Alex Novak

Email: alex.novak@example.com **Phone:** +48 600 100 200

Position: Python / Django Developer **Address:** ul. Prosta 12/4, Warsaw

PROJECTS

Rolefolio — portfolio platform

↑ Cut the time to publish a polished portfolio from days to under an hour.

A Django platform for building shareable professional portfolios with public profiles, CV export and an AI assistant.

Problem:

Developers had no single place to present projects as case studies and keep a CV in sync with a public profile.

Solution:

Built a Django + HTMX app with a dashboard, public profiles, PDF/TXT export and a per-profile AI chat backed by LangChain.

Published: Nov. 10, 2025

Python Django HTMX AI

pytest integration test suite

↑ Stopped a class of bugs that had previously only surfaced in staging.

A full integration test suite for a Django API using pytest, factory_boy fixtures and a real PostgreSQL test database.

Problem:

The service had only unit tests; integration bugs repeatedly reached staging because mocked tests did not catch contract violations.

Solution:

Wrote integration tests using pytest-django and factory_boy, spun up a dedicated test DB in CI, and blocked merges below 80% coverage.

Published: May 20, 2025

Python pytest Django

Inventory API service

↑ Eliminated oversells and gave the ops team a queryable history of every change.

A REST API for multi-warehouse stock management with role-based access and audit logging.

Problem:

A retailer tracked stock in spreadsheets, causing oversells and no audit trail.

Solution:

Designed a Django REST Framework API with PostgreSQL, optimistic locking and a full audit log, containerised with Docker.

Published: April 22, 2025

Python Django postgresql docker

HTMX live search and filter

↑ Filter response time dropped from ~800 ms (full reload) to ~90 ms (partial).

A server-rendered product catalogue with instant search, multi-tag filtering and infinite scroll — no JavaScript framework required.

Problem:

The existing catalogue required a full page reload for every filter change, making it slow and frustrating on mobile.

Solution:

Replaced the form-submit loop with HTMX partial requests, Tailwind CSS for the UI and a Django view that returns HTML fragments.

Published: Feb. 6, 2025

Python Django HTMX Tailwind CSS

CI test-coverage dashboard

↑ Made coverage regressions visible in code review and raised average coverage.

A small Django app that ingests pytest coverage reports and trends them per service over time.

Problem:

Teams could not see whether test coverage was improving or regressing.

Solution:

Built an ingestion endpoint and HTMX dashboard charting coverage trends.

Published: Sept. 5, 2024

Python Django HTMX

Async email notification pipeline

↑ Eliminated request timeouts and provided visibility into delivery failures.

A Celery + Redis worker pipeline that sends transactional emails asynchronously, with retry logic and dead-letter tracking.

Problem:

Sending emails synchronously during request-response cycles caused timeouts and blocked the web server under load.

Solution:

Extracted email dispatch into Celery tasks backed by Redis, added exponential back-off retries and a Django admin view for failed tasks.

Published: July 18, 2024

Python Django Celery Redis

DRF customer self-service portal

↑ Cut support tickets by 60% within two months of launch.

A Django REST Framework API powering a self-service portal where B2B customers manage their accounts, invoices and API keys.

Problem:

Customer requests were handled manually by the support team, creating bottlenecks and delays.

Solution:

Built versioned REST endpoints with DRF, JWT authentication and a PostgreSQL-backed audit log for every sensitive action.

Published: May 3, 2024

Python Django REST API postgresql

Custom Django admin panel

↑ Reduced ops onboarding time and eliminated support tickets about wrong buttons.

A heavily customised Django admin for an internal operations team, adding bulk actions, inline editing and a role-based permission layer.

Problem:

The out-of-the-box admin was too generic and exposed fields the ops team did not need, while hiding the ones they used every day.

Solution:

Extended ModelAdmin, added custom views and mixins, and locked down permissions using Django's built-in group system.

Published: March 12, 2024

Python Django

Stripe subscription billing integration

↑ Fully automated the billing cycle and reduced failed-payment churn.

End-to-end Stripe Billing integration for a SaaS product, handling subscriptions, invoices, proration and webhook reconciliation.

Problem:

Billing was tracked in a spreadsheet; failed payments went unnoticed and plan upgrades required manual intervention.

Solution:

Integrated Stripe Billing via webhooks, synced subscription state to PostgreSQL with idempotency keys, and built a Django admin overlay.

Published: Jan. 28, 2024

Python Django Stripe REST API

AWS deployment pipeline

↑ Cut deploy time from 45 minutes to under 5, with zero-downtime releases.

A Docker-based CI/CD pipeline deploying a Django application to AWS ECS with automated rollbacks and environment promotion.

Problem:

Deployments were manual, error-prone and required SSH access to production.

Solution:

Containerised the app with Docker, set up GitHub Actions to build and push images to ECR, and configured ECS rolling deployments.

Published: Nov. 14, 2023

Python docker aws

Webhook event dispatcher

↑ Decoupled three downstream services and removed all polling queries.

An internal Django service that fans out domain events to subscribed external endpoints with delivery guarantees and retry logic.

Problem:

Services that needed to react to business events had to poll the main database, creating coupling and load.

Solution:

Built a webhook registry in PostgreSQL, a dispatch worker with exponential retries, and a subscriber management API.

Published: Aug. 22, 2023

Python Django Webhooks postgresql

CSV bulk data import CLI

↑ Migrated 400 k records in under 10 minutes with zero data loss.

A Click-based CLI tool that validates, transforms and bulk-inserts large CSV exports into a PostgreSQL database.

Problem:

A client migrating from a legacy system needed to import hundreds of thousands of records without a graphical tool or manual entry.

Solution:

Built a Click CLI with schema validation, progress reporting, transactional inserts with rollback on error, and dry-run mode.

Published: May 9, 2023

Python CLI postgresql

SKILLS

Python (Expert, 5 yr)

Django (Expert, 4 yr)

HTMX (Advanced, 2 yr)

PostgreSQL (Advanced, 4 yr)

Docker (Advanced, 3 yr)

Git (Expert, 5 yr)

Problem solving (Advanced, 5 yr)

English (Advanced, 10 yr)

WORK EXPERIENCE

Backend Developer

Northpoint Software

03.2023 — present • Warsaw

Building and maintaining Django services for a B2B SaaS product.

- Design and implement REST APIs in Django
- Own database schema and migrations
- Review pull requests and mentor juniors
- Reduced API p95 latency by 40% through query optimisation
- Introduced a CI test gate that cut regressions

Junior Python Developer

Bright Agency

06.2021 — 02.2023 • Remote

Delivered Django websites and integrations for agency clients.

- Implement features against client briefs
- Integrate third-party APIs and payment providers
- Shipped 8 client projects on time
- Built a reusable Django starter that sped up new projects

EDUCATION

BSc Computer Science — Computer Science

Warsaw University of Technology

2018 — 2021 • Warsaw

Focus on software engineering, algorithms and databases.

Thesis on web application performance; graduated with distinction.