Squelette Django + HTMX + Tailwind + Alpine + Lucide — Boutique Ebook (CinetPay mock)

Objectif : V1 exploitable d'un site simple façon Acquisition.com (promesse forte, preuves, FAQ, garantie) pour vendre l'ebook **« Audit des services publics »** avec 3 offres (Standard, Personnalisation, Formation), carrousel d'exemples d'irrégularités, paiement **CinetPay** (mock + endpoints prêts), téléchargement sécurisé.

---

0) Quickstart

```
# 1) Crée l'environnement
python -m venv .venv
source .venv/bin/activate   # Windows: .venv\Scripts\activate

# 2) Packages
pip install "Django>=5.0" whitenoise pillow requests python-dotenv

# 3) Démarre le projet
django-admin startproject config .
python manage.py startapp core
python manage.py startapp store

# 4) Colle les fichiers ci-dessous (écrase/ajoute selon l'arborescence)
# 5) Migrations & superuser
python manage.py migrate
python manage.py createsuperuser

# 6) Seed de données exemple
python manage.py seed_store

# 7) Run
python manage.py runserver
```

Prod : activer `DEBUG=False`, configurer `ALLOWED_HOSTS`, servir via `whitenoise`, et envisager Tailwind CLI pour un CSS purgé (cf. note en fin de doc).

---

1) Arborescence visée

```
./
  manage.py
  config/
    __init__.py
    settings.py
    urls.py
    wsgi.py
    asgi.py
  core/
    __init__.py
    urls.py
    views.py
    templates/core/
      home.html
      about.html
      policy.html
      cgv.html
      contact.html
  store/
    __init__.py
    urls.py
    views.py
    models.py
    admin.py
    services/
      __init__.py
      cinetpay.py
    management/commands/
      __init__.py
      seed_store.py
    templates/store/
      base.html
      product_detail.html
      offers.html
      examples.html
      thank_you.html
  media/
  static/
```

---

2) `config/settings.py`

```python
from pathlib import Path
import os

BASE_DIR = Path(__file__).resolve().parent.parent

SECRET_KEY = os.environ.get("DJANGO_SECRET_KEY", "dev-insecure-key")
DEBUG = os.environ.get("DJANGO_DEBUG", "1") == "1"
ALLOWED_HOSTS = ["*"]

INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "core",
    "store",
]

MIDDLEWARE = [
    "django.middleware.security.SecurityMiddleware",
    "whitenoise.middleware.WhiteNoiseMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
    "django.middleware.common.CommonMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "django.contrib.auth.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "django.middleware.clickjacking.XFrameOptionsMiddleware",
]

ROOT_URLCONF = "config.urls"

TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "DIRS": [BASE_DIR / "templates"],
        "APP_DIRS": True,
        "OPTIONS": {
            "context_processors": [
                "django.template.context_processors.debug",
                "django.template.context_processors.request",
                "django.contrib.auth.context_processors.auth",
                "django.contrib.messages.context_processors.messages",
            ],
        },
    },
]

WSGI_APPLICATION = "config.wsgi.application"
ASGI_APPLICATION = "config.asgi.application"

DATABASES = {
    "default": {
        "ENGINE": "django.db.backends.sqlite3",
        "NAME": BASE_DIR / "db.sqlite3",
    }
}

AUTH_PASSWORD_VALIDATORS = []

LANGUAGE_CODE = "fr-fr"
TIME_ZONE = "Africa/Bamako"
```

```
USE_I18N = True
USE_TZ = True

STATIC_URL = "/static/"
STATIC_ROOT = BASE_DIR / "staticfiles"
STATICFILES_DIRS = [BASE_DIR / "static"]
STATICFILES_STORAGE =
"whitenoise.storage.CompressedManifestStaticFilesStorage"

MEDIA_URL = "/media/"
MEDIA_ROOT = BASE_DIR / "media"

DEFAULT_AUTO_FIELD = "django.db.models.BigAutoField"
```

---

3) `config/urls.py`

```
from django.contrib import admin
from django.urls import path, include
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path("admin/", admin.site.urls),
    path("", include("core.urls")),
    path("", include("store.urls")),
]

if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL,
document_root=settings.MEDIA_ROOT)
```

---

4) App core

`core/urls.py`

```
from django.urls import path
from . import views

app_name = "core"

urlpatterns = [
    path("", views.home, name="home"),
    path("a-propos/", views.about, name="about"),
    path("politique/", views.policy, name="policy"),
    path("cgv/", views.cgv, name="cgv"),
    path("contact/", views.contact, name="contact"),
]
```

`core/views.py`

```python
from django.shortcuts import render
from store.models import Product, OfferTier, ExampleSlide

def home(request):
    product = Product.objects.filter(is_published=True).first()
    offers = OfferTier.objects.select_related("product").all()
    examples = ExampleSlide.objects.all()[:3]
    return render(request, "core/home.html", {"product": product, "offers":
offers, "examples": examples})

def about(request):
    return render(request, "core/about.html")

def policy(request):
    return render(request, "core/policy.html")

def cgv(request):
    return render(request, "core/cgv.html")

def contact(request):
    return render(request, "core/contact.html")
```

`core/templates/core/home.html`

```html
{% extends "store/base.html" %}
{% block title %}Accueil{% endblock %}
{% block content %}
<section class="py-16">
  <div class="max-w-5xl mx-auto px-4 text-center">
    <h1 class="text-3xl md:text-5xl font-bold tracking-tight">Préparez votre
structure à l’audit — sans stress ni surprises.</h1>
    <p class="mt-4 text-gray-600">Un ebook concret, des offres
personnalisées, des exemples réels.</p>
    <div class="mt-8 flex items-center justify-center gap-3">
      {% if product %}
      <a href="{% url 'store:product_detail' product.slug %}" class="px-6
py-3 rounded-2xl bg-blue-600 text-white">Voir le livre</a>
      {% endif %}
      <a href="{% url 'store:offers' %}" class="px-6 py-3 rounded-2xl
border">Voir les offres</a>
    </div>
  </div>
</section>
```

```
<section class="py-10 bg-gray-50">
  <div class="max-w-6xl mx-auto px-4">
    <h2 class="text-xl font-semibold mb-6">Exemples d’irrégularités</h2>
    <div class="grid md:grid-cols-3 gap-4">
      {% for ex in examples %}
      <article class="rounded-2xl border bg-white p-4">
        <h3 class="font-semibold">{{ ex.title }}</h3>
        <p class="text-sm mt-2"><strong>Irrégularité :</strong>
{{ ex.irregularity }}</p>
        <p class="text-sm mt-1"><strong>Traitement :</strong> {{ ex.remedy }}
</p>
      </article>
      {% endfor %}
    </div>
    <div class="mt-6"><a href="{% url 'store:examples' %}" class="text-
blue-700">Voir plus →</a></div>
  </div>
</section>
{% endblock %}
```

Pages statiques simples

`core/templates/core/about.html`, `policy.html`, `cgv.html`, `contact.html` → démarrez avec des sections basiques (contenu à insérer).

---

5) App store

`store/models.py`

```
from django.db import models
from django.utils import timezone
import uuid

class Product(models.Model):
    slug = models.SlugField(unique=True)
    title = models.CharField(max_length=200)
    subtitle = models.CharField(max_length=300, blank=True)
    price_fcfa = models.PositiveIntegerField(default=15000)
    hero_image = models.ImageField(upload_to="products/", blank=True,
null=True)
    guarantee_text = models.TextField(blank=True)
    faq_json = models.JSONField(default=list, blank=True)
    social_proofs_json = models.JSONField(default=list, blank=True)
    deliverable_file = models.FileField(upload_to="deliverables/",
blank=True, null=True)
    is_published = models.BooleanField(default=False)
    created_at = models.DateTimeField(auto_now_add=True)
    def __str__(self):
        return self.title

class OfferTier(models.Model):
    STANDARD = "STANDARD"
    PERSONNALISATION = "PERSONNALISATION"
    FORMATION = "FORMATION"
    KIND_CHOICES = [
        (STANDARD, "Standard"),
        (PERSONNALISATION, "Personnalisation"),
        (FORMATION, "Formation & Assistance"),
    ]
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    kind = models.CharField(max_length=32, choices=KIND_CHOICES)
    price_fcfa = models.PositiveIntegerField(blank=True, null=True)
    description_md = models.TextField(blank=True)
    cta_type = models.CharField(max_length=10, default="BUY")  # BUY|QUOTE|
CALL

    class Meta:
        unique_together = ("product", "kind")

    def __str__(self):
        return f"{self.product.title} - {self.kind}"

class ExampleSlide(models.Model):
    product = models.ForeignKey(Product, on_delete=models.CASCADE,
related_name="examples")
    title = models.CharField(max_length=200)
    irregularity = models.TextField()
    indicators = models.TextField(blank=True)
    legal_ref = models.CharField(max_length=300, blank=True)
    remedy = models.TextField()
    risks = models.TextField(blank=True)
    sample_doc_url = models.URLField(blank=True)
    image = models.ImageField(upload_to="examples/", blank=True, null=True)

    def __str__(self):
        return self.title

class MediaAsset(models.Model):
    PDF_EXTRACT = "PDF_EXTRACT"
    VIDEO = "VIDEO"
    KIND_CHOICES = [(PDF_EXTRACT, "PDF extrait"), (VIDEO, "Vidéo")]
    product = models.ForeignKey(Product, on_delete=models.CASCADE,
related_name="media")
    kind = models.CharField(max_length=20, choices=KIND_CHOICES)
    title = models.CharField(max_length=200)
    file_or_url = models.CharField(max_length=500)
    thumb = models.ImageField(upload_to="media/thumbs/", blank=True,
null=True)
```

```
    def __str__(self):
        return f"{self.product.title} - {self.kind}: {self.title}"

class Order(models.Model):
    PENDING = "pending"
    PAID = "paid"
    FAILED = "failed"
    STATUS_CHOICES = [(PENDING, "En attente"), (PAID, "Payé"), (FAILED,
"Échec")]

    order_id = models.UUIDField(default=uuid.uuid4, editable=False,
unique=True)
    product = models.ForeignKey(Product, on_delete=models.PROTECT)
    email = models.EmailField()
    amount_fcfa = models.PositiveIntegerField()
    status = models.CharField(max_length=10, choices=STATUS_CHOICES,
default=PENDING)
    cinetpay_payment_id = models.CharField(max_length=100, blank=True)
    created_at = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return f"Order {self.order_id} - {self.status}"

class DownloadToken(models.Model):
    order = models.OneToOneField(Order, on_delete=models.CASCADE)
    token = models.UUIDField(default=uuid.uuid4, editable=False, unique=True)
    expires_at = models.DateTimeField(default=lambda: timezone.now() +
timezone.timedelta(hours=72))

    def is_valid(self):
        return timezone.now() < self.expires_at

    def __str__(self):
        return f"Token for {self.order.order_id}"
```

`store/admin.py`

```
from django.contrib import admin
from .models import Product, OfferTier, ExampleSlide, MediaAsset, Order,
DownloadToken

@admin.register(Product)
class ProductAdmin(admin.ModelAdmin):
    list_display = ("title", "price_fcfa", "is_published")
    prepopulated_fields = {"slug": ("title",)}

admin.site.register(OfferTier)
admin.site.register(ExampleSlide)
admin.site.register(MediaAsset)
admin.site.register(Order)
admin.site.register(DownloadToken)
```

`store/services/cinetpay.py` **(mock prêt pour branchement réel)**

```
"""
Service CinetPay — MOCK pour dev.
Pour la prod, remplacer init_payment() par l'appel API réel + vérif de
signature HMAC côté webhook.
"""
from urllib.parse import urlencode

# URL de retour simulée (succès). En prod, renvoyer l'URL CinetPay.
RETURN_PATH = "/payment/return/"


def init_payment(order, request):
    # MOCK: simule une redirection de paiement réussie
    params = urlencode({"status": "success", "order": str(order.order_id)})
    return f"{RETURN_PATH}?{params}"


def verify_webhook(request):
    # MOCK: en prod, vérifier la signature HMAC envoyée par CinetPay
    return True
```

`store/urls.py`

```
from django.urls import path
from . import views

app_name = "store"

urlpatterns = [
    path("ebook/<slug:slug>/", views.product_detail, name="product_detail"),
    path("offres/", views.offers, name="offers"),
    path("exemples/", views.examples, name="examples"),

    path("buy/<slug:slug>/", views.buy, name="buy"),
    path("payment/return/", views.payment_return, name="payment_return"),
    path("payment/callback/", views.payment_callback,
name="payment_callback"),
    path("telecharger/<uuid:token>/", views.download, name="download"),
]
```

store/views.py

```python
from django.shortcuts import get_object_or_404, render, redirect
from django.http import HttpResponse, FileResponse, Http404
from django.utils import timezone
from .models import Product, OfferTier, ExampleSlide, MediaAsset, Order,
DownloadToken
from .services import cinetpay
import io

# --- Pages ---

def product_detail(request, slug):
    product = get_object_or_404(Product, slug=slug, is_published=True)
    media = MediaAsset.objects.filter(product=product)
    faqs = product.faq_json or []
    proofs = product.social_proofs_json or []
    return render(request, "store/product_detail.html", {
        "product": product, "media": media, "faqs": faqs, "proofs": proofs
    })


def offers(request):
    product = Product.objects.filter(is_published=True).first()
    tiers = OfferTier.objects.filter(product=product) if product else []
    return render(request, "store/offers.html", {"product": product, "tiers":
tiers})


def examples(request):
    product = Product.objects.filter(is_published=True).first()
    slides = ExampleSlide.objects.filter(product=product) if product else []
    return render(request, "store/examples.html", {"slides": slides,
"product": product})

# --- Paiement ---

def buy(request, slug):
    if request.method != "POST":
        return redirect("store:product_detail", slug=slug)

    product = get_object_or_404(Product, slug=slug, is_published=True)
    email = request.POST.get("email") or request.GET.get("email") or
"client@example.com"

    order = Order.objects.create(
        product=product,
        email=email,
        amount_fcfa=product.price_fcfa,
    )
    payment_url = cinetpay.init_payment(order, request)
    return redirect(payment_url)


def payment_return(request):
    status = request.GET.get("status")
    order_id = request.GET.get("order")
    order = Order.objects.filter(order_id=order_id).first()
    if not order:
        return render(request, "store/thank_you.html", {"ok": False,
"message": "Commande introuvable."})

    if status == "success":
        order.status = Order.PAID
        order.save(update_fields=["status"])
        token, _ = DownloadToken.objects.get_or_create(order=order)
        context = {"ok": True, "order": order, "token": token}
    else:
        order.status = Order.FAILED
        order.save(update_fields=["status"])
        context = {"ok": False, "order": order, "message": "Paiement non
confirmé."}

    return render(request, "store/thank_you.html", context)


def payment_callback(request):
    # Webhook CinetPay (appel serveur→serveur)
    if request.method != "POST":
        return HttpResponse("Method not allowed", status=405)

    if not cinetpay.verify_webhook(request):
        return HttpResponse("Invalid signature", status=400)

    # TODO: lire payload, retrouver la commande, passer en PAID si confirmé
    return HttpResponse("OK")


def download(request, token):
    dt = DownloadToken.objects.filter(token=token).select_related("order",
"order__product").first()
    if not dt or not dt.is_valid():
        raise Http404("Lien de téléchargement invalide ou expiré.")

    product = dt.order.product
    f = product.deliverable_file
    if not f:
        raise Http404("Fichier non disponible.")

    response = FileResponse(f.open("rb"), as_attachment=True,
filename=f.name.split("/")[-1])
    return response
```

---

6) Templates — base & pages clés

`store/templates/store/base.html`

```
<!doctype html>
<html lang="fr" class="h-full">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>{% block title %}Boutique{% endblock %} – Audit des services
publics</title>
  <meta name="description" content="Ebook et offres personnalisées pour se
préparer aux missions d'audit." />

  <!-- Tailwind CDN (dev). En prod, utiliser Tailwind CLI + fichiers
statiques. -->
  <script src="https://cdn.tailwindcss.com"></script>
  <script>tailwind.config = { theme: { extend: { borderRadius: { '2xl':
'1rem' }}}}</script>

  <!-- Alpine.js -->
  <script defer src="https://unpkg.com/alpinejs@3.x.x/dist/cdn.min.js"></
script>
  <!-- htmx -->
  <script src="https://unpkg.com/htmx.org@1.9.12"></script>
  <!-- Lucide -->
  <script src="https://unpkg.com/lucide@latest"></script>

  <script>
    document.addEventListener('DOMContentLoaded', () => { if (window.lucide)
lucide.createIcons(); });
    // CSRF pour htmx
    document.addEventListener('htmx:configRequest', (event) => {
      const csrftoken = document.cookie.split('; ').find(row =>
row.startsWith('csrftoken='))?.split('=')[1];
      if (csrftoken) event.detail.headers['X-CSRFToken'] = csrftoken;
    });
  </script>
</head>
<body class="min-h-full bg-white text-gray-900">
  <header class="sticky top-0 z-40 bg-white/90 backdrop-blur border-b">
    <div class="max-w-6xl mx-auto px-4 py-3 flex items-center justify-
between">
      <a href="/" class="font-bold">Audit Ebooks</a>
      <nav class="flex items-center gap-4 text-sm">
```

```
        <a href="{% url 'store:offers' %}" class="hover:text-
blue-700">Offres</a>
        <a href="{% url 'store:examples' %}" class="hover:text-
blue-700">Exemples</a>
        <a href="{% url 'core:about' %}" class="hover:text-blue-700">À
propos</a>
      </nav>
    </div>
  </header>

  <main class="min-h-[70vh]">{% block content %}{% endblock %}</main>

  <footer class="border-t mt-16">
    <div
class="max-w-6xl mx-auto px-4 py-8 grid md:grid-cols-3 gap-6 text-sm text-
gray-600">
      <div>
        <div class="font-semibold">Audit Ebooks</div>
        <p class="mt-2">Préparez-vous efficacement aux missions d’audit.</p>
      </div>
      <div>
        <div class="font-semibold">Liens</div>
        <ul class="mt-2 space-y-1">
          <li><a href="{% url 'core:policy' %}" class="hover:text-
blue-700">Politique</a></li>
          <li><a href="{% url 'core:cgv' %}" class="hover:text-
blue-700">CGV</a></li>
          <li><a href="{% url 'core:contact' %}" class="hover:text-
blue-700">Contact</a></li>
        </ul>
      </div>
      <div>
        <div class="font-semibold">Confiance</div>
        <ul class="mt-2 space-y-1">
          <li class="flex items-center gap-2"><i data-lucide="shield"></i>
Paiement sécurisé (CinetPay)</li>
          <li class="flex items-center gap-2"><i data-lucide="check-
circle"></i> Garantie 7 jours</li>
        </ul>
      </div>
    </div>
  </footer>
</body>
</html>
```

`store/templates/store/product_detail.html`

```
{% extends "store/base.html" %}
{% block title %}{{ product.title }}{% endblock %}
{% block content %}
```

```
<section class="py-14">
  <div class="max-w-5xl mx-auto px-4 grid md:grid-cols-2 gap-8 items-center">
    <div>
      <h1 class="text-3xl md:text-4xl font-bold">{{ product.title }}</h1>
      <p class="mt-3 text-gray-600">{{ product.subtitle }}</p>
      <div class="mt-6 flex items-center gap-3">
        <form hx-post="{% url 'store:buy' product.slug %}">
          <button class="px-6 py-3 rounded-2xl bg-blue-600 text-white">Obtenir
l’ebook — {{ product.price_fcfa }} FCFA</button>
        </form>
        <a href="{% url 'store:offers' %}" class="px-6 py-3 rounded-2xl
border">Voir les offres</a>
      </div>
      <ul class="mt-6 space-y-2 text-sm">
        <li class="flex gap-2"><i data-lucide="check"></i> Bonus inclus :
Dossier irrégularités & solutions (1 texte au choix)</li>
        <li class="flex gap-2"><i data-lucide="check"></i> Guide : L’audit en
7 étapes</li>
        <li class="flex gap-2"><i data-lucide="lock"></i> Téléchargement
sécurisé (lien 72h)</li>
      </ul>
    </div>
    <div>
      {% if product.hero_image %}
        <img src="{{ product.hero_image.url }}" alt="Couverture"
class="rounded-2xl border" />
      {% endif %}
    </div>
  </div>
</section>

<section class="py-10 bg-gray-50">
  <div class="max-w-5xl mx-auto px-4">
    <h2 class="text-xl font-semibold">Ce que vous allez obtenir</h2>
    <div class="mt-4 grid md:grid-cols-3 gap-4">
      <div class="rounded-2xl border bg-white p-4"><i data-lucide="book-
open"></i> Ebook complet</div>
      <div class="rounded-2xl border bg-white p-4"><i data-lucide="file-
text"></i> Dossier irrégularités (personnalisé)</div>
      <div class="rounded-2xl border bg-white p-4"><i data-lucide="list-
checks"></i> Guide 7 étapes</div>
    </div>
  </div>
</section>

<section class="py-10">
  <div class="max-w-5xl mx-auto px-4">
    <h2 class="text-xl font-semibold">Media / Extraits</h2>
    <div class="mt-4 grid md:grid-cols-2 gap-4">
      {% for m in media %}
        <div class="rounded-2xl border p-4">
```

```
          <div class="font-medium">{{ m.title }}</div>
          {% if m.kind == 'PDF_EXTRACT' %}
            <a href="{{ m.file_or_url }}" target="_blank" class="text-
blue-700 text-sm">Ouvrir l’extrait PDF →</a>
          {% elif m.kind == 'VIDEO' %}
            <div class="aspect-video mt-2">
              <iframe src="{{ m.file_or_url }}" class="w-full h-full rounded-
xl" allowfullscreen></iframe>
            </div>
          {% endif %}
        </div>
      {% empty %}
        <p class="text-gray-600 text-sm">Extraits à venir.</p>
      {% endfor %}
    </div>
  </div>
</section>

<section class="py-10 bg-gray-50">
  <div class="max-w-5xl mx-auto px-4">
    <h2 class="text-xl font-semibold">FAQ</h2>
    <div class="mt-4 space-y-3">
      {% for item in faqs %}
      <div x-data="{open:false}" class="border rounded-2xl p-4">
        <button @click="open=!open" class="w-full flex justify-between">
          <span class="font-medium">{{ item.q }}</span>
          <i data-lucide="chevron-down" :class="{'rotate-180':open}"
class="transition"></i>
        </button>
        <div x-show="open" x-collapse class="mt-2 text-sm text-
gray-600">{{ item.a }}</div>
      </div>
      {% endfor %}
    </div>
  </div>
</section>

<section class="py-10">
  <div class="max-w-5xl mx-auto px-4">
    <div class="rounded-2xl border p-6 flex items-center justify-between">
      <div>
        <div class="font-semibold">Garantie 7 jours</div>
        <p class="text-sm text-gray-600">Satisfait ou remboursé si le contenu
ne vous aide pas.</p>
      </div>
      <form hx-post="{% url 'store:buy' product.slug %}">
        <button class="px-6 py-3 rounded-2xl bg-blue-600 text-white">Obtenir
l’ebook — {{ product.price_fcfa }} FCFA</button>
      </form>
    </div>
  </div>
```

```
</section>

<!-- Bandeau CTA sticky (mobile) -->
<div class="fixed bottom-0 inset-x-0 md:hidden bg-white border-t p-3 flex
items-center justify-between">
  <div class="font-semibold">{{ product.price_fcfa }} FCFA</div>
  <form hx-post="{% url 'store:buy' product.slug %}">
    <button class="px-4 py-2 rounded-xl bg-blue-600 text-white">Obtenir
l’ebook</button>
  </form>
</div>
{% endblock %}
```

store/templates/store/offers.html

```
{% extends "store/base.html" %}
{% block title %}Offres{% endblock %}
{% block content %}
<section class="py-14">
  <div class="max-w-6xl mx-auto px-4">
    <h1 class="text-3xl font-bold">Nos offres</h1>
    <div class="mt-8 grid md:grid-cols-3 gap-6">
      {% for t in tiers %}
      <div class="rounded-2xl border p-6">
        <div class="text-sm text-gray-500">{{ t.get_kind_display }}</div>
        <div class="mt-2 font-semibold">{{ product.title }}</div>
        {% if t.price_fcfa %}
          <div class="mt-2 text-2xl font-bold">{{ t.price_fcfa }} FCFA</div>
        {% else %}
          <div class="mt-2 text-2xl font-bold">Sur devis</div>
        {% endif %}
        <div class="prose prose-sm mt-4">{{ t.description_md|linebreaks }}</
div>
        <div class="mt-6">
          {% if t.cta_type == 'BUY' %}
            <form hx-post="{% url 'store:buy' product.slug %}"><button
class="px-5 py-3 rounded-xl bg-blue-600 text-white w-full">Acheter</button></
form>
          {% elif t.cta_type == 'QUOTE' %}
            <a href="{% url 'core:contact' %}" class="px-5 py-3 rounded-xl
border w-full inline-flex justify-center">Demander un devis</a>
          {% else %}
            <a href="{% url 'core:contact' %}" class="px-5 py-3 rounded-xl
border w-full inline-flex justify-center">Planifier un appel</a>
          {% endif %}
        </div>
      </div>
      {% endfor %}
    </div>
  </div>
```

```
</section>
{% endblock %}
```

`store/templates/store/examples.html`

```
{% extends "store/base.html" %}
{% block title %}Exemples d’irrégularités{% endblock %}
{% block content %}
<section class="py-14">
  <div class="max-w-5xl mx-auto px-4">
    <h1 class="text-3xl font-bold">Exemples d’irrégularités & traitements</
h1>

    <div x-data="{i:0, n: {{ slides|length }}}" class="mt-8">
      <div class="relative rounded-2xl border p-6 bg-white">
        {% for s in slides %}
        <div x-show="i==={{ forloop.counter0 }}" x-transition>
          <h2 class="text-xl font-semibold">{{ s.title }}</h2>
          <p class="mt-2 text-sm"><strong>Irrégularité :</strong>
{{ s.irregularity }}</p>
          {% if s.indicators %}<p class="mt-1 text-
sm"><strong>Indicateurs :</strong> {{ s.indicators }}</p>{% endif %}
          {% if s.legal_ref %}<p class="mt-1 text-sm"><strong>Référence :</
strong> {{ s.legal_ref }}</p>{% endif %}
          <p class="mt-1 text-sm"><strong>Traitement recommandé :</strong>
{{ s.remedy }}</p>
          {% if s.risks %}<p class="mt-1 text-sm"><strong>Risques :</strong>
{{ s.risks }}</p>{% endif %}
          {% if s.sample_doc_url %}<a class="mt-2 inline-block text-blue-700
text-sm" href="{{ s.sample_doc_url }}" target="_blank">Voir un exemple de
pièce →</a>{% endif %}
        </div>
        {% endfor %}
        <div class="mt-6 flex items-center gap-2">
          <button @click="i=(i-1+n)%n" class="px-3 py-2 rounded-lg
border">Préc.</button>
          <button @click="i=(i+1)%n" class="px-3 py-2 rounded-lg
border">Suiv.</button>
        </div>
      </div>
    </div>

    <div class="mt-8 flex gap-3">
      {% if product %}<a href="{% url 'store:product_detail' product.slug %}"
class="px-5 py-3 rounded-xl bg-blue-600 text-white">Voir le livre</a>{% endif
%}
      <a href="{% url 'store:offers' %}" class="px-5 py-3 rounded-xl border">Demander
une personnalisation</a>
    </div>
  </div>
```

```
</section>
{% endblock %}
```

`store/templates/store/thank_you.html`

```
{% extends "store/base.html" %}
{% block title %}Merci{% endblock %}
{% block content %}
<section class="py-14">
  <div class="max-w-md mx-auto px-4 text-center">
    {% if ok %}
      <h1 class="text-2xl font-bold">Merci pour votre achat !</h1>
      <p class="mt-2 text-gray-600">Votre paiement est confirmé.</p>
      <a href="{% url 'store:download' token.token %}" class="mt-6 inline-
block px-6 py-3 rounded-2xl bg-blue-600 text-white">Télécharger l’ebook</a>
    {% else %}
      <h1 class="text-2xl font-bold">Paiement non confirmé</h1>
      <p class="mt-2 text-gray-600">{{ message }}</p>
    {% endif %}
  </div>
</section>
{% endblock %}
```

---

7) Seed de données — `store/management/commands/seed_store.py`

```
from django.core.management.base import BaseCommand
from store.models import Product, OfferTier, ExampleSlide, MediaAsset

class Command(BaseCommand):
    help = "Seed produit/tiers/exemples"

    def handle(self, *args, **options):
        product, _ = Product.objects.get_or_create(
            slug="audit-services-publics",
            defaults=dict(
                title="Audit des services publics",
                subtitle="Le guide concret pour éviter les mauvaises
surprises.",
                price_fcfa=15000,
                guarantee_text="Satisfait ou remboursé 7 jours.",
                faq_json=[
                    {"q": "Format et mises à jour ?", "a":
"PDF, mises à jour incluses pendant 3 mois."},
                    {"q": "Remboursement ?", "a": "Oui, sous 7 jours si le
contenu ne vous aide pas."},
                ],
                social_proofs_json=[
                    {"t": "Directeur Financier", "q": "Clair et
actionnable."},
                    {"t": "Auditeur Interne", "q": "Nous a fait gagner des
semaines."},
                ],
                is_published=True,
            ),
        )

        # Offres
        OfferTier.objects.get_or_create(product=product,
kind=OfferTier.STANDARD, defaults=dict(
            price_fcfa=15000,

description_md="Ebook + Dossier irrégularités (1 texte) + Guide 7 étapes.",
            cta_type="BUY",
        ))
        OfferTier.objects.get_or_create(product=product,
kind=OfferTier.PERSONNALISATION, defaults=dict(
            price_fcfa=None,
            description_md="Adaptation à votre structure (textes,
organisation, circuits). Prix au nombre de pages traitées.",
            cta_type="QUOTE",
        ))
        OfferTier.objects.get_or_create(product=product,
kind=OfferTier.FORMATION, defaults=dict(
            price_fcfa=None,
            description_md="Cas pratiques, cahier du participant, assistance
au suivi des recommandations.",
            cta_type="CALL",
        ))

        # Exemples
        ExampleSlide.objects.get_or_create(product=product,
title="Fractionnement de marchés", defaults=dict(
            irregularity="Découpage artificiel pour éviter les seuils.",
            indicators="Plusieurs bons de commande rapprochés, mêmes
fournisseurs, mêmes objets.",
            legal_ref="Code des marchés – seuils UEMOA.",
            remedy="Consolider les besoins dans le PPM, respect des seuils,
visa CF.",
            risks="Annulation de procédure, sanctions, perte de confiance
bailleurs.",
        ))
        ExampleSlide.objects.get_or_create(product=product,
title="Rapprochements bancaires en retard", defaults=dict(
            irregularity="Rapprochements effectués hors délais.",
            indicators="Absence de signatures, écarts non justifiés.",
            legal_ref="Règlementation comptable du secteur public.",
            remedy="Rapprochement mensuel signé, justification des écarts.",
            risks="Erreurs non détectées, risques de fraude.",
        ))
        ExampleSlide.objects.get_or_create(product=product, title="Pièces
justificatives manquantes", defaults=dict(
            irregularity="Dépenses sans pièces complètes.",
            indicators="Manque de factures, bons de commande/ livraison.",
            legal_ref="Textes nationaux + procédures bailleurs.",
            remedy="Check-list pièces; dossier unique; archivage.",
            risks="Désengagement bailleurs, sanctions disciplinaires.",
        ))

        # Media (extrait PDF + vidéo démo YouTube)
        MediaAsset.objects.get_or_create(product=product,
kind=MediaAsset.PDF_EXTRACT, title="Extrait (7 pages)", defaults=dict(
            file_or_url="https://example.com/extrait.pdf",
        ))
        MediaAsset.objects.get_or_create(product=product,
kind=MediaAsset.VIDEO, title="Présentation 2 min", defaults=dict(
            file_or_url="https://www.youtube.com/embed/dQw4w9WgXcQ",
        ))

        self.stdout.write(self.style.SUCCESS("Seed OK."))
```

---

8) Notes d'intégration CinetPay (prod)

- Stocker `CINETPAY_API_KEY`, `CINETPAY_SECRET` dans `.env` (utiliser `python-dotenv`) et les lire en settings ou dans `services/cinetpay.py`.
- `init_payment(order)` → appeler l'API CinetPay pour créer une transaction, récupérer l'URL de paiement et rediriger.
- `payment_callback` (webhook) → vérifier **signature HMAC**, mettre `Order.status='paid'`, générer `DownloadToken`.
- `payment_return` (retour navigateur) → lire l'état, afficher merci/téléchargement si déjà confirmé (sinon message d'attente).

---

9) Tailwind en production (optionnel mais recommandé)

1. `npm init -y && npm i -D tailwindcss postcss autoprefixer`
2. `npx tailwindcss init -p` → configurer `content` pour scanner `templates/**/*.html`.
3. `static/src/input.css` :

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

4. Build : `npx tailwindcss -i ./static/src/input.css -o ./static/dist/styles.css --minify`
5. Remplacer le CDN par `<link rel="stylesheet" href="{% static 'dist/styles.css' %}">` dans `base.html`.
6. Exécuter `collectstatic` pour la prod.

---

10) Prochaines améliorations

- Watermark (email acheteur) sur le PDF délivré
- Coupons/bundles, multi-produits
- Pages légales enrichies (mentions, RGPD si nécessaire)
- Emails transactionnels (order paid, lien de téléchargement)
- Tests unitaires (services, webhook) ```