Vadim Humeniuk

Software Engineer

As a highly efficient and fast-learning Software Engineer, I specialize in developing scalable, high-performance applications with a strong focus on precision, reliability, and maintainability. I have extensive experience in backend development using Go and Python, building robust APIs with FastAPI, Django, and GraphQL. My expertise includes working with databases such as PostgreSQL, DynamoDB, and Amazon RDS, leveraging SQLAlchemy for efficient data management.

I have a deep understanding of cloud-based architectures and have worked extensively with Amazon Web Services (AWS), including AWS Lambda, S3, DynamoDB, EC2, SQS, IoT Core, Cognito, API Gateways, and IAM. I have designed and implemented serverless and event-driven systems using AWS CloudFormation, Terraform, and the AWS CLI, ensuring seamless CI/CD workflows and infrastructure automation.

Contact

Phone
+380681163022

Email
v.humeniuk96@gmail.com

Address
Kyiv

Education

2013
Khmelnytsky politechnic college

2011
Yarmolintsi technological lyceum

Skills

- Programming languages
  - Python (3 years)
  - Go Lang (6 monts)
  - React.JS
  - SQL (PostgreSQL dialect mainly)
- Cloud
  - AWS (Lambda, DynamoDB, SQS, IoT core, IAM, and other services)
- Software & technologies
  - Web: Django, FastAPI, Flask
  - gRPC, protobuf
  - react, tailwind
  - ORM: SQLAlchemy, Django
  - pytest, selenium
  - OpenCV, PyTesseract
  - Docker
- Operating systems
  - Linux (ubuntu ,raspberry pi OS)
  - Windows (WSL)
  - macOS

Experience

2024 - 2025

Sygnal.io — Miami, Florida (Remote)

Software engineer (Backend + Frontend)

Responsibilities:

- Development of backend services with Go gRPC.
- Implementation of GraphQL resolvers and business logic
- Processing and exporting MongoDB event data for analytics with Python.
- Writing and optimizing test frameworks (Selenium, unit tests).
- Performance optimization and error handling in production services.

Key Achievements:

- gRPC & Go development:
  - Solved type conversion issues between gRPC client models and Go struct models, ensuring a smooth service interaction.
- GraphQL & Go backend services:
  - Implemented GraphQL resolvers for user invitation and authentication processes
  - Ensured robust error handling and optional password logic, addressing nil pointer errors in user authentication flows.
  - Debugged and optimized GraphQL mutations, ensuring correct integration with database logic.
- MongoDB data processing & Python:
  - Developed event processing for purchase events, extracting and filtering key data.
  - Mapped and transformed data fields (e.g., renaming order-related fields) for export in CSV format.
- Frontend (React)
  - Worked on React components to integrate backend APIs into frontend dashboards.
  - Implemented data visualization for analytics dashboards using React.

2022 - 2024

Dinamicka Development — Ukraine, Kiev

Software engineer (Backend)

Responsibilities: building backend applications, code reviews, mentoring

Key achievements:

- Key backend developer in building and delivering REST-based applications that power data-centric SaaS dashboard applications, had both of main and assisting developer in delivering projects in:
  - IoT (data ingestion, building software packages for Python-based controllers)
  - EdTech startups
  - Real-estate
- Was a key team member who established strict coding standards and architectural approaches on building reliable, scalable, modular backend applications through code review, consulting, and mentoring other team members on the job.
- Helped the team with transitioning to a cloud-first development approach through building fully-serverless (AWS Lambda/DynamoDB) proof-of-concept (and prototype) applications in IoT/SaaS dashboards area, which enabled in developing of cost-efficient solutions for early-stage startups.

- Databased I worked with
  - DynamoDB
  - PostgreSQL
  - MySQL
  - MongoDB

- Team collaboration experience
  - Agile frameworks
    - Scrum
    - Kanban
  - Experience working with JIRA

2021 - 2022

ARK CONNECT & MARKET GROUP — Ukraine, Kiev

Software developer

Development of software-automation solution of Android applications, mainly in datascraping for data analysis team. Utilised computer vision algorithms and various automation-centric approaches for getting an efficient architecture for the business needs

Key achievements:

- Was able to deliver project in time for a business critical need.
- The built solution was able to massively expand the profits for the company that application was written
- Was in charge of developing both the code and infrastructure of the project

Commercial projects worked on & technical experience

Real-estate startup

CRM dashboard and real-estate listings website

Built a OpenAI integration with Large language model for creating agent-based chat-bot inside of Django app. The initial AI application was a Slack-based bot that used government and open real-estate databases for giving summarised information on the current real-estate market. The prototype was later expanded and delivered in the MVP.

Technical stack:

- Django + Django Rest Framework (PostgreSQL). Created a webhook endpoint in a django app to handle Slack/OpenAI integration

Document management web application

Developed a SaaS application backend for generating documents by using jinja2 templates and serving FastAPI REST service for React.JS web user interface. Used specfirst (OpenAPI) code-generation approaches, was tasked to develop Stripe payment integration and build a background worker solution for asynchronous processing for document generation requests. The project was developed in a cross-functional team, where I was a key backend developer who was responsible for delivering the final solution to the client and iterating on it post-MVP release.

Technical Stack:

- PostgreSQL (SQLAlchemy)
- FastAPI
- React.JS

IoT project (REST-based API for dashboard and data-ingestion pipeline)

Developed a scalable architecture for an IoT solar panel ingestion pipeline. The system used **AWS SQS** for batch processing of incoming data and then storing that data and aggregates in **DynamoDB** tables. The app used **AWS Lambda** with python deployment for doing all of the application logic. Additionally, I had to solely build a prototype CI/CD pipeline and Infrastructure as code project that automatically deployed the server-less stack. At different stages of development the CI/CD part used **AWS CloudFormation** and **Terraform**.