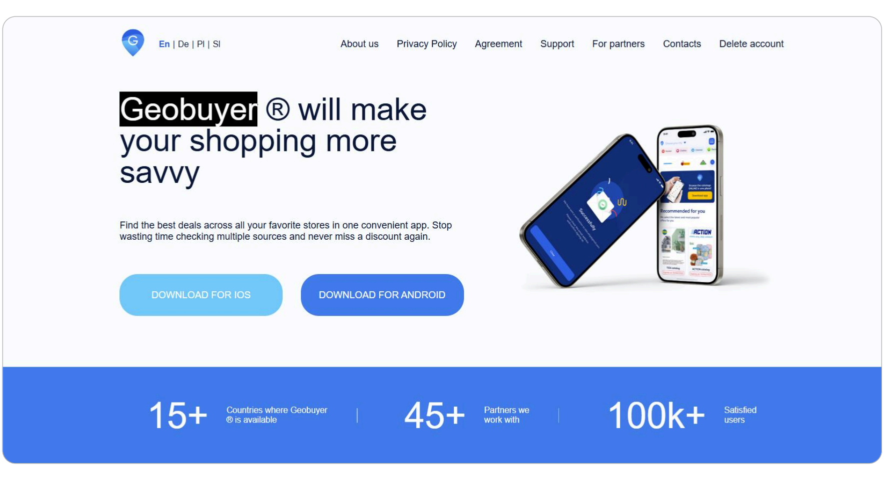
Geobuyer

MVP with Room to Grow: Geobuyer's Journey

Geobuyer is a smart shopping companion that brings together promotions from various stores in a single, easy-to-use app. It helps users discover discounts across all their favorite stores in one place, while giving retailers an opportunity to showcase their best offers.

The Geobuyer product can be compared to the good old paper promotional booklets. The difference is that they're all gathered in one place for a specific location and are always at your fingertips, right on your mobile phone.



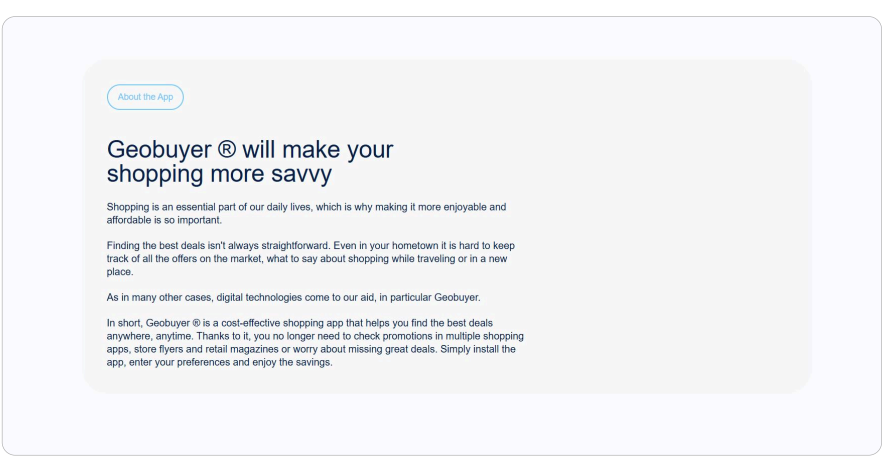
Project Description

Geobuyer is currently at the MVP stage. Initially, our client had an infrastructure that was sufficient for product development. But when it came to the product launch preparations, Geobuyer faced certain limitations, even despite having Kubernetes in place.

Our client's current goal is to attract initial traffic and test the infrastructure capacity in real-world conditions. Their long-term goal is to progressively expand into European markets and beyond.

IT Outposts' main task was to ensure that Geobuyer has a reliable infrastructure that will help them launch successfully. We're also keeping their future plans in mind and designing solutions that will allow our client's setup to scale smoothly as their business grows.

Since we're rebuilding the infrastructure from scratch, we've utilized our AMICSS solution. This is a prebuilt setup that comes with all essential components, such as multiple environments, VPN networking, security measures, monitoring and alerting, including 20+ expertly configured tools. This allowed our client to save time and budget on engineering hours and move on to planning their next goals sooner, in just one week after the start of our cooperation.



Provided Services

Extra services

- Containerization
- CI/CD configuration
- Cloud migration

AMICSS Infrastructure Establish package

Work agenda

Client
Geobuyer, a location-based deals aggregator

Location
Cyprus

Technical team
Solution architect
DevOps engineers
Project manager

Project timeframe
October 2024 - ongoing

Project goals

- Free Geobuyer from vendor dependency by migrating to DigitalOcean
- Create architecture supporting multi-country deployment across Europe
- Implement a proper CI/CD pipeline
- Replace outdated monitoring with modern, container-friendly solutions
- Develop an expansion roadmap with cost projections for different migration scenarios

Tasks and Challenges

01 Vendor lock-in risk

The way the client had set up their infrastructure initially was quite risky because they were heavily dependent on their previous DevOps provider. Everything was hosted on a proprietary cloud platform owned by the vendor, which meant Geobuyer didn't have control over their own infrastructure. They didn't have access to the scripts, managing permissions, or configuration settings.

02 Inadequate CI/CD pipeline

Their existing setup lacked the automation and reliability needed for smooth development workflows. The pipeline was more of a manual, error-prone process that created bottlenecks for their development team.

03 Old-fashioned monitoring system

The client relied on Zabbix, an older monitoring tool that wasn't built for today's modern, container-rich environments. As a result, the team ended up spending too much time fixing and maintaining the monitoring system itself, rather than focusing on improving their application using the insights it provided.

04 Multi-country deployment complexity

Another challenge our client faced was how to efficiently deploy their app across multiple European countries, each with its own data requirements. Their plans involve launching in Poland, Slovenia, Germany, Austria, and Ukraine, which is just the start of their international expansion. Yet, each country needs its own isolated environment with dedicated resources, but all of these have to work together smoothly.

05 Future scalability requirements

The client's goals go far beyond just launching in Europe. They have a big, worldwide vision for Geobuyer — to eventually operate in countries around the globe. This requires careful planning and thinking several steps ahead so as not to redo everything later.

Solutions

1. DigitalOcean cloud migration

We fully migrated the client's infrastructure from their old vendor's proprietary cloud to DigitalOcean. DigitalOcean is a great choice for their MVP phase. It has the right balance between performance and affordability, with clear, straightforward pricing that works well within a startup's budget.

2. CI/CD pipeline introduction

Our engineers set up a proper CI/CD pipeline from scratch. Now, whenever developers push new code, the system automatically initiates a build process.

3. Comprehensive monitoring setup

We replaced the client's outdated Zabbix monitoring system with a modern, comprehensive solution, part of our AMICSS package. At the core is Prometheus, a strong open-source tool that's perfect for cloud-based environments. We also customized Grafana dashboards to display everything from resource use to application metrics, making it simple to spot trends or problems at a quick glance.

4. Namespace architecture

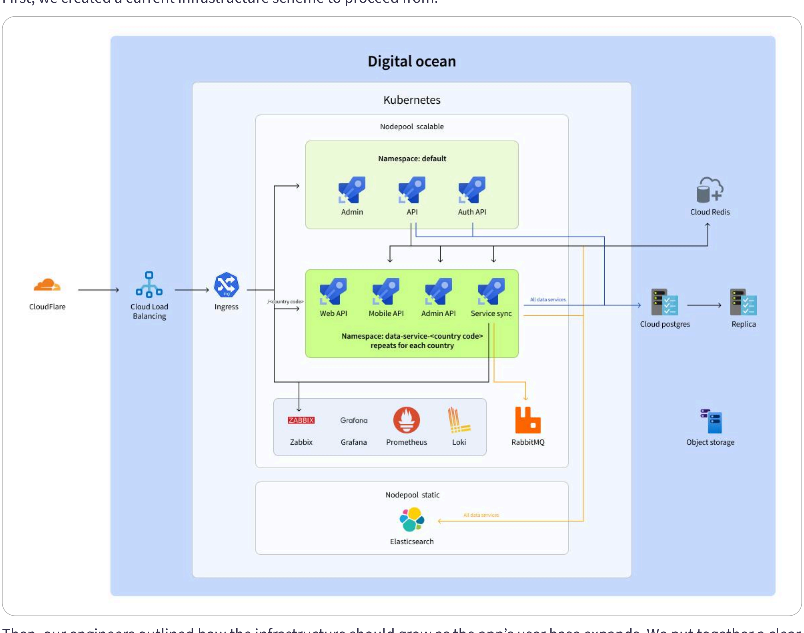
In the Kubernetes cluster, we set up separate namespaces for each country — Poland has its namespace, Germany has its own, Slovenia has another one. Each country namespace contains all the services needed for that specific region, but they can still use shared resources like the core services that are common across all countries.

For monitoring, alerting, and logging, our team connected each namespace to central systems, and this gives us visibility across all countries. In the future, when Geobuyer expands to more distant regions like Asia or America, we'll set up new clusters in those geographic areas to keep performance strong for local users.

5. Expansion plan creation

While our main focus was on launching the app and getting it ready for initial users, we also thought ahead about the future. The current DigitalOcean setup works well for now, but there are some limitations when it comes to global expansion, especially since DigitalOcean doesn't have data centers in regions like Africa or Latin America.

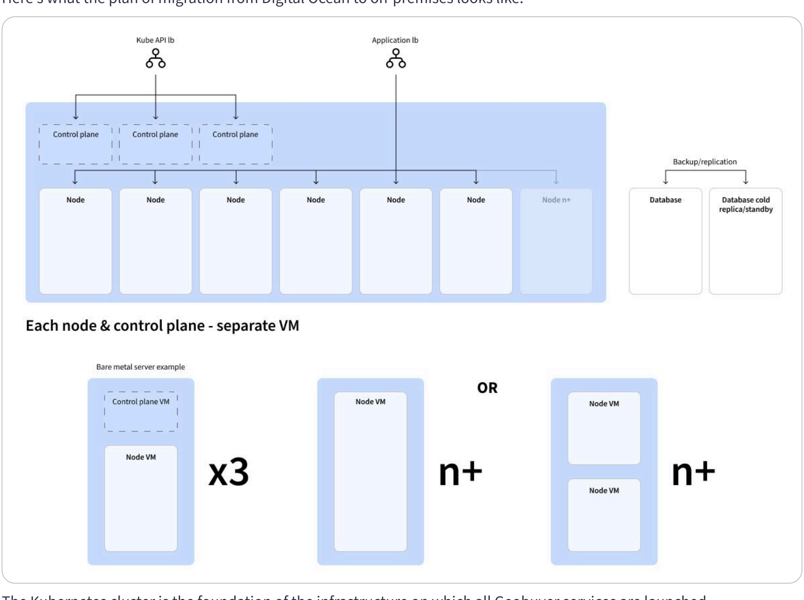
First, we created a current infrastructure scheme to proceed from:



Then, our engineers outlined how the infrastructure should grow as the app's user base expands. We put together a clear step-by-step plan showing what changes are needed at 5,000, 30,000, and 100,000 users. For each stage, we also estimated the costs involved.

Looking even further, our team developed a plan for eventually moving from cloud hosting to physical servers. We mapped out the equipment they'd need, how long the transition would take, and what it would cost.

Here's what the plan of migration from Digital Ocean to on-premises looks like:



The Kubernetes cluster is the foundation of the infrastructure on which all Geobuyer services are launched.

It would consist of:

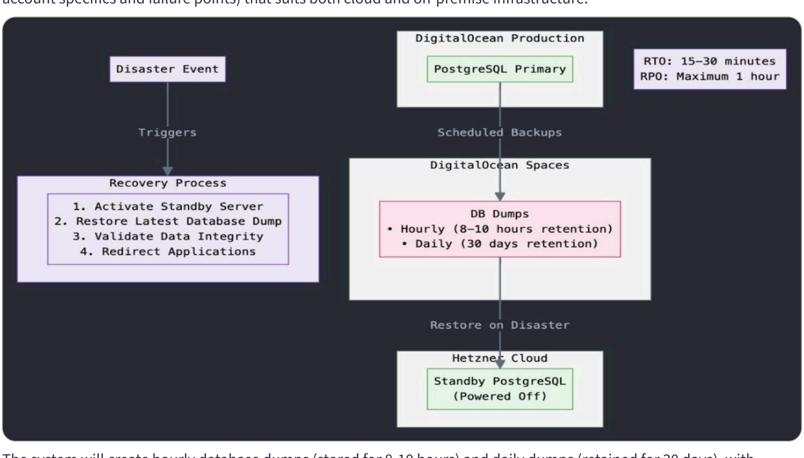
- Control plane (control nodes) that is responsible for coordinating the work of the entire cluster
- Worker nodes (worker nodes) that launch applications and microservices

For reliability, three control plane nodes must always be used, hosted on separate virtual machines. Each component (control plane and node) would be hosted as a separate virtual machine, even if they run on the same physical server.

For fault tolerance, we'd double the hardware - what runs on one server in the cloud is deployed on two on-premise, so that the system can continue to operate if any element fails.

The database (for example, PostgreSQL) would have a separate cold replica or backup copy, which would be activated when the main database crashes.

In addition, our team developed a basic disaster recovery plan (basic as it requires detailed documentation, taking into account specifics and failure points) that suits both cloud and on-premise infrastructure:



The system will create hourly database dumps (stored for 8-10 hours) and daily dumps (retained for 30 days), with snapshots stored in object storage across multiple providers (for instance, DigitalOcean Spaces and Hetzner) for redundancy.

A standby PostgreSQL server will be deployed on a cost-effective provider like Hetzner, remaining powered off until needed. During failures, this server will be activated and restored from the latest dump.

The team will receive comprehensive documentation and training, with quarterly recovery drills and regular backup integrity checks to ensure system readiness.

Expected outcomes are that recovery time will be 15-30 minutes (depending on database size), with a maximum data loss of one hour if the latest backup remains intact. This approach will significantly reduce costs compared to continuous replication while maintaining reliable recovery capabilities.

Now, our client has a clear strategy to follow if they find that cloud expenses become too high as they scale up.

Results

- We helped the client take full control over their infrastructure by moving from their previous provider's proprietary cloud.
- The Geobuyer app is now fully able to attract real users and gather valuable feedback to test and improve their business model.
- By using our ready-made AMICSS solution, we reduced the core infrastructure setup time from several months down to just one week.
- Clear cost projections for different user milestones and migration scenarios make it easier for our client to plan finances and engage with investors.

DevOps tech stack

Kubernetes, Loki, PostgreSQL, Redis, Elasticsearch, ArgoCD, Reflector, Prometheus, Zabbix