

Vultr Docs - Tech Stack, Architecture, and More

Explore Vultr's documentation to understand their tech stack, architecture, and infrastructure. Learn how Vultr powers cloud computing solutions for developers.

Contents

01	Introduction	3
02	Vultr Docs	3
03	Vultr Creator Dashboard	4
04	Cloud Infrastructure	6
05	Provisioning and Deployment	7
06	Resources	8
07	Conclusion	8

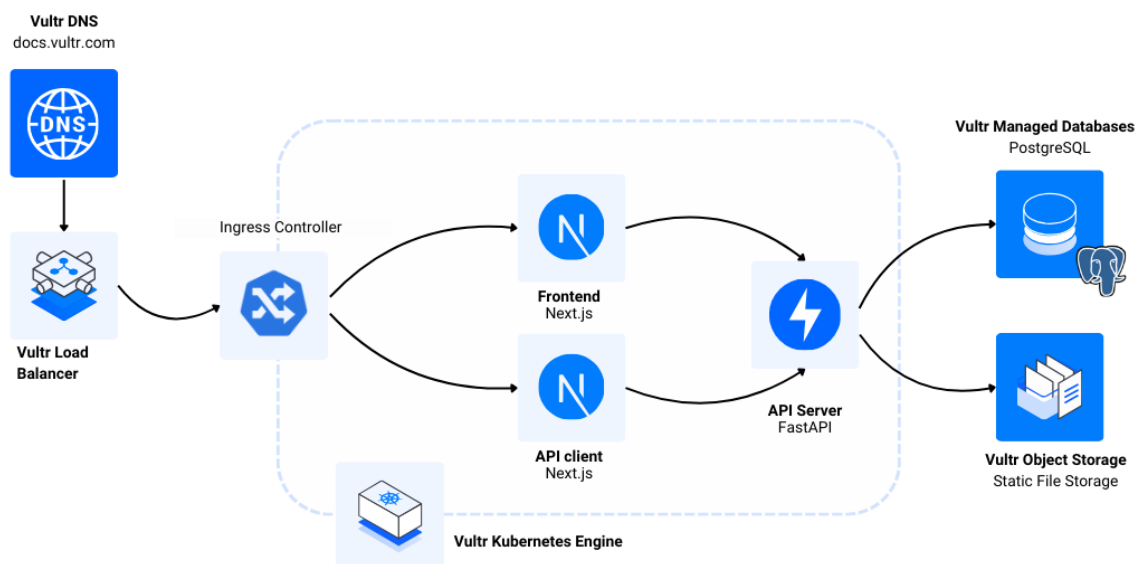
Introduction

[Vultr Docs](#) and [Vultr Creator Dashboard](#) are the two platforms used hand in hand to assist developers and their experience with Vultr products, Both platforms are built and scaled using cloud-native technologies and Vultr Cloud infrastructure.

This article explains various cloud-native technologies, tools, frameworks, and how they have been used to build the Vultr Docs and Vultr Creator Dashboard.

Vultr Docs

Vultr Docs is a library of technical guides, tutorials, and how-to articles that help developers and system administrators to get started with Vultr products and services. Most articles are authored by creators who are part of the Vultr Creator Program.



docs.vultr.com

In this section, you will learn about the various components that make up the Vultr Docs platform and understand the tech stack including different languages and prominent libraries.

Components

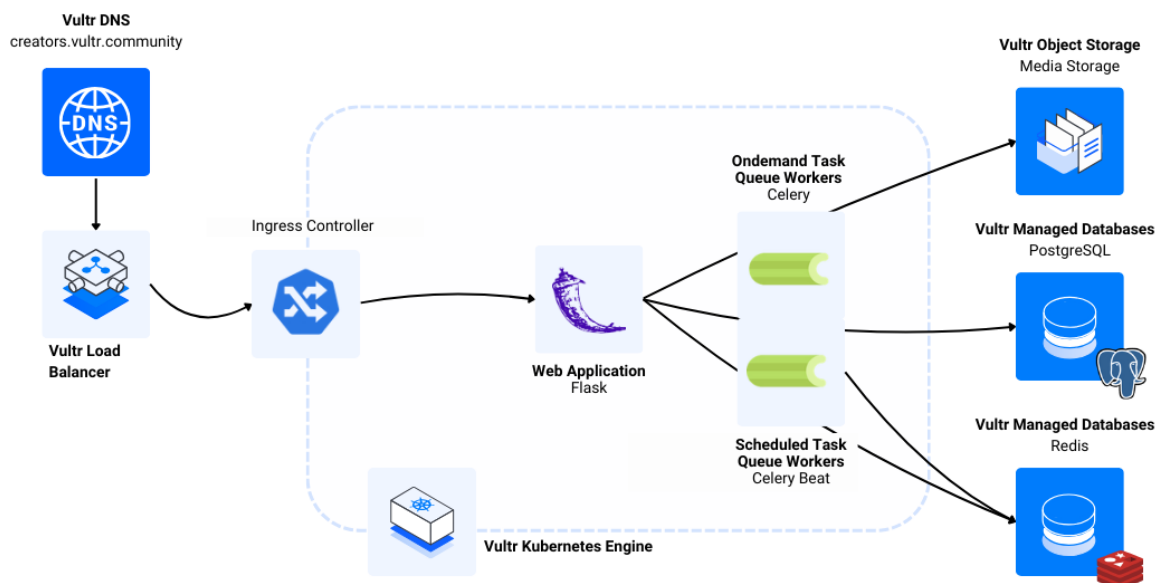
- **Frontend:** The Frontend of the Vultr Docs enables the functionality for the readers to interact with the documentation library and search for relevant information to get started with any of the Vultr products.
- **API Server:** The API Server for Vultr Docs is built using **FastAPI** a Python web framework, that enables the creation of efficient and scalable endpoints for the frontend and API client to interact.
- **API Client:** The API Client is a private component non-accessible to the readers, it is used to manage the Vultr Docs library via a user interface.

Tech Stack

- **Languages:** Python, JavaScript
- **Prominent Libraries and Packages**
 - **FastAPI:** Used to build the API server for Vultr Docs, it is a modern Python web framework that enables the creation of efficient and scalable endpoints.
 - **Next.js:** Used to build the frontend of the Vultr Docs, it is a React framework that enables the creation of server-side rendered applications.
 - **Tailwind CSS:** Used to style the components and ensure a consistent design across the platform.
- **Database Management**
 - **SQLAlchemy:** Used to interact with the PostgreSQL database.
 - **Alembic:** Used to handle all database migrations and also maintain the data integrity as the database changes.

Vultr Creator Dashboard

Vultr Creator Dashboard serves as the medium between Vultr Creators who want to contribute and Vultr Docs. The contributions made by the creators are published on the Vultr Docs and Vultr Creator Dashboard is the platform used to manage creators and their contributions.



docs.vultr.com

In this section, you will learn about the various components that make up the Vultr Creator Dashboard and understand the tech stack including different languages and prominent libraries.

Components

- **Web Application:** The Web Application is the platform where Vultr Creators are registered and use it for contributing towards Vultr Docs.
- **On-demand Task Queue Workers:** In **Celery** a distributed task queue system for Python, these workers are dynamically created based on the workload and they maintain efficient resource utilization by adjusting the workers according to the workload
- **Scheduled Task Queue Workers:** These workers are managed by **Celery Beat**, a scheduler included with celery. They are dedicated to executing tasks according to a predefined schedule or timetable, regardless of the current workload.

Tech Stack

- **Language:** Python

- **Prominent Libraries and Packages**

- **Flask:** Used to build the web application for Vultr Creator Dashboard.
- **Celery:** Used to manage the task queues and execute tasks asynchronously.
- **Celery Beat:** Used to manage the scheduled tasks and execute them at the predefined time.
- **Bootstrap:** Used to style the components and ensure a consistent design across the platform. Jinja2 is used as the templating engine. JavaScript is used for client-side interactivity.

- **Database Management**

- **SQLAlchemy:** Used to interact with the PostgreSQL database.
- **Alembic:** Used to handle all database migrations and also maintain the data integrity as the database changes.

Cloud Infrastructure

This section breaks down the various parts of the Vultr cloud-native infrastructure on which the Vultr Docs and the Vultr Creator Dashboard rely. These infrastructure products serve as the foundational block to make the platforms reliable and scalable globally across all locations.

- **Vultr Kubernetes Engine:** To ensure high availability and scalability across the globe, both platforms are deployed and managed using Vultr Kubernetes Engine.
 - **Vultr Optimized Cloud Compute:** Vultr Docs and Vultr Creator Dashboard are hosted on Vultr Optimized Cloud Compute Instances, These instances act as Kubernetes cluster nodes, hence the availability of Optimized Cloud Compute Instances across 32 data center locations globally provides resilience and maximum uptime guarantee to the platform.
 - **Vultr Block Storage:** Stores data as block objects built using the Vultr Storage Class for Persistent Volumes to provision PVCs and automatically mount storage units onto pods
 - **Vultr Load Balancer:** The Kubernetes cluster also utilizes the **Vultr Load Balancer** integrated service that acts as an open endpoint for

the cluster, it also helps to manage the load between multiple pods efficiently. **Kubernetes Ingress Controller** is used to manage external access to the services inside the cluster.

- [Vultr Managed Databases](#)
 - [Vultr Managed Databases for PostgreSQL](#): Used in both Vultr Docs and Vultr Creator Dashboard to store and manage the databases and ensure availability, atomicity, and scalability.
 - [Vultr Managed Databases for Caching](#): Used for the Vultr Creator Dashboard for the execution of scheduled tasks using Celery.
- [Vultr Object Storage](#): Every type of file and media in Vultr Docs and images/videos in the Vultr Creator Dashboard are stored using the Vultr Object Storage.
- [Vultr DNS](#): Vultr DNS is utilized for managing the DNS records for both Vultr Docs and the Vultr Creator Dashboard. Additionally, it is configured to point to the IP address(es) associated with the Vultr Load Balancer to handle load distribution.

Provisioning and Deployment

The provisioning and deployment of the Vultr Docs and Vultr Creator Dashboard are managed using Terraform and GitHub Actions respectively.

- **Terraform**: Used to provision the cloud infrastructure on Vultr, it is a tool for building, changing, and versioning infrastructure safely and efficiently. The Terraform configuration files are used to define the infrastructure components and their dependencies.
- **Docker**: Used to containerize the applications and manage the dependencies, it is used to build the images for the Vultr Docs and Vultr Creator Dashboard.
- **Kubernetes**: Used to manage the containerized applications and ensure high availability and scalability across the globe. The Kubernetes cluster is used to deploy the Vultr Docs and Vultr Creator Dashboard.

- **GitHub Actions:** Used to automate the deployment process of the Vultr Docs and Vultr Creator Dashboard. The GitHub Actions workflow is triggered on every push to the main branch and it builds the Docker images, pushes them to the Vultr Container Registry, and deploys the updated images to the Kubernetes cluster.

Resources

- [Set up Nginx Ingress Controller with SSL on Vultr Kubernetes Engine.](#)
- [Implement a CI/CD Pipeline with GitHub Actions and Vultr Kubernetes Engine](#)
- [Provision Cloud Infrastructure on Vultr using Terraform](#)
- [Manage Users in Vultr Managed Databases for PostgreSQL](#)

Conclusion

In this article, you understood the functionality of Vultr Docs and Vultr Creator Dashboard respectively. You also explored the technical infrastructure and cloud-native technology used to build the two platforms. In future developments, Vultr Docs will also be integrated with the all new [Vultr CDN](#) to serve content with no disruptions, and the functionality of the recently released [Vultr Cloud Inference](#) is also going to be used to improve the overall experience for the developers, readers, and community members.



VULTR

