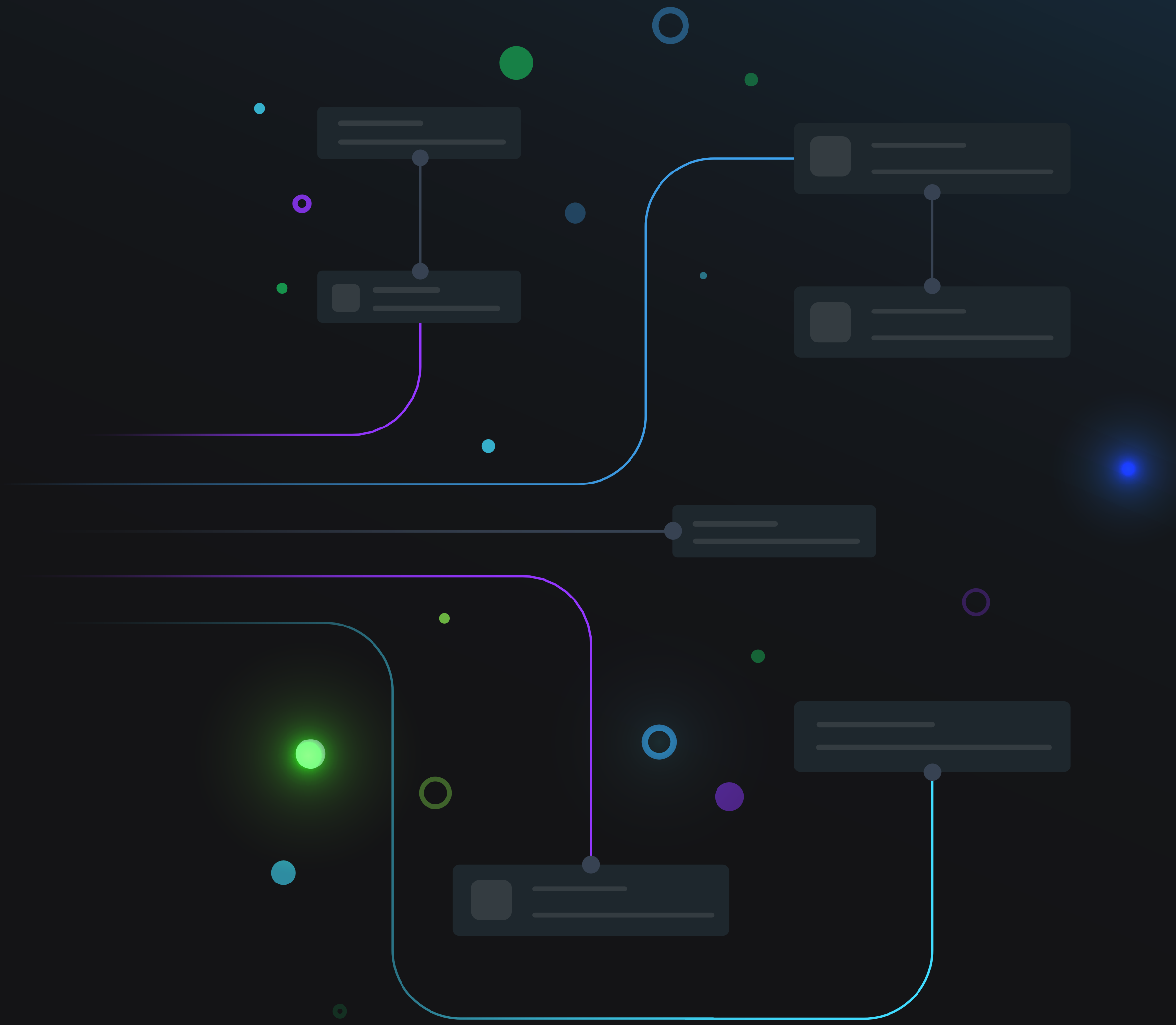


WHITE PAPER

Infrastructure **from** Code

Solution Overview

StackGen

Table of Contents

Introduction

In the rapidly evolving landscape of cloud-native application deployment, “Infrastructure from Code” represents a pivotal advancement. This approach, which automates infrastructure provisioning and management by analyzing application code, is emerging as the next crucial step for organizations seeking to enhance scalability, improve security and velocity, and ensure consistent compliance.

As businesses increasingly rely on cloud environments, traditional methods of managing infrastructure through manual processes or static scripts are proving inadequate. At the same time, advancements in template-based automation have not proven widely scalable beyond platform teams to developers. Infrastructure from code addresses these challenges by enabling dynamic, programmable, and version-controlled infrastructure, thereby streamlining deployment processes and reducing the risk of human error.

One of the key advantages of infrastructure from code is its seamless integration with existing toolsets, making it the most straightforward path for organizations aiming to modernize their infrastructure pipelines. Whether it’s integrating with CI/CD workflows, leveraging Infrastructure as Code (IaC) tools like Terraform or OpenTofu, or utilizing configuration management tools, infrastructure from code provides a cohesive and flexible solution that enhances the efficiency and reliability of cloud operations.

Furthermore, infrastructure from code is highly adaptable, catering to both individual developers and large-scale enterprise operations. For single users, it offers an accessible way to manage personal projects with professional-grade infrastructure practices. For platform engineering teams within enterprises, infrastructure from code facilitates consistent, scalable, and secure infrastructure management across multiple environments and applications.

This whitepaper delves into how StackGen is providing Infrastructure from Code by first demonstrating the transformative potential of infrastructure from code, exploring its benefits, implementation strategies, and integration techniques. By adopting infrastructure from code, organizations can achieve a new level of agility and resilience in their cloud-native journeys, positioning themselves for sustained success in an increasingly competitive digital landscape.

What is Infrastructure from Code

Infrastructure from Code is a paradigm that automates the provisioning, configuration, and management of IT infrastructure by generating IaC that is more tightly aligned with application code. In the words of Infoq.com:

> Infrastructure-from-Code (IfC) is an approach that creates, configures, and manages cloud resources understanding a software application's source code, without explicit description.

Infrastructure from Code is built on the premise that the application code is the source of truth. IaC should be generated based on what the application requires vs. retrofitting templated IaC to fit the application.

At StackGen, Infrastructure from Code means we scan application code, create a view of the deployment architecture and apply all the standards and policies required when generating the IaC. We believe that Infrastructure from Code should:

- Require no application code standard changes
- Help avoid learning new programming languages, as IaC is generated based on languages already in use
- Require no user to be an expert in cloud architectures
- Apply least-privileged access controls by default
- Only create the infrastructure needed for the application, no overprovisioning due to one-size-fits-all templating
- Be easy, intuitive, and fit in your existing build-deploy workflows

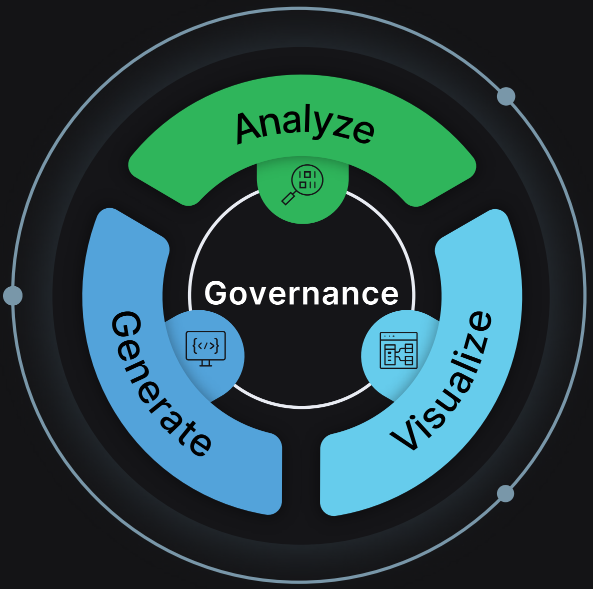
Infrastructure from Code builds upon and expands the benefits of IaC. The principles of IaC are rooted in automation, repeatability, and version control, enabling infrastructure to be treated with the same rigor and discipline as application code. By writing declarative or imperative scripts, developers and IT professionals can define the desired state of their infrastructure, which tools like Terraform, AWS CloudFormation, or Pulumi can then execute to create and manage resources. This approach ensures that infrastructure changes are consistent, auditable, and reproducible, minimizing human error and enhancing operational efficiency. Infrastructure from Code holds these same principles and harnesses the same technologies, but abstracts away from scripts and configuration to enable users of all experience levels to safely and efficiently generate the infrastructure they need.

A key aspect of infrastructure from code is the move from templates to standards. Templates are predefined configurations that provide a quick and easy way to deploy common infrastructure setups. They serve as blueprints that can be reused and adapted for various projects, ensuring consistency and speeding up the deployment process. Underlying these templates are standards. Standards refer to the best practices and policies that govern how infrastructure should be configured and managed. Adhering to these standards ensures that all infrastructure deployments are secure, compliant, and optimized for performance. While templates provide the 'how', standards define the 'why' and 'what' of infrastructure management, guiding organizations toward more robust and scalable implementations. Standards drive acceleration and velocity of deployments. If the how (templates) can be automated, platform teams are freed to focus solely on the why and what (standards); this is the goal of infrastructure from code.

For developers, infrastructure from code offers numerous benefits. It simplifies the infrastructure management process, allowing developers to focus more on coding and less on manual infrastructure configuration tasks. With infrastructure from code, developers can easily spin up and tear down environments as their application development needs change, facilitating rapid development and testing cycles. This agility accelerates the software development lifecycle, enabling quicker iteration and innovation. Additionally, infrastructure from code is closely aligned with application versions. It allows infrastructure changes to be tracked and rolled back more easily if necessary, reducing the risk of errors and improving the overall stability of the development environment.

Enterprises stand to gain significantly from adopting infrastructure from code. By codifying infrastructure based on the application itself, organizations can achieve greater consistency and reliability across their IT environments, along with a 'best fit' approach to provisioning. This is particularly valuable for large-scale operations where manual configuration is not only time-consuming but also prone to inconsistencies, and over-provisioning. Infrastructure from code enhances security and compliance by automatically embedding policies directly into the infrastructure code, ensuring that all deployments adhere to organizational standards from their inception. Moreover, the automation of infrastructure management reduces operational overhead, freeing up platform teams to focus on more strategic initiatives. Ultimately, infrastructure from code empowers enterprises to build more resilient, scalable, and efficient infrastructure, supporting their growth and digital transformation goals.

If the how (templates) can be automated, **platform teams are freed to focus solely on the why and what** (standards); this is the goal of infrastructure from code.



StackGen empowers users to **validate and optimize their infrastructure configurations** before deployment.

How StackGen Works

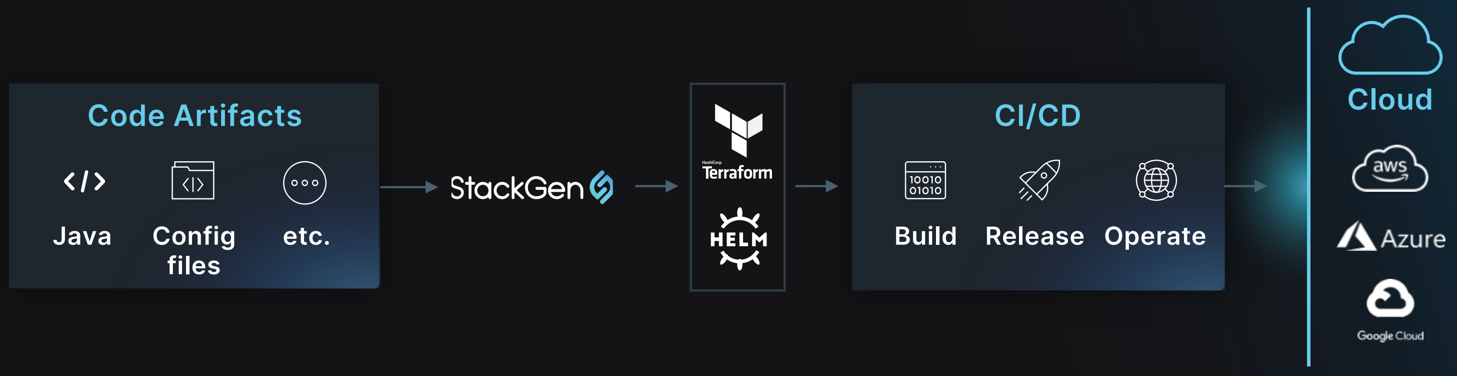
StackGen streamlines application deployment and infrastructure provisioning by generating IaC tailored to an application's requirements. StackGen automates IaC generation from application code while enforcing golden standards and ensuring compliance with industry regulations.

By leveraging sophisticated analysis of Python and Java codebases, StackGen identifies an application's infrastructure requirements, including cloud dependencies, APIs, service configurations, databases, and environment variables.

Using this analysis, StackGen generates Terraform or Helm charts, adhering to predefined golden standards and policies. These standards encompass a wide range of regulatory frameworks such as SOC 2, HIPAA, NIST-CSF, PCI, and GDPR, as well as best practices for least privileged access, Identity and Access Management (IAM), infrastructure security, and compliance validation. With StackGen, organizations can ensure that their infrastructure deployments not only meet regulatory requirements but also follow industry-leading security practices.

StackGen's intuitive interface allows users to visualize the deployment architecture defined by their IaC, providing insights into resource connections and dependencies. With the ability to drag and drop resources, while ensuring all connections have necessary guardrails, StackGen empowers users to validate and optimize their infrastructure configurations before deployment. This ensures that deployments are efficient, secure, and compliant from the outset.

StackGen Workflow



The platform seamlessly integrates with public and private repositories, Source Control Management providers (SCMs), and local repositories. Furthermore, StackGen simplifies application management by extracting application traits into a unified workload specification with versioning functionality.

Supported languages include Java and Python, and AWS and Azure are currently the primary supported cloud platforms. Additional languages are being added and version specifications can be viewed **here**. StackGen is compatible with popular CI/CD platforms such as Jenkins, CircleCI, and GitHub. StackGen also supports Helm chart creation, enhancing Kubernetes applications' deployment, and ensuring a seamless transition to automated infrastructure deployment for development teams.

StackGen complements existing Secure Software Development Lifecycle (SSDLC) and policies and integrates into existing workflows so there is no need to rebuild existing pipelines.

Users can try StackGen via the Developer Edition, a SaaS version or on-prem. Users can also use the StackGen CLI to access all editions.

StackGen Concepts

There are three key features encapsulating StackGen's process:

1 **appStack**

2 **Topology**

3 **Generated IaC**

appStack

An appStack is a collection of components that make up an application. These repositories, when considered together, help StackGen understand an application's architecture and generate appropriate IaC for provisioning and deploying the infrastructure an application needs.

appStacks are currently limited to one cloud provider and one cloud service. If a user wanted to see infrastructure for an app on both AWS ECS and AWS EKS, the user would need to create two separate appStacks for the application.

When creating an appStack, users are prompted to provide specific information to help StackGen accurately analyze and set up an application's environment. Information required includes:

appStack Name

- **Description:** The unique name assigned to an appStack for identification.
- **Format:** Enter a descriptive name without spaces, which should be URL-safe.

Description (optional)

- **Description:** A brief overview of what an appStack is and what it represents.
- **Format:** Free text field, optional for additional context.

Target Cloud Service

- **Description:** Specify the cloud service provider where an application will be deployed.
- **Options:** Choose from cloud services like AWS, Azure, EKS or AKS.

Topology

The Topology view in StackGen offers an interactive representation of an application's infrastructure architecture. The topology is the primary way to edit an architecture in StackGen. The various elements available to build infrastructure are inferred from the application code, such as services, databases, and storage, and are visualized for quick review and easier understanding of interconnections and dependencies.

The Topology view is dynamic, allowing users to fine-tune parameters, add or remove resources, and establish connections to accurately model an application infrastructure architecture.

Generated IaC

StackGen automates the creation of IaC files, crucial for the consistent and repeatable provisioning and management of infrastructure. After analyzing an application appStack, and getting any inputs from the topology, StackGen produces IaC files stored locally.

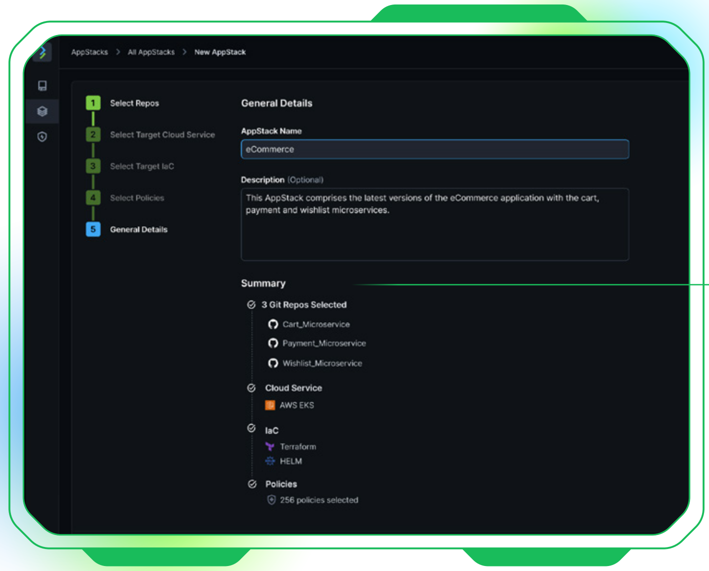
Locating Generated IaC Files

IaC generated through the SaaS version will download to wherever a user has configured a browser to download files.

IaC generated through the CLI will be directed to the destination specified.

Analyze, Visualize, Generate

StackGen has three main functions: analyze, visualize and generate.



AppStacks > All AppStacks > New AppStack

1 Select Repos
2 Select Target Cloud Service
3 Select Target IaC
4 Select Policies
5 General Details

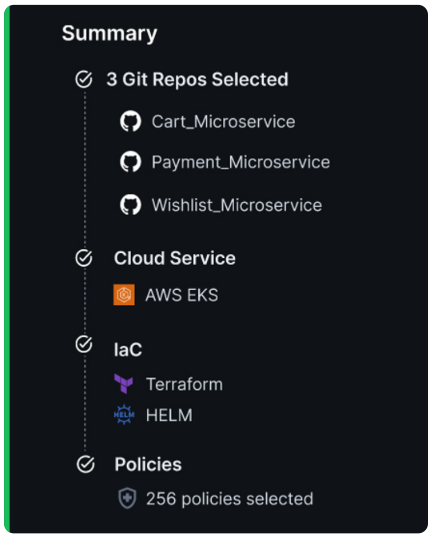
General Details

AppStack Name
eCommerce

Description (Optional)
This AppStack comprises the latest versions of the eCommerce application with the cart, payment and wishlist microservices.

Summary

- 3 Git Repos Selected
  - Cart_Microservice
  - Payment_Microservice
  - Wishlist_Microservice
- Cloud Service
  - AWS EKS
- IaC
  - Terraform
  - HELM
- Policies
  - 256 policies selected



Summary

- 3 Git Repos Selected
  - Cart_Microservice
  - Payment_Microservice
  - Wishlist_Microservice
- Cloud Service
  - AWS EKS
- IaC
  - Terraform
  - HELM
- Policies
  - 256 policies selected

1. Analyze

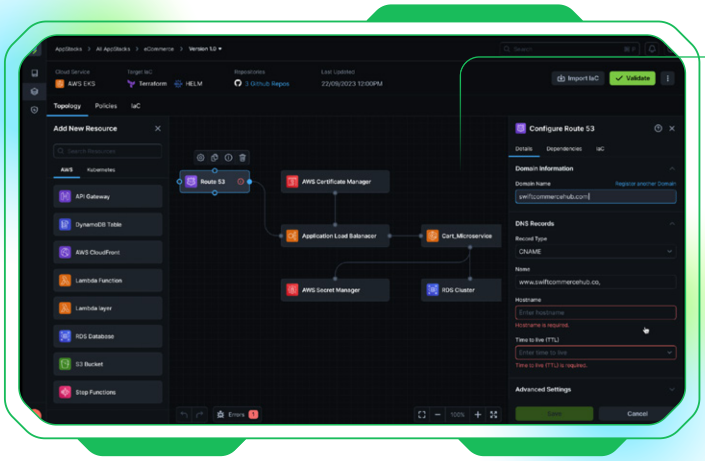
The analyze function applies to applications written in Java and Python languages, with more coming soon. It performs a static code analysis of source code files, Dockerfiles, configuration files, and any other relevant files in a Git repository. The static analysis is a point-in-time process that temporarily clones a user's git repositories for analysis, identifies what infrastructure is required during provisioning, and then removes the clone. If the code changes, a new analysis can be done and IaC updated with version control features within StackGen.

Key features include

Extract Traits: StackGen looks at the core of an application component to extract all pertinent traits from the source files. Whether it's storage, network interactions with other microservices, or connections to cloud provider services, StackGen comprehensively analyzes and brings to light the integral components of an application.

Deduplication: The deduplication process ensures that any redundant traits extracted from different source files are identified and removed. This streamlining step is crucial, guaranteeing that the analysis is not only accurate but also clutter-free, focusing solely on unique attributes that matter.

Manifest File Generation: Beyond simple analysis, StackGen takes it a step further by generating a detailed, combined manifest file. This file encompasses a component dependency graph specification alongside all extracted traits of components of the application, providing a blueprint of an application's infrastructure needs and interactions.

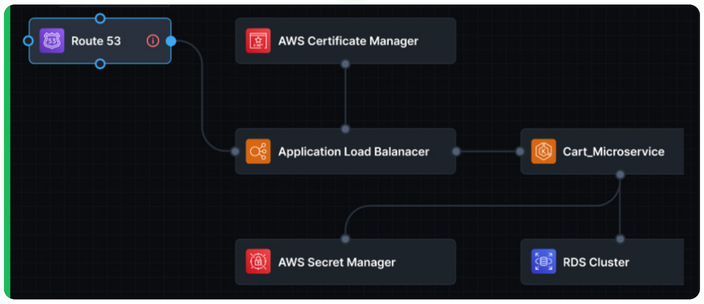


Cloud Service: AWS EKS
Target IaC: Terraform, HELM
Repositories: 3 Github Repos
Last Updated: 22/09/2023 12:00PM

Domain Information
Domain Name: swiftcommercehub.com (Register another Domain)

DNS Records
Record Type: CNAME
Name: www.swiftcommercehub.co,
Hostname: Enter hostname
Hostname is required.
Time to live (TTL): Enter time to live
Time to live (TTL) is required.

Advanced Settings



2. Visualize

Once the analysis is complete, StackGen provides a visualization of the deployment architecture. Visualizing the topology enables insight into the entire system including its dependencies and any policy violations from dragged-and-dropped resources. This visibility enables proactive monitoring and management, allowing users to detect and address issues before IaC is created, while still minimizing inputs compared to templates.

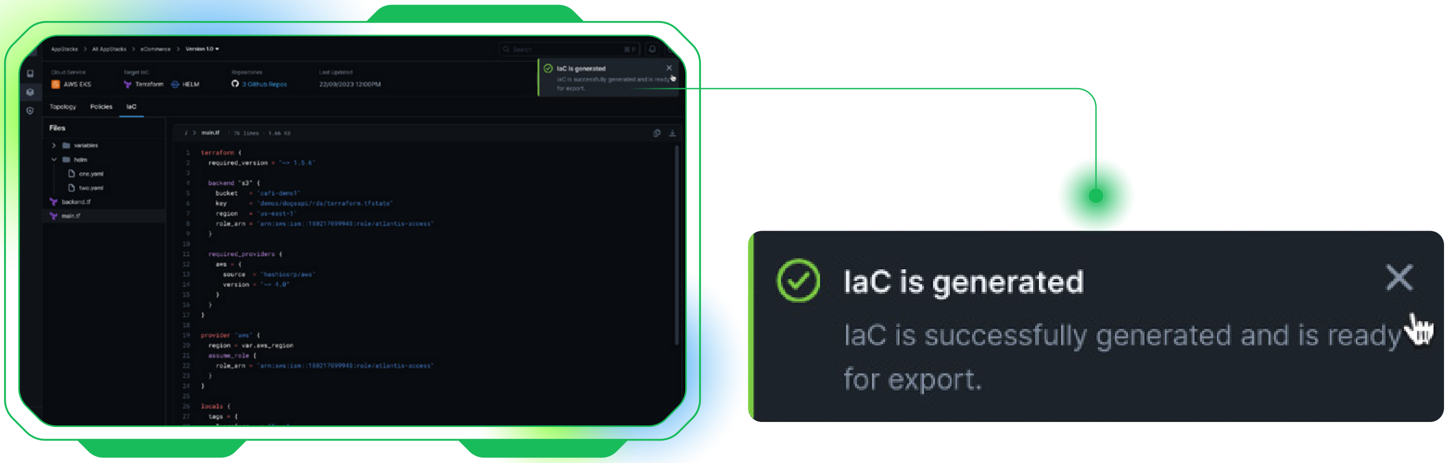
Key features include

User Interface: The user interface displays the topology of an application, derived from the analyzed traits. This visual representation allows users to quickly grasp the complex interactions within their applications, enabling informed decision-making and streamlined infrastructure planning.

Drag and Drop: Users can enhance the deployment architecture by dragging and dropping resources. StackGen does not allow connections to meet unless it complies with policies, and will call out defaults of newly added resources that violate policies so they can be fixed. Developers who want to build their own architecture from scratch (bypassing the analyze step) can use the visualization feature.

Validate: The topology is validated as the user interacts with the deployment architecture to ensure it complies with all standards. Policy violations are obvious and users are prevented from exporting IaC until it is in compliance.

Well-Architected Framework: StackGen visualization helps you achieve AWS Well-Architected Framework, a set of foundational questions that help you to understand if a specific architecture aligns well with cloud best practices.



```
terraform {
  required_version = "~> 1.5.6"

  backend "s3" {
    bucket   = "cafi-demo1"
    key      = "demos/dogeapi/rds/terraform.tfstate"
    region   = "us-east-1"
    role_arn = "arn:aws:iam::180217099948:role/atlantis-access"
  }

  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.0"
    }
  }
}

provider "aws" {
  region = var.aws_region
  assume_role {
    role_arn = "arn:aws:iam::180217099948:role/atlantis-access"
  }
}

locals {
  tags = {
```

IaC is generated

IaC is successfully generated and is ready for export.

3. Generate

Built on the foundation laid by the analysis, StackGen transforms the analyzed manifest file and topology into IaC files (Terraform or Helm) tailored for a seamless cloud deployment. StackGen supports AWS, Azure, Kuberentes and local-clusters. This doesn't just transcribe the application's needs into code; it infuses the process with standards that address security, compliance, and deployment requirements. The file format allows direct integration into existing pipelines, no need to retool.

Key features include

Access Control: StackGen offers robust access control mechanisms. By generating IaC files that define minimal access, it ensures that the various components or microservices of an application have only the necessary permissions, significantly reducing the attack surface.

Support for Deployment Scenarios: StackGen can generate IaC files for single-node installations, multi-node setups, or complex cluster deployments. This versatility ensures that infrastructure can grow and adapt alongside the application.

Customization and Best Practices: Understanding that each application is unique, StackGen offers extensive customization options for the generated IaC files. Tailor resource names, bundles, and configurations to meet the specific needs of a project, all while adhering to industry best practices for infrastructure deployment.

Versioning: StackGen can manage the lifecycle of the IaC deployment files by maintaining the different versions of the IaC files. It can also provide a diff so users can track, monitor and review changes over time.

Policies Applied

During IaC generation, StackGen can apply AWS and Azure best practices through pre-packaged policies. By applying policies at IaC generation, organizations can reduce risk by enforcing security and compliance at creation. Both AWS and Azure policies by domain include:

- Infrastructure security
- Identity and access management
- Security Best Practices
- Messaging
- Data Protection
- Logging and monitoring
- Compliance validation
- Resilience

AWS POLICIES BY FRAMEWORK	AZURE POLICIES BY FRAMEWORK
AWS Security	CIS
CIS for Azure, AWS, GKE in various levels	NY-DFS
SOC2	AWS Security
GDPR	CSC
Cloud Cost Management (CCM)	CIS - Azure, GKE
PCI	SOC2
NIST	
HIPAA	
ISO	
AWS Well Architected Framework	
NY-DFS	

Compatibility

StackGen is compatible with:

Operating System
Windows, macOS, and Linux.

Cloud Providers
AWS, Azure and private clouds.
GCP and local clouds coming soon.

Git Repositories
Git repositories hosted on any Git hosting provider, such as GitHub, GitLab, and Bitbucket.

Programming Languages
Java and Python with Go, and JavaScript coming soon.

Summary

Infrastructure from Code represents the future of IaC due to its transformative impact on agility, reliability, and scalability in modern software development and deployment practices. By auto-generating infrastructure configurations and management processes, infrastructure from code enables organizations to automate and standardize deployment workflows, reducing manual errors and enhancing consistency across environments. This approach not only accelerates the pace of software delivery but also facilitates rapid iteration and innovation by providing developers with the flexibility to spin up and tear down infrastructure resources on-demand.

Moreover, infrastructure from code aligns seamlessly with DevOps principles, fostering collaboration between development and operations teams and promoting a culture of continuous improvement. As organizations increasingly embrace cloud-native architectures and dynamic deployment models, infrastructure from code emerges as the cornerstone for achieving efficient, secure, and compliant infrastructure management, positioning it at the forefront of IaC evolution.

StackGen provides organizations of all sizes the generative infrastructure from code solution that requires no application code changes and no upheaval in existing processes. It allows teams to create secure and standardized IaC without introducing SDLC bottlenecks. From small-scale applications to enterprise-level systems, StackGen ensures that the deployment strategy is efficient, secure and scalable.

StackGen

At StackGen, we want to enable every organization to securely deploy applications without delay by being application-centric, cloud agnostic and developer-first. Our mission is to remove the burden of infrastructure as code by auto-generating it from the application with golden standards applied.

stackgen.com info@stackgen.com