# DevOpsVaultX – Zero to Hero Setup Guide

## 1. Overview

**DevOpsVaultX** is a digital platform to sell **DevOps learning resources**:

- Categories: 1) Notes (PDF) 2) Interview Q/A (PDF)
- Tech Stack:
- Backend: Django
- Database: PostgreSQL
- Payment: Razorpay
- Frontend: Django Templates (HTML + CSS)

This guide will take you **from zero to hero**, ready-to-launch with **PDF digital products** and **online payments**.

---

## 2. Prerequisites

- Python 3.10+
- PostgreSQL installed & running
- Django installed (`pip install django`)
- Razorpay account for payments
- Basic HTML/CSS knowledge

---

## 3. Project Setup

```
# Create Django project
django-admin startproject devopsvaultx
cd devopsvaultx

# Create products app
python manage.py startapp products

# Install required packages
pip install psycopg2-binary razorpay
```

---

## 4. PostgreSQL Database Setup

```
-- Create Database
CREATE DATABASE devopsvaultx_db;

-- Create User
CREATE USER devopsuser WITH PASSWORD 'password';

-- Grant Privileges
GRANT ALL PRIVILEGES ON DATABASE devopsvaultx_db TO devopsuser;
```

Update `settings.py`:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'devopsvaultx_db',
        'USER': 'devopsuser',
        'PASSWORD': 'password',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

---

## 5. Models

```
# products/models.py
from django.db import models

CATEGORY_CHOICES = [
    ('notes', 'Notes'),
    ('interview', 'Interview Q/A'),
]

class Product(models.Model):
    title = models.CharField(max_length=255)
    description = models.TextField()
    category = models.CharField(max_length=50, choices=CATEGORY_CHOICES)
    price = models.DecimalField(max_digits=6, decimal_places=2)
    pdf_file = models.FileField(upload_to='pdfs/')
    created_at = models.DateTimeField(auto_now_add=True)
    def __str__(self):
        return self.title

class Order(models.Model):
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    customer_email = models.EmailField()
    razorpay_payment_id = models.CharField(max_length=100)
    created_at = models.DateTimeField(auto_now_add=True)
```

---

## 6. Admin Setup

```
# products/admin.py
from django.contrib import admin
from .models import Product, Order

admin.site.register(Product)
admin.site.register(Order)
```

- Upload Notes / Interview PDFs via Django Admin.

---

## 7. URLs & Views

```
# products/urls.py
from django.urls import path
from . import views

urlpatterns = [
    path('', views.home, name='home'),
    path('category/<str:category>/', views.product_list, name='product_list'),
    path('product/<int:pk>/', views.product_detail, name='product_detail'),
    path('checkout/<int:pk>/', views.checkout, name='checkout'),
    path('success/', views.checkout_success, name='checkout_success'),
]

# devopsvaultx/urls.py
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('products.urls')),
]
```

```
# products/views.py
from django.shortcuts import render, get_object_or_404, redirect
from .models import Product, Order
import razorpay
from django.conf import settings

client = razorpay.Client(auth=(settings.RAZORPAY_KEY_ID,
settings.RAZORPAY_KEY_SECRET))

def home(request):
    products = Product.objects.all()
    return render(request, 'home.html', {'products': products})

def product_list(request, category):
    products = Product.objects.filter(category=category)
    return render(request, 'product_list.html', {'products': products,
'category': category})

def product_detail(request, pk):
    product = get_object_or_404(Product, pk=pk)
    return render(request, 'product_detail.html', {'product': product})

def checkout(request, pk):
    product = get_object_or_404(Product, pk=pk)
    payment = client.order.create({'amount': int(product.price*100), 'currency':
'INR', 'payment_capture': '1'})
    return render(request, 'checkout.html', {'product': product, 'payment':
payment})

def checkout_success(request):
    return render(request, 'checkout_success.html')
```

---

## 8. Razorpay Integration

```
# settings.py
RAZORPAY_KEY_ID = 'your_key_id'
RAZORPAY_KEY_SECRET = 'your_secret'
```

- Payment flow triggers **PDF download** after success.

---

## 9. Templates Structure

```
templates/
├── base.html
├── home.html
├── product_list.html
├── product_detail.html
├── checkout.html
└── checkout_success.html
```

- **HomePage:** Hero + Featured Products + CTA
- **Product List:** Filter by category
- **Product Detail:** Buy button → Razorpay checkout
- **Checkout Success:** PDF download link

---

## 10. Static & Media Setup

```python
# settings.py
MEDIA_URL = '/media/'
MEDIA_ROOT = BASE_DIR / 'media'
STATIC_URL = '/static/'
STATIC_ROOT = BASE_DIR / 'staticfiles'
```

```python
# devopsvaultx/urls.py
from django.conf import settings
from django.conf.urls.static import static

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

- Upload PDFs → `/media/pdfs/filename.pdf`

---

## 11. Database Usage

- Products, Orders, Users, PDFs stored in PostgreSQL
- Django ORM handles queries automatically
- Example: `Product.objects.filter(category='notes')`
- Track orders: `Order.objects.create(...)`

---

## 12. Launch Checklist

✅DB migrated ( `python manage.py migrate` )
✅Superuser created ( `python manage.py createsuperuser` )
✅Products uploaded via admin
✅Razorpay sandbox test → live
✅Templates styled and tested
✅PDF delivery verified

---

## 13. Future Enhancements

- Subscription / membership model
- Analytics: downloads, top-selling products
- Blog / free resources
- Email notifications for buyers
- More product categories

---

**Congratulations!** ‼️ Your DevOpsVaultX platform is ready to sell PDFs online with full Django + PostgreSQL + Razorpay integration.