voucherify

Resiliency & Security

In this document, we showcase the robustness of Voucherify in terms of **reliability** and **resilience**. Voucherify consistently supports F500 enterprises during high-demand events, such as Black Fridays, Cyber Mondays, and the holiday season, with no service disruptions.

Voucherify is a **cloud-native SaaS product**, designed to serve multiple clients concurrently. The cloud-native approach is at the core of our development philosophy. Voucherify gives you access to a collection of **promotion & loyalty REST APIs** and a **visual Dashboard**. For over a decade now, Voucherify has been creating a scalable, dependable, and secure product that is highly resilient against security risks and downtime.

Within this document, we delve into the fundamental architecture of Voucherify, highlighting key aspects of our microservices approach, detailing our **backup** and **recovery strategies**, and presenting our **exceptional customer support services**. At Voucherify, we have unwavering confidence in our product, and our goal is to instil the same level of confidence in you.

Table of Contents:

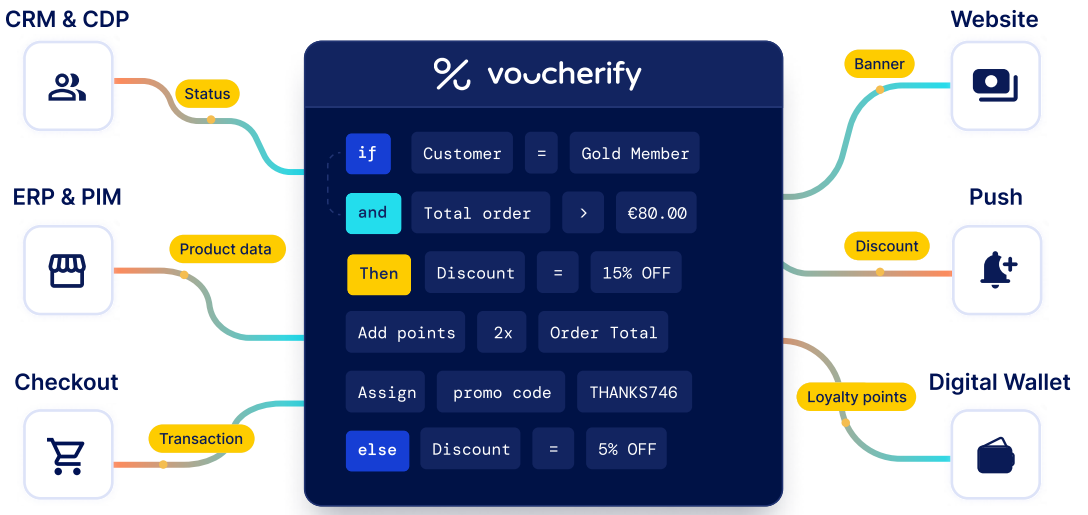
What is Voucherify?

Voucherify is an **API-first Promotion & Loyalty Engine** that helps companies launch and manage personalized **coupons**, **gift cards**, **auto-applied promotions**, **loyalty programs**, and **referral campaigns**.

Voucherify gives you access to a collection of **promotion & loyalty REST API endpoints** and **a visual Dashboard**.

Every integration is unique – however, at its core, the integration of Voucherify with your system relies on **passing relevant context to Voucherify API**, which validates and redeems incentives and rewards in accordance with the rules you set up with our comprehensive Rules Engine.

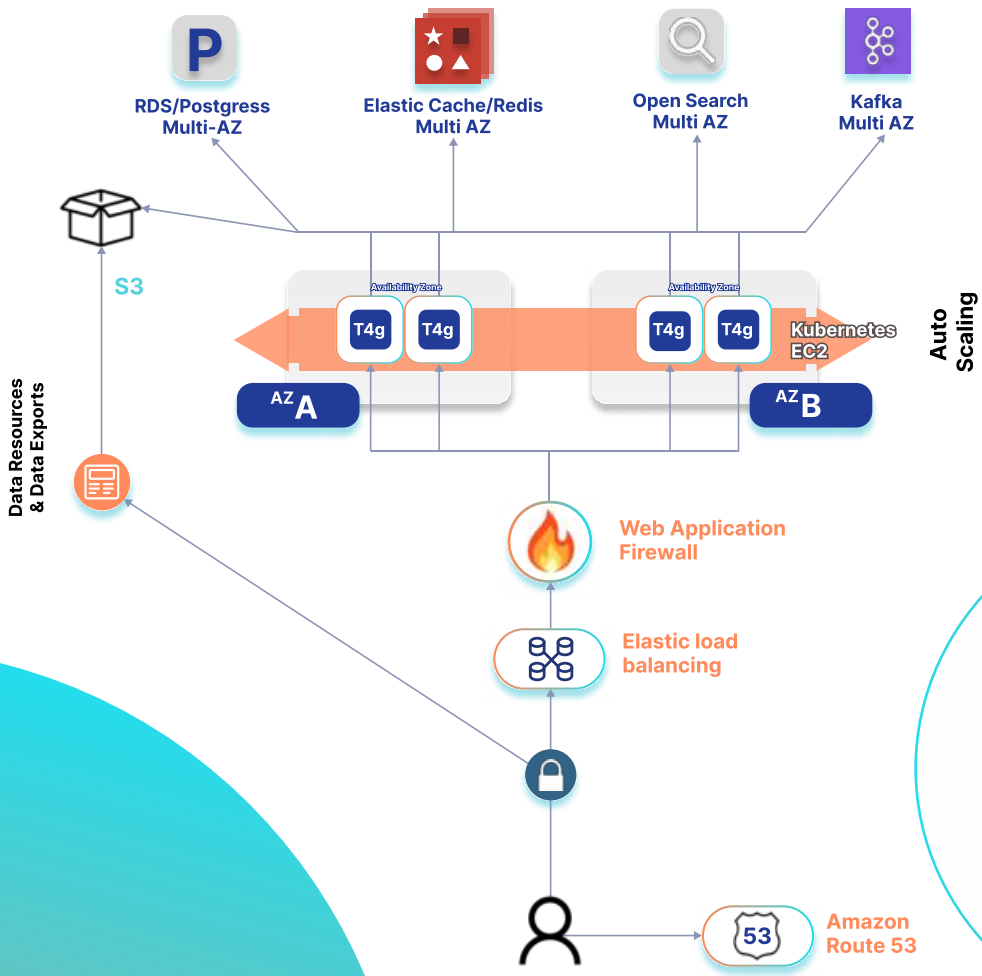
As a **MACH-certified vendor**, Voucherify follows the principles of composability, giving you flexibility and decreasing the time needed for integration.



Basic Architecture

This section provides an in-depth look at Voucherify's **multi-tenant**, **cloud-native**, and **microservices-based architecture**. Our architectural approach aligns with the latest industry standards, enabling us to maintain agility in our design while bolstering our resilience to potential failures.

The diagram below illustrates the sophistication of Voucherify's microservices-based, cloud-native architecture.

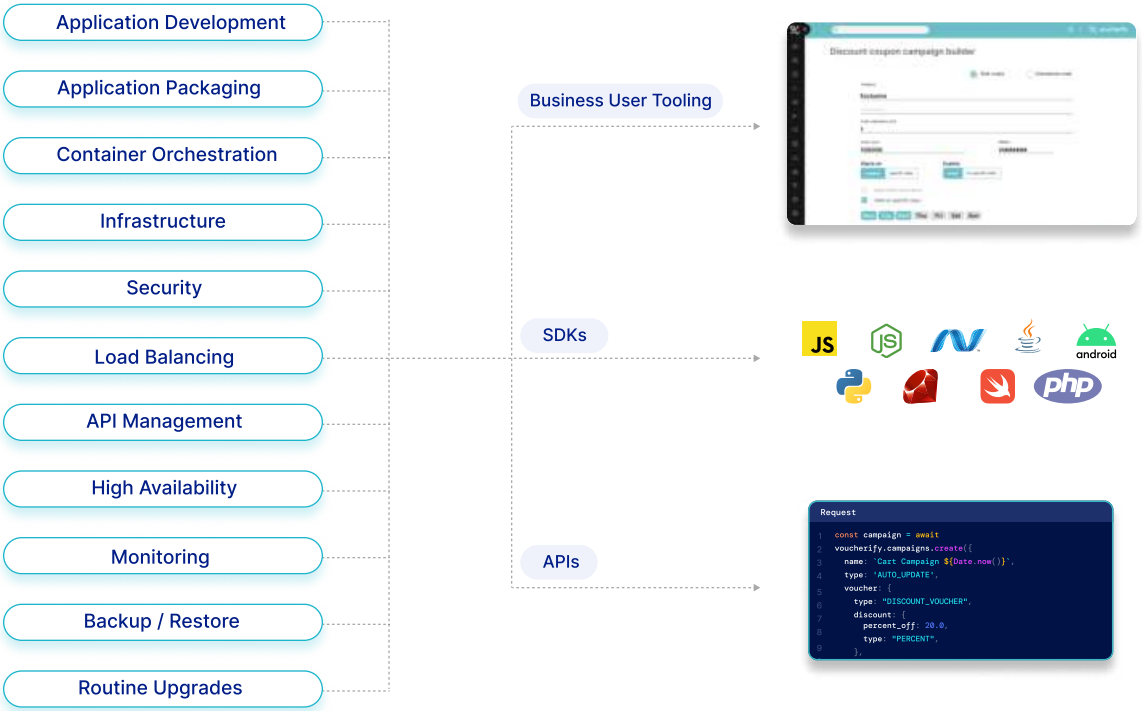


Cloud Services

Voucherify is using **Amazon Web Services** (AWS) as a 3rd-party vendor, which runs in a multi-tenant, geographically distributed environment to support the availability of services through the use of redundant architecture. Voucherify is an official AWS partner.

As Voucherify utilizes an Infrastructure as Code (IaC), the whole infrastructure can be rebuilt and recovered to the previous working state. Also, any change can be reverted back if necessary.

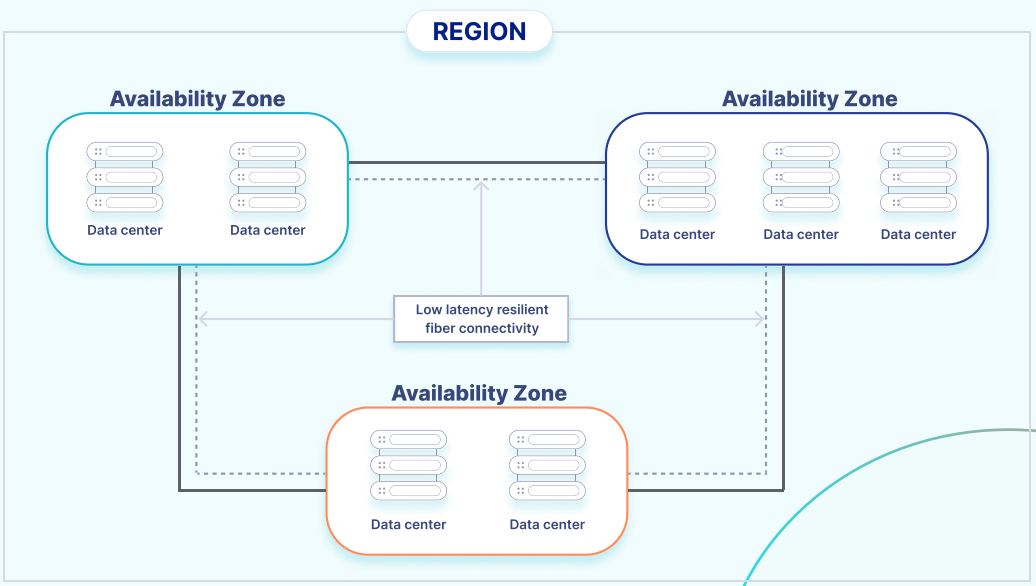
The illustration below offers an overview of how Voucherify's architecture integrates and collaborates within the cloud environment.



Regions and Availability Zones

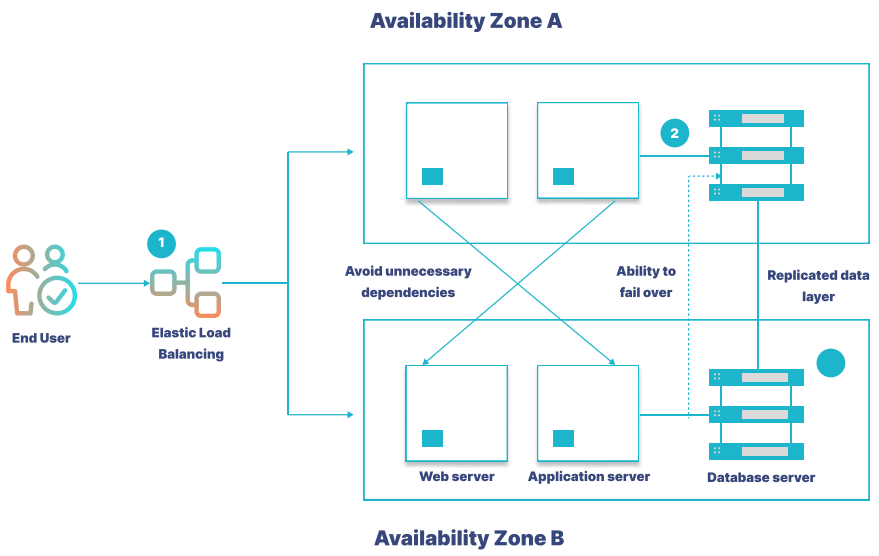
Cloud architecture empowers the distribution of hardware infrastructure worldwide through **regions**. In our software stack, most of our services capitalize on **a minimum of three Availability Zones within each Region**. This cloud deployment strategy inherently safeguards against application node failures. Consequently, hardware failures become a non-issue as they are automatically managed, and cloud providers maintain redundancies.

Regions represent distinct geographic areas comprising multiple Availability Zones. Each Availability Zone serves as a deployment zone for cloud resources and is treated as a discrete failure domain. These zones are physically separated with independent network connections and power supplies. However, they are strategically positioned within a region to ensure network latencies of single digit milliseconds for a round trip. Availability Zones enable cloud deployments to achieve full active/active availability within a region.



Simply being in the cloud does not automatically utilize the benefits of Availability Zones. Customers retain the flexibility to deploy on-premise applications within a single cloud zone. However, when applications are not specifically designed to be cloud-native, customers may find themselves constrained by this deployment model. Opting for such an approach might mean missing out on some of the inherent advantages that cloud computing offers.

In every designated region, a minimum of **triple redundancy** is maintained, guaranteeing uninterrupted service for each individual component (such as applications, databases, etc.) irrespective of hardware failures, system glitches, or zone disruptions.

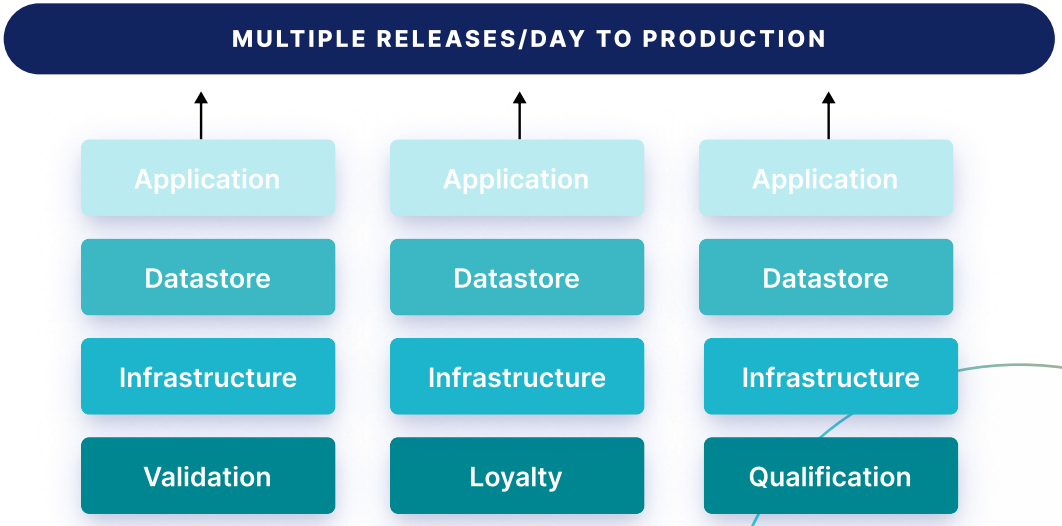


Independent APIs Backed by Microservices

Our product features are delivered through **autonomous REST APIs** supported by a number of microservices. Those are managed by a **Kubernetes cluster**, and each REST API microservice has a minimum of **two replicas** distributed across at least two Availability Zones. Thanks to a Rolling-Update policy, there is zero downtime for deployments, as there is always a minimum number of applications that can handle incoming traffic.

We also host microservices which act as **Kafka queues consumers** and **producers**. Those applications take care of asynchronous processing, events, side effects, and more.

This architectural approach empowers specialized teams to implement updates through separate development and release cycles, ensuring agility. We have incorporated **Continuous Integration** and **Continuous Deployment (CI/CD)** pipelines for our services. This methodology harnesses automation and lifecycle monitoring to streamline the process of introducing code changes into production, facilitating secure deployments to production environments within minutes.



Microservices vs. Monoliths

Voucherify follows the **microservices approach**. Contrary to the monolithic structures where all processes are interconnected, microservices are designed as standalone units.

Each unit, like Qualifications API, **operates independently**. They interact subtly – changes in one of them are tracked by another. However, a failure in one does not cripple the other. For instance, if the Qualification API encounters issues, it affects only its direct users, while the Vouchers API, continues its operations undisturbed.

This modularity contrasts with monoliths where **a failure in one piece can lead to the system collapse**. In Voucherify, compartmentalization not only enhances resilience but also streamlines development.

While microservices largely prevent cascading failures, certain critical services, like Authentication service, play pivotal roles. A malfunction here would not halt other services but would impair their functionality, notably in API authentication. Recognizing such **Single Points of Failure** is crucial, underscoring Voucherify's resilience compared to monolithic counterparts. Aware of potential Single Points of Failure, we adopt multiple strategies to mitigate risks.

What is MACH Alliance?

The MACH Alliance is an industry body that advocates for open and best-of-breed technology ecosystems, empowering businesses to transition from legacy infrastructure to a composable approach. The MACH name comes from its goal to promote tools that are Microservices-based, API-first, Cloud-native and Headless. Voucherify was built on these principles from the get-go, and was the first promotion engine to join the Alliance in 2020.

Database Sharding

Database sharding empowers us to create a **horizontally scalable data infrastructure on-demand**, surpassing the limitations of conventional virtual machines. We have implemented data sharding in **Postgres** (partitioned tables), **ElasticSearch** (shards), and **Kafka** (topic-partitions).

What is database sharding?

In distributed systems, data is often partitioned, or 'sharded,' into keyed partitions that are distributed across a large number of virtual machine instances and disks.

Domain-Driven Design (DDD)

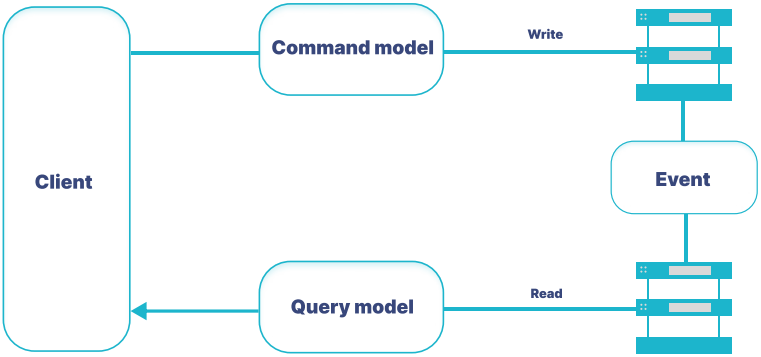
Domain-Driven Design (DDD) is an approach to software development that emphasizes modelling software systems based on the **real-world business domain** they serve. It encourages creating a shared, well-defined vocabulary and breaking down complex domains into manageable components called bounded contexts.

DDD distinguishes between entities and value objects, uses aggregates to group related objects, and employs domain events, services, and repositories to represent and interact with domain logic. DDD helps align software architecture with the intricacies of the business domain, **fostering collaboration between domain experts and developers** to build software that accurately reflects real-world complexities.

Command Query Responsibility Segregation (CQRS)

Voucherify uses the **Command Query Responsibility Segregation** design patterns, which ensures that data is not treated the same way for writing and reading operations. In this pattern, incoming update requests are processed and transformed into multiple events. A response, if needed, is given immediately, and other events are then delivered to independent microservices (like Kafka) for asynchronous handling. This approach minimizes the load and maximizes resiliency.

CQRS significantly elevates our Voucherify's availability by allowing each data store to scale independently. Datastores for handling write operations are configured differently from those for read operations.

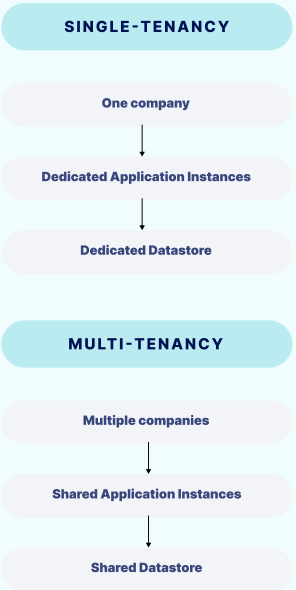


Our architecture uses two types of data consistency: **strong consistency** and **eventual consistency**. Strong consistency ensures that changes take effect immediately after the API returns a response. In contrast, eventual consistency, processes an API call that modifies an entity by queuing it for later dispatch as an event (this is performed leveraging external tools like Kafka).

By harnessing both strong and eventual consistency, Voucherify can handle a substantial volume of API requests efficiently, balancing the need for immediate updates with scalability and responsiveness.

Multi-tenancy

Voucherify is designed to be **multi-tenant**, allowing multiple customers to share infrastructure and application resources securely without accessing each other's data. To achieve this, we implement logical data isolation in the shared RDS instance, based on the tenant ID and project ID properties. Access to these separate projects is managed through API credentials.



- **Frequent product releases**
- **Highest possible security: one stack to manage, not one per tenant**
- **Access functionality any time**
- **Higher availability, globally distributed**
- **Better support due to one codebase for provider to support**
- **Low TCO, incremental licensing model, no fixed overhead**

Our multi-tenant system offers significantly higher limits than a single-tenant system. However, there are still limits, and is a tenant exceeds these limits, we automatically detect and mitigate the issue by restricting the resources allocated to that tenant (API Rate Limiting).

Due to our resilient and secure multi-tenant solution, **we do not perform one-off deployments**. This approach ensures a consistent experience for all customers in a region and allows immediate access to performance enhancements and fixes.

Self-service customers can choose the following multi-tenant clusters located in:

- Europe (Ireland)
- Asia (Singapore)
- US (East Coast)

For Enterprise customers, we offer dedicated clusters in the following locations:

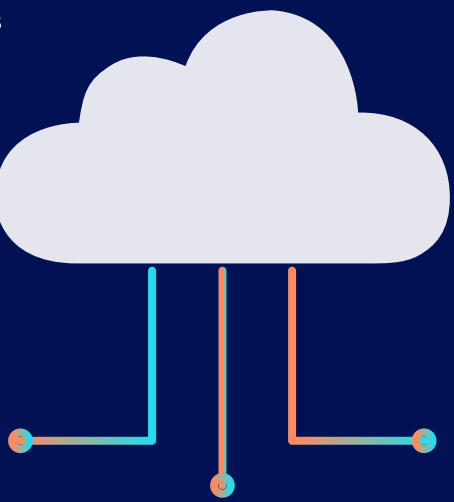
- Canada (Central)
- Canada West (Calgary)
- US East (Ohio)
- US East (North Virginia)
- US West (North California)
- US West (Oregon)
- South America (Sao Paulo)
- Europe (Frankfurt)
- Europe (Zurich)
- Europe (Stockholm)
- Europe (Milan)
- Europe (Spain)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- Asia Pacific (Hong Kong)
- Asia Pacific (Tokyo)
- Asia Pacific (Seoul)
- Asia Pacific (Osaka)
- Asia Pacific (Mumbai)
- Asia Pacific (Hyderabad)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Jakarta)
- Asia Pacific (Melbourne)
- Africa (Cape Town)
- Israel (Tel Aviv)
- Middle East (UAE)
- Middle East (Bahrain)

Cloud Security

Cloud service deployments offer **efficient environments** and **streamlined processes** for updating services with significantly fewer assets. Unlike on-premises setups, cloud-native solutions reduce the substantial risks associated with **securing**, **managing**, and **funding infrastructure**, facilities, and networks. Our cloud-native services enable swift deployment of crucial code updates to a minimal set of systems, substantially reducing the time required for time-sensitive changes.

Furthermore, our native cloud services significantly **trim the inventory of infrastructure software** to manage and protect, maintaining a modest system footprint well below that of on-premises environments. By leveraging **Kubernetes** in the cloud, we eliminate the need to secure servers as hosts. With our cloud-native services, customers are relieved from overseeing the vast scale and complexity of infrastructure, hardware, and the scope of vulnerabilities linked to on-premises services.

In essence, we alleviate the burdens and security risks inherent in enterprise operations associated with on-premises systems. Our customers **no longer need to shoulder the extensive controls, efforts, operations, monitoring, and management** required to secure on-premises products, all while upholding our unwavering standards of resiliency.



Handling Traffic Spikes

Voucherify consistently supports F500 enterprises during high-demand events, such as Black Fridays, Cyber Mondays, and the holiday season, with no service disruptions.

You can check out historical uptime via the **status page**.

We understand the challenges of traffic management in a SaaS environment. In order to decrease the negative impact of unexpected traffic spikes, we implement several safeguards at Voucherify:

- **Resource buffers** – Voucherify operates with substantial resource buffers, maintaining usage at around 50% capacity for handling sudden traffic increases. Together with Horizontal Pod Autoscaling in Kubernetes, this safety margin is our first line of defense, allowing us to quickly (30-45 seconds) scale up the number of application pods (replicas).
- **Auto-scaling** – made possible thanks to the AWS cloud hosting and our stateless apps' architecture – spinning up a new AWS node and making it available for the Kubernetes cluster takes between 1-2 minutes, booting up a new application takes 15-30 seconds, scaling up Postgres databases takes minutes to hours and is conducted with zero-downtime thanks to our multi-AZ setup with failover configuration.
- **API Rate Limiting** – limiting strategies (per-minute buckets, per project), which can help throttle uncontrolled bursts (e.g., misconfigured 3rd-party integrations), without affecting your standard legitimate traffic. On top of that, with the option to use separate API keys (e.g., one API key per integration) it is easier to quickly pinpoint and deactivate a faulty integration.
- **Caching mechanisms** – internal caching on multiple levels (including in-memory, Redis, Postgres).
- **Async processing** – offloading non-critical operations to our queuing systems for async processing. This technique does not only allow for faster API responses, but also allows for batch processing, which is more resource-efficient, and it offers retrying in case of failures.
- **Support** – engineering support during migrations and planned traffic peaks.

Software Bugs and Misconfigurations

Voucherify has adopted a multi-tiered methodology to guard against bugs and misconfigurations. These include **thorough code reviews**, **rigorous automated testing**, and **staged deployments via CI/CD methodologies**. Changes undergo detailed validation in staging environments before production rollout. We deploy changes incrementally, starting with one cluster, to minimize potential disruptions. In case of a detected bug in production, swift rollbacks (that take less than two minutes) are in place.

Infrastructure misconfiguration

Our infrastructure configurations are versioned and applied during a deployment. Should a misconfiguration be introduced, we can easily roll back to a previous version. For time-critical situations, we can apply manual changes. These changes are always done with at least two engineers managing the change.

Stateless Component Failure

Each service, particularly those handling API requests, is stateless and distributed across multi-zone Kubernetes clusters. It means that individual failures only affect ongoing requests. Kubernetes swiftly replaces failed instances, and our multi-zone setup guarantees continuity even in the event of a zone failure.

Database resilience

Our reliance on AWS Postgres RDS with multiple availability zones supported shields us from most database-related outages. Deployed across multiple availability zones, it offers automatic failover and recovery. In extreme scenarios requiring data rebuilds, we have comprehensive backup systems in place. In case of a catastrophic failure, we can easily create a new Postgres cluster with a backup from AWS RDS.

Utilizing Managed Services

We also leverage other cloud-managed services like Load Balancers for optimal performance and reliability, ensuring we always have the most effective solutions without overburdening our engineering team.

Data Backup

Voucherify has three primary components: **data**, **infrastructure**, and **application**. Our key principle is to engineer these components to be as **stateless** as possible. Data backup is central to our backup and recovery strategies.

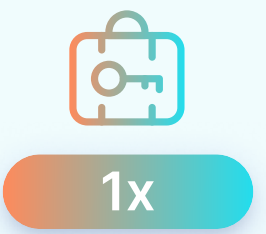
A mission-critical part of our data resides in **AWS RDS databases** which provide an automatic data back-up (full copy of the database and incremental snapshots) created every 24 hours and kept for 7 days. For additional safety, we use **AWS Backup** that copies the backups to a Backup Vault located on a separate AWS account, with very limited user access.

We also automatically back up other services: search engine, cache, and application logs. Search and cache can be recreated from scratch from RDS data, but to speed up the potential recovery, we perform backups of these services independently – daily, with 7 day retention.

Our backup routine follows the **3-2-1 backup principle**. This involves maintaining a minimum of three copies of our data, storing two copies on distinct storage media, and ensuring one copy is kept offsite.

Maintain at least 3 copies of your data

Keep 2 copies stored at separate locations



Store at least 1 copy at an off-site location

Data Recovery

Our backups occur automatically, but the data recovery process is manual, tailored to the unique nature of each incident to ensure optimal service restoration. We adhere strictly to our Service Level Agreements (SLAs) for Recovery Point Objective (RPO) and Recovery Time Objective (RTO) to guarantee the prompt reinstatement of functionality.

In-Region Recovery

In the unusual event of an issue progressing to production, our system is designed for rapid identification and rollback to a previously stable version, resolving the issue before it affects customers.

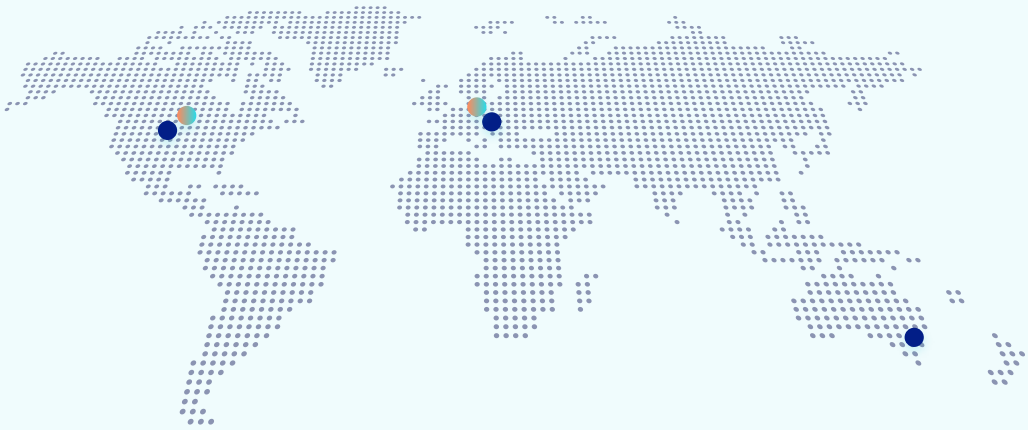
Should a rollback be impossible, we are prepared to quickly develop and deploy a corrective fix. The time to create a fix may vary, but its deployment is swift – typically within minutes.

Occasionally, a critical problem might necessitate re-provisioning of a specific service within the affected region and loading up the data from the back-up. The complexity of this task depends on the service and parallels the steps in multi-regional recovery, though focused on the impacted service.

Multi-Region Recovery

Voucherify is equipped to initiate a recovery in a different region if required. An illustrative example is the recovery procedure in separate cloud regions like us-west-2 (Oregon) or eu-central-1 (Frankfurt) or any AWS region that offers 3 Availability Zones (vast majority of them do).

It's important to note that regional outages impact all SaaS applications in that region, not just those deployed by a particular vendor. Therefore, restoring our services might not fully reinstate the functionality of your entire solution.



We utilize disk snapshots for rapid customer data recovery. These snapshots are transformed into new virtual disks in any region and zone within AWS and attached to a new Postgres RDS cluster. This method speeds up recovery by eliminating the need for data installation or synchronization.

Once such a new Postgres cluster is up, the applications' configuration is updated, and they are directed to switch to the new database host. Platform configuration data is stored in a Git repository, encrypted, also separately backed up every 24 hours.

Infrastructure Recovery Procedure

Our infrastructure details are managed via Terraform and tracked in Git repositories. Every service, every connection between them, every property and every bit of configuration is managed via Terraform modules. Terraform is used to set up, for example, all RDS databases, Kubernetes clusters, ElasticSearch services, Kafka queues, monitoring, logging and alerting systems and dashboards, DNS records, load balancers, WAF firewalls, and more, in all regions.

This allows us to keep all Voucherify clusters uniform in terms of configuration and keep track of changes, while at the same time parameterizing each cluster.

Application Recovery Steps

Post infrastructure setup and snapshot conversion to virtual disks, we begin installing Voucherify's microservices.

This process mimics our standard CI/CD deployment, involving:

- Building Docker containers.
- Uploading containers to a Docker registry.
- Deploying these containers to Kubernetes.
- Configuring Kubernetes.
- Our Docker registry, which backs up all containers to AWS ECR, is replicated across regions. In recovery scenarios, we bypass the initial two steps, speeding microservices deployment.

Search Engine Recovery Specifics

ElasticSearch is critical for powering key services, such as customer profiles, segmentation, and search.

Once the Voucherify apps are running, a set of predefined scripts is executed to ensure the search engine and caching systems are in sync with the Postgres database, our single source of truth. These scripts are optimized for batch operation, which guarantees a speedy execution. Based on the data volume, this phase takes 20-60 minutes.

Cutover

The new region will inherit all projects and configurations from the failed region. The final phase involves redirecting all DNS records to the new setup. This redirection occurs within the cloud system, without the need for DNS propagation, ensuring immediate traffic rerouting. All URLs, Keys, and Secrets remain unchanged. Client applications will require no modifications and should resume normal operation post-cutover. However, a reboot may be necessary for some client applications.

Performance Tests Results

Recently, we completed performance tests for our prospects that proved Voucherify can handle high traffic at a required response time speed, constantly. We proved Voucherify can:

Handle high traffic

2500+ transactions per minute with fast API response times (**<50ms**).

Support sustained load

4000+ transactions per minute with fast response times (**avg. 100ms**). Handles **12 000+ requests/minute** under sustained load.

Process loyalty points quickly

Average **1 second delay** despite asynchronous handling.

Support heavy promotion load

3300+ promo codes/cart promotion redemptions per minute in a POS with median response time of **100 ms per transaction**.

Scale in minutes

Voucherify can seamlessly transition from a low-traffic idle state to supporting a surge of **3300+ customers per minute within a mere 3 minutes.**

Update loyalty balance near real-time

5000+ customers making orders and checking their loyalty balance per minute with **1 second per transaction.**

**Test 1 included 1 million customers total (incl. 60k customers in a segment based on geo-location), a total of 100k voucher codes in a discount campaign (incl. 60k codes published to aforementioned customers) and 200 products with geo-location information stored in metadata.*

***Test 2 included 5,000 customers making purchases every minute and subsequently checking their updated loyalty points balance.*

Support

Voucherify's support is handled by experienced, full-time engineers dedicated to offering SLA-guided incident responses. This team includes Customer Success Specialists who oversee all of Voucherify's product-related inquiries.

By employing top-tier support tools in a cohesive framework, SRE and engineering teams ensure efficient management of automated alerts and staff scheduling. Our internal processes are consistently refined, drawing valuable insights from each incident to enhance our efficiency.

In the case of a service interruption, customers can monitor the status of our products via our **API status page**. This page also allows users to register for proactive notifications of incidents or degradations. New issues can be reported via **email**, phone or using the **Community Chat**, depending on the service plan.

Whenever Voucherify's automated monitoring systems or on-call rotations identify issues in a client's specific usage patterns, our support team takes proactive steps by contacting the account owner. It is highly advised that customers keep their account owner details current.

In line with our commitment to ongoing improvements, we regularly conduct post-mortem meetings to review incidents. These sessions focus on learning from process and technology, bolstering our overall resiliency.

Data Security

As an ISO-27001-certified company, Voucherify implements several security standards and practices to protect your and your customers' data:

- AWS cloud security (Virtual Private Cloud).
- Encryption (AWS KMS, TLS 1.2, data encryption at rest with AES-256).
- Regular PCI scans, security audits, and pen tests performed by a third party IT security company.
- Web Application Firewall with active blocking rules.
- DDOS protection (connection limiting, WAF).
- Login brute-force protection.
- Logging and monitoring systems, along with alerting and anomaly detection (Prometheus, Grafana, NewRelic, CloudWatch, PagerDuty).
- Role-based access and policy enforcement (AWS IAM, VPN, access logs, periodic permission reviews).
- All critical systems secured with multi-factor authentication and/or authenticating through SSO (enforced).
- GDPR & CCPA compliance.
- Disaster Recovery Plan and custom disaster recovery protocols defined in the SLA.
- Two-factor authentication, strong password policies, SAML.
- Reliability and backup (RAID class hardware, AWS S3).
- Redundancy of all underlying for High Availability.
- Automated daily data backups; additionally, snapshots copied over to a separate AWS account with limited access as an extra layer of security. We continuously check whether automated backups succeed and are available. Moreover, the procedure of recovering data from snapshots is tested regularly.

Contact our Customer Success team for a complete Security Architecture Model or the recent pentest report.

Reach your campaign goals with Voucherify experts

With support teams based across time zones and secure cloud servers on every continent, you can expect ongoing support and personalized onboarding for both your tech and non-tech teams.

We want you to succeed with Voucherify. To simplify the usage of our platform, we've created resources such as **Developer Documentation**, **Community Chat**, **User Guides**, and **on-demand webinars**. Our **GitHub repositories** also contain a lot of valuable samples and tools.

Technical Account Management team

sales@voucherify.io

voucherify