your purchase email and at coodecraft.com/account. Admin → **Licence** → paste the key → **Save and activate**. The store checks the key with coodecraft.com, then re-checks about once a week (if coodecraft.com can't 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 the admin shows a banner. One key covers up to three domains; use **Deactivate on this domain** to move it.

8. Set up your store

Products and variants

Admin → **Products** → **Add product**. Enter name, descriptions, regular and sale price (in your base currency), SKU, stock, low-stock alert level and weight, choose a category and upload a main image and gallery images. For sizes or colours add **variants**, one line per combination written as `Size: Large; Colour: Blue`, each with its own SKU, stock and optional price. SEO title, description and URL slug are at the bottom right.



Product editor with price, stock, variants, images and SEO fields.

Categories can be nested (Home Decor → Lighting). The 24 demo products use photos in `public/uploads/demo/`; edit or delete them, or start empty by setting `SEED_DEMO_CATALOGUE=false` before seeding.

Payments

Admin → **Payments**: tick the methods to offer and set their order and wording. Built in: Cash on delivery, Bank transfer (shows the bank details you enter), UPI (a QR code is generated from your UPI ID for each order) and Pay on pickup. Card gateways come as add-ons (section 11).

Shipping

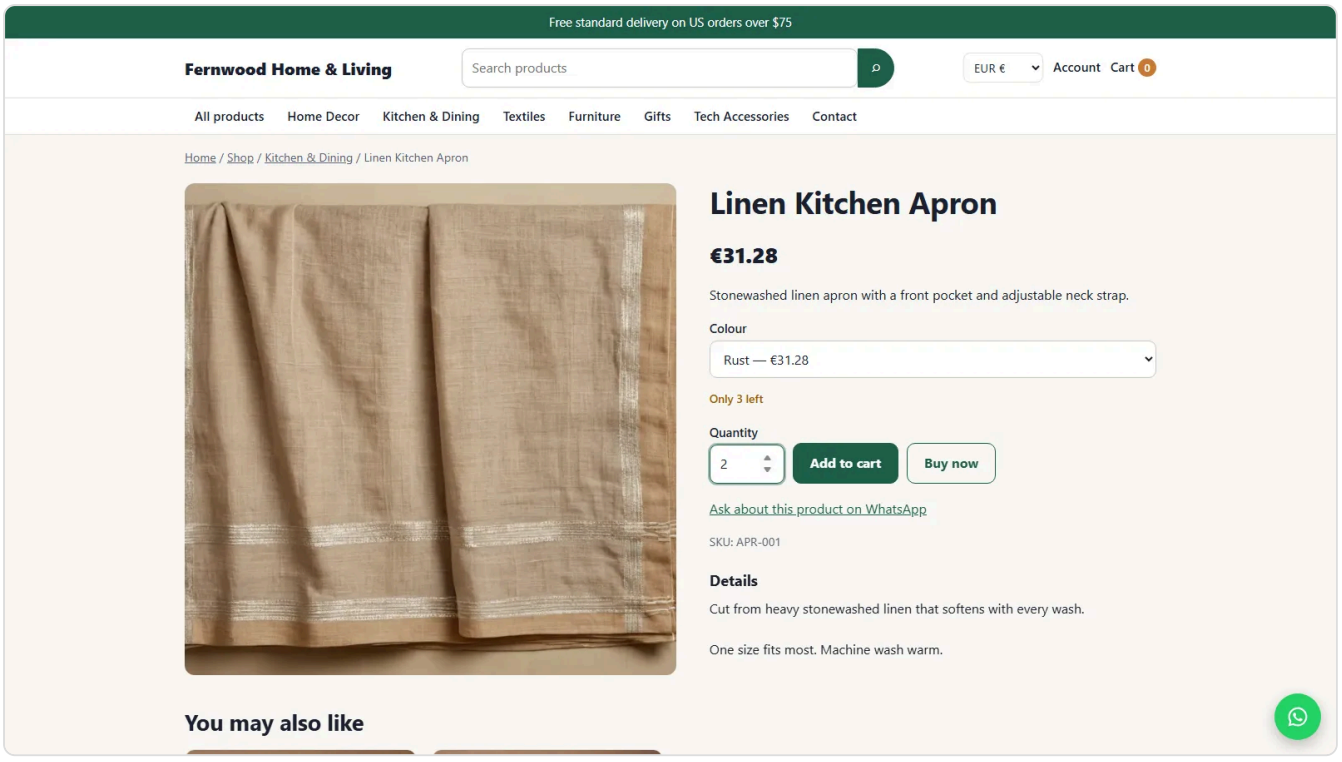
Admin → **Shipping**: create zones from 2-letter country codes (`US, CA`), optionally narrowed to states/regions, and `*` for "everywhere else". Each zone has methods: flat rate, free over an amount, weight table, order-value table (lines like `1:5`, `5:9`, `*:15` = "up to : cost") or local pickup. The customer sees the methods of the most specific matching zone.

Taxes

Admin → **Taxes**: add a rate per country and optionally per region, choose whether your prices already include tax, and enter your tax number for invoices.

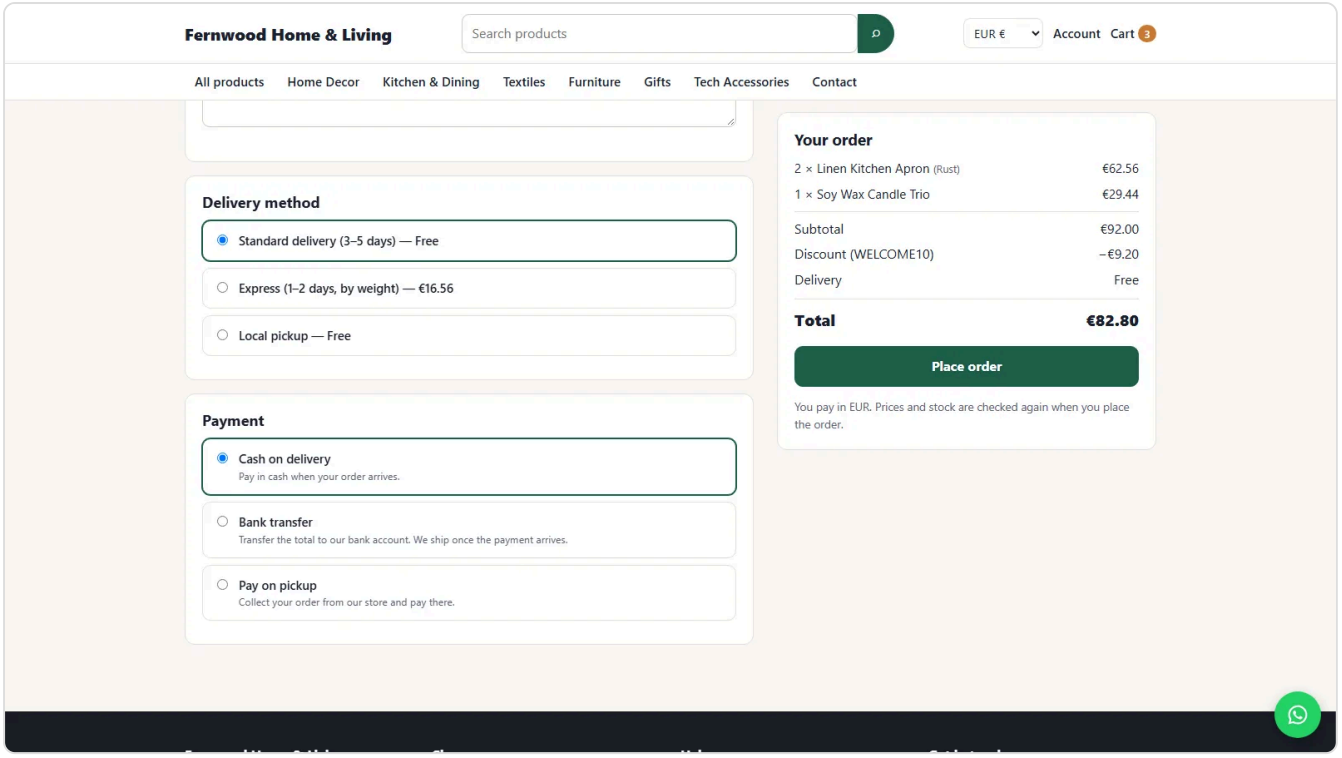
Currencies

Admin → **Currencies**: enable extra currencies with a rate (1 base = rate), symbol, decimals and symbol position. Visitors switch currency in the header. Orders store the currency and rate they were placed with.



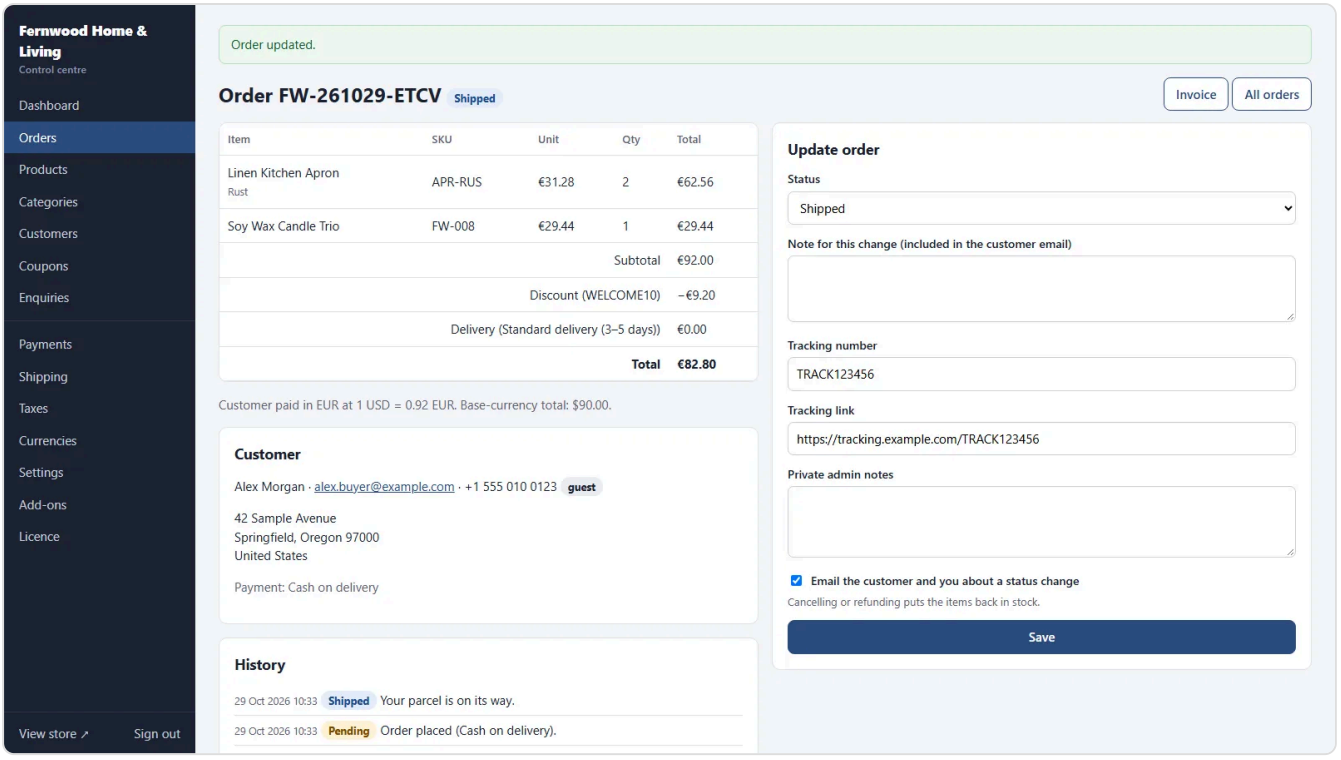
A product with variants, shown in euros after the visitor switched currency.

9. Orders, emails and invoices



Checkout: any-country address form, delivery methods for the address, the payment methods you enabled, and live totals.

Every new order emails the customer and you. Admin → **Orders** lists orders with search, status and date filters, plus **Export CSV**. On an order you can change the status (pending → paid → processing → shipped → delivered, or cancelled / refunded), add a note for the customer, a tracking number and link, and private notes. Each change is logged and emailed. Cancelling or refunding returns the items to stock. **Invoice** opens a printable invoice (print or save as PDF).



Order screen: items, totals in the customer's currency, history, status and tracking.

Email (SMTP)

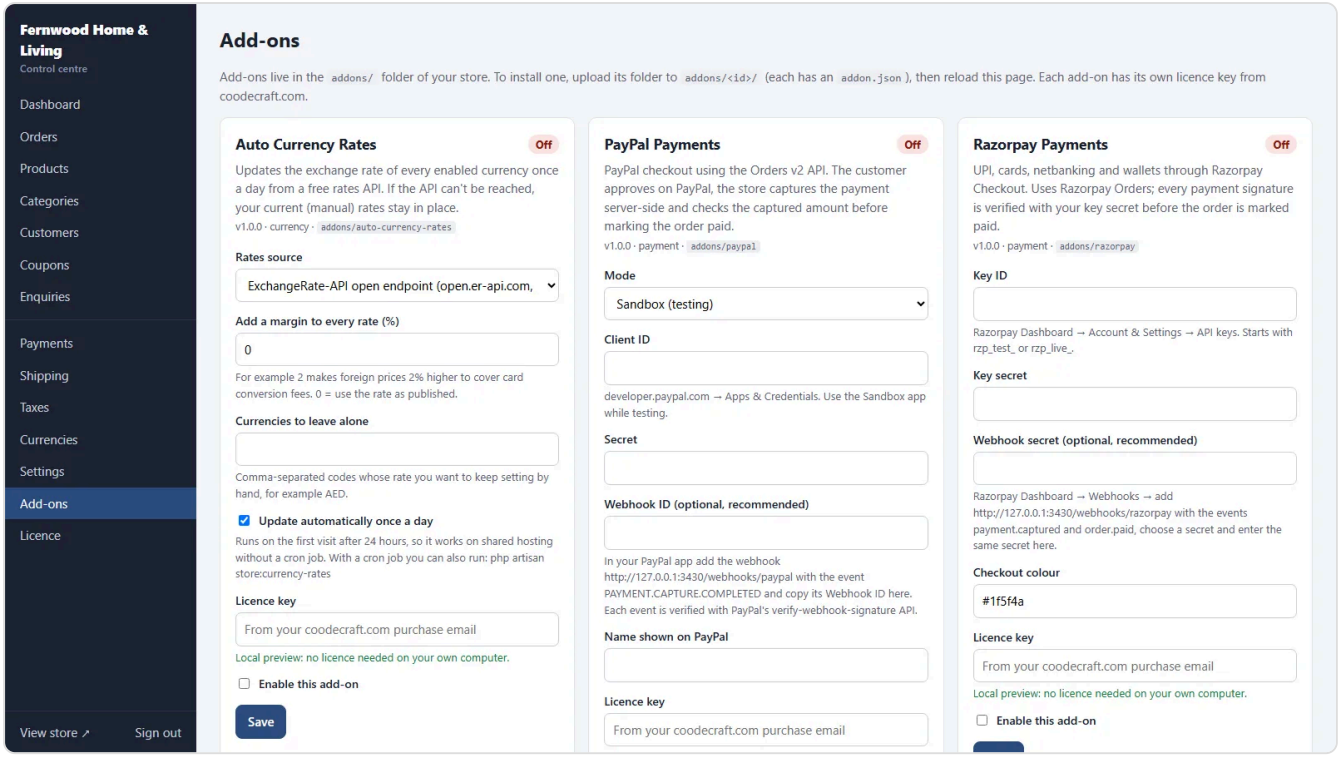
Until you set up email, messages are only written to `storage/logs/laravel.log`. Admin → **Settings** → **Email (SMTP)**: enter your mailbox's SMTP server, port (587 TLS or 465 SSL), username and password (stored encrypted), then use **Send a test email**. On cPanel create a mailbox first (Email Accounts).

10. Chat, WhatsApp, analytics and SEO

Admin → **Settings** → **Chat, WhatsApp & analytics**: live chat (Tawk.to, Crisp, Tidio or your own snippet), the floating WhatsApp button and its message, GA4 / Google Tag Manager / Meta Pixel IDs, the announcement bar and the cookie notice (with it on, analytics load only after the visitor accepts). **SEO & store mode** holds default title and description, social share image, a noindex switch and maintenance mode. **Business details**, **Appearance** and **Pages** hold your name, logo, phone, address, map, hours, social links, colours, home banner and policy pages.

11. Add-ons

Add-ons are folders you upload to `store/addons/<id>/`. They then appear under Admin → **Add-ons** with their own settings form and licence key field. Tick **Enable** and save; for payment add-ons also tick the new method under Admin → Payments.



Add-ons screen with Auto Currency Rates, PayPal, Razorpay and Stripe installed.

Add-on	What it does
Stripe Payments	Stripe Checkout Sessions; orders marked paid from Stripe's signed webhook or a server-side session check.
Razorpay Payments	Razorpay Orders + Checkout; payment signature verified with your key secret; optional signed webhook.
PayPal Payments	PayPal Orders v2; captured server-side and amount checked; webhooks verified with PayPal's API.
Auto Currency Rates	Updates your currency rates daily from a free rates API; manual rates kept if it fails.

> Payment add-ons are written against each provider's official API documentation and were tested with simulated (mocked) responses. Always run a test payment in the provider's test or sandbox mode before switching to live keys.

12. Going live checklist

- `.env`: `APP_ENV=production`, `APP_DEBUG=false`, correct `APP_URL` with https.
- Licence active (Admin → Licence). SSL certificate installed.

- Business details, logo, colours and policy pages replaced (no placeholder address or phone left).
- Shipping zones, tax rates, currencies and payment methods checked; one test order placed and its status changed.
- SMTP set up and the test email received; demo products replaced.
- Back up `.env`, `database/database.sqlite` (or your MySQL database) and `public/uploads/` regularly.

13. Troubleshooting

Problem	Fix
"500 Server Error"	Check `storage/logs/laravel.log`. Usually: `.env` missing, no `APP_KEY` (run key:generate), or `storage/` not writable.
"could not find driver"	Enable pdo_sqlite (or pdo_mysql) in Select PHP Version / PHP Configuration.
Page shows a directory listing or the wrong page	The domain is not pointing at `public/`; see section 3, step 2.
Images upload but don't show	Make `public/uploads` (or `public_html/uploads`) writable.
"Ordering is switched off"	Activate the licence (section 7).
Changes to `.env` ignored	Run `php artisan config:clear` (or delete `bootstrap/cache/config.php`).

Need help? **coodecraft.com/contact** — or order "We set it up" and we install it on your hosting for you.