

Cloud Federation with Radius

Zach Casper

Principal Product Manager
Azure Incubations





Azure Incubations

MISSION

Partner across Microsoft and the open-source community to explore and deliver industry-changing products



GRADUATED
Aug 2023



GRADUATED
Nov 2024



SANDBOX
Apr 2024



SANDBOX
Sep 2023



SANDBOX
Jan 2025

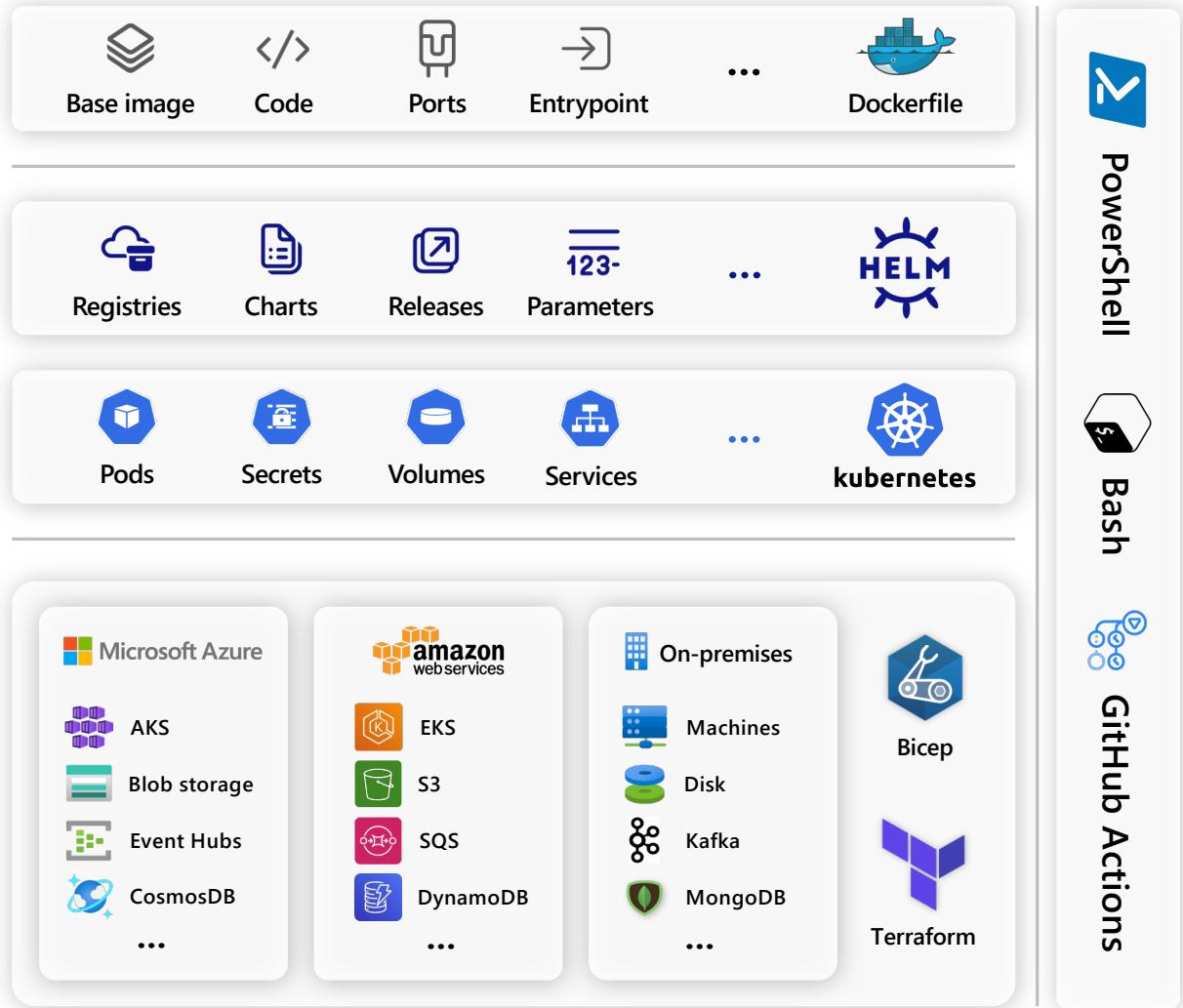




**Cloud-native application platform
that works on any cloud**



Cloud native apps are more than just **kubernetes**



```

terraform {
  required_providers {
    kubernetes = {
      ...
    }
  }

  resource "kubernetes_deployment" "postgres" {
    spec {
      container {
        name  = "postgres"
        image = "postgres:16-alpine"

        env_from {
          secret_ref {
            name = ...
          }
        }
      }
    }
  }

  resource "kubernetes_service" "postgres" {
    spec {
      selector = {
        app = "postgres"
      }
      port {
        port      = 5432
        target_port = 5432
      }
      type = "ClusterIP"
    }
  }
}

```



```

terraform {
  required_providers {
    aws = {
      ...
    }
  }

  resource "aws_db_instance" "postgres" {
    identifier          = "demo-postgre"
    engine               = "postgres"
    engine_version       = "16.3"
    instance_class       = "db.t3.micro"
    allocated_storage    = 20
    db_name              = var.db_name
    username             = var.db_username
    password             = ...
    db_subnet_group_name = ...
    vpc_security_group_ids = ...
    skip_final_snapshot = true
    publicly_accessible = true
    deletion_protection = false
  }

  ...
}

```



```

terraform {
  required_providers {
    azurerm = {
      ...
    }
  }

  resource "azurerm_postgresql_flexible_server" "postgres" {
    name           = ...
    resource_group_name = ...
    location       = ...
    administrator_login = var.admin_username
    administrator_password = ...
    version        = "16"
    sku_name       = "B_Standard_B1ms"
    storage_mb     = 32768
    backup_retention_days = 7
    zone          = "1"
    authentication {
      password_auth_enabled = true
    }
  }

  resource "azurerm_postgresql_flexible_database" "app_db" {
    name      = var.db_name
    server_id = ...
    collation = "en_US.utf8"
    charset   = "UTF8"
  }
}

```



Multi-Cloud Challenges

How do developers define their application resources across different cloud platforms?

How do developers work across multiple cloud platforms seamlessly?

How do developers control what resources are used in the underlying cloud environment?

How do developers manage their apps and application resources across multiple cloud platforms?



How do developers define their application resources across different cloud platforms?

How do developers work across multiple cloud platforms seamlessly?

How do control what resources are used in the underlying cloud environment?

How do developers manage their apps and application resources across multiple cloud platforms?



Radius Resource Types

Custom-defined resource types agnostic of any cloud platform.



Radius Environments

Define the cloud provider, the region, and how to deploy resources



Infrastructure Recipes

Define platform-specific configurations using existing IaC tools that enforce security and cost best-practices

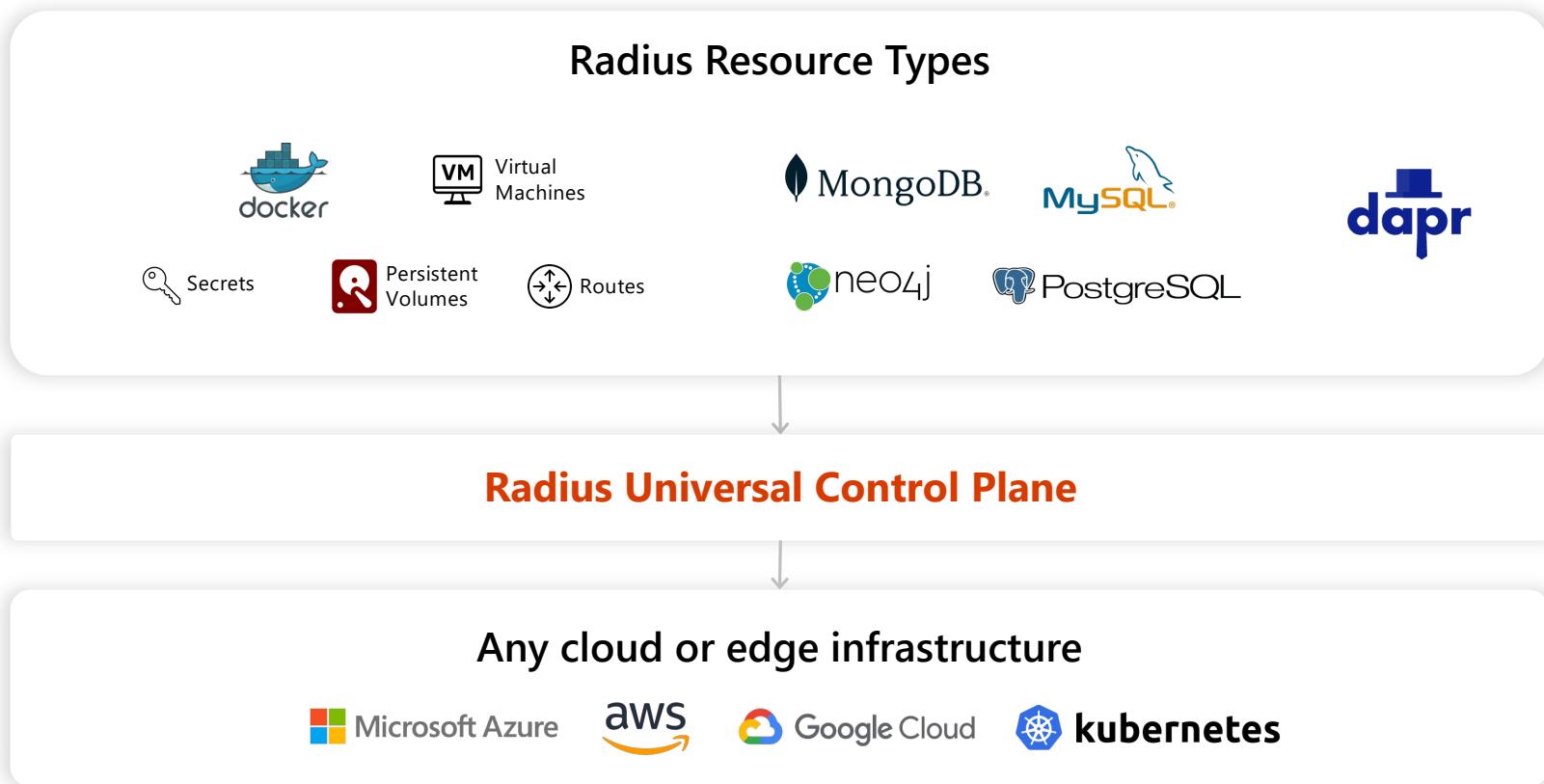


CLI, Dashboard, & IDE Integration

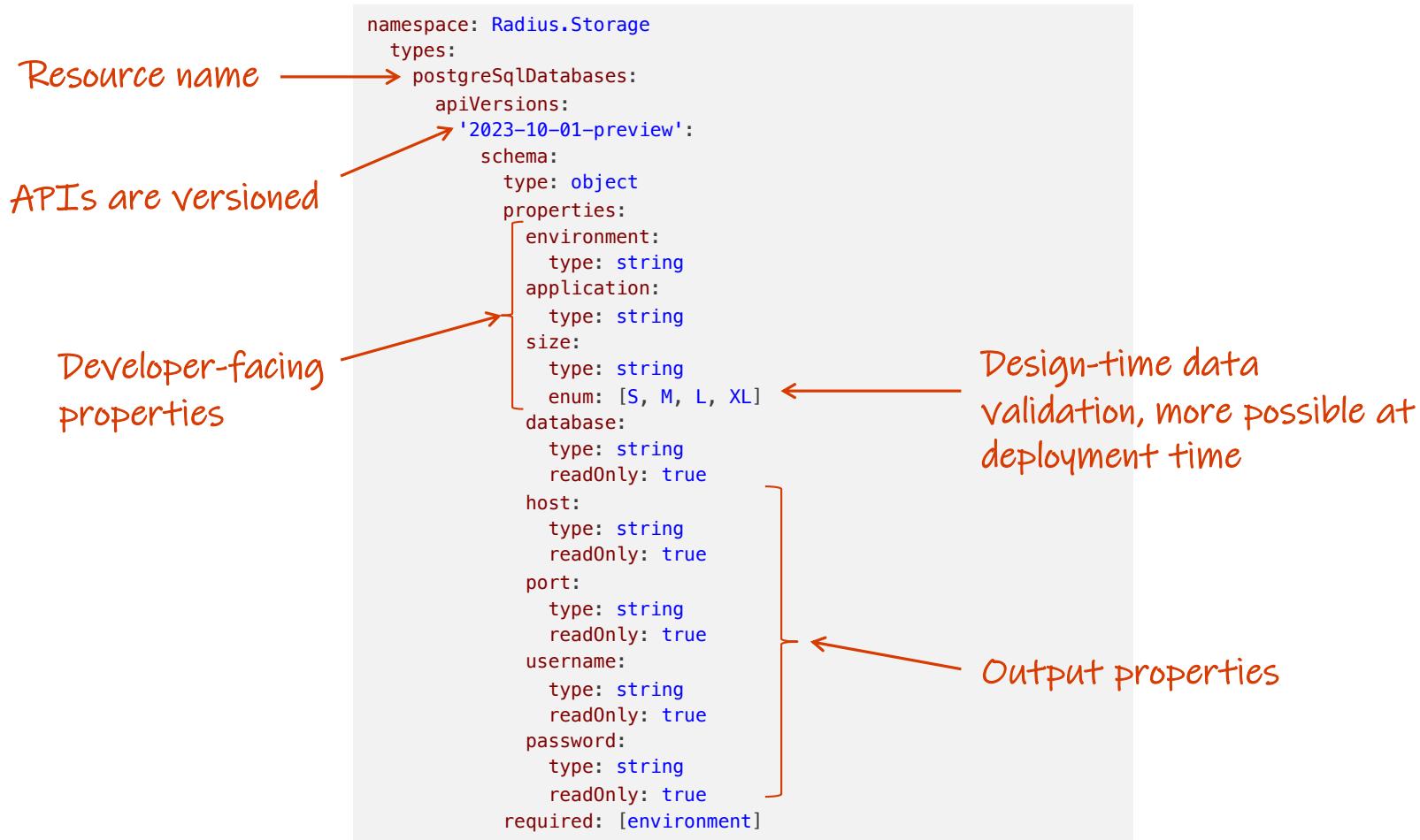
Radius CLI deploys any cloud platform via the Radius control plane. Works with VS Code IntelliSense.



Defining cross-platform resource types



Example: Blob Storage Bucket



How do developers define their application resources across different cloud platforms?

How do developers work across multiple cloud platforms seamlessly?

How do control what resources are used in the underlying cloud environment?

How do developers manage their apps and application resources across multiple cloud platforms?



Radius Environments

Define the cloud provider, the region, and how to deploy resources



Radius Environments

The screenshot shows the Radius application interface. On the left is a dark sidebar with navigation icons for Home, Resource Types, Environments, Applications, Resources, and Recipes. The main area has a header "Environment" and a sub-header "Displaying details for Applications.Core/environments: aws-eu-central-1". Below this is a breadcrumb trail: Home / Environments / aws-eu-central-1. There are three tabs: OVERVIEW (selected), RESOURCES, and DETAILS. The OVERVIEW section displays the environment name "customer-1 / aws-eu-central-1" and two cluster details: "Kubernetes Cluster" and "Radius control plane cluster". To the right of these are the AWS Account ID (817312594854) and AWS Region (eu-central-1). At the bottom is a "Settings" button.

The screenshot shows the Radius application interface. On the left is a dark sidebar with navigation icons for Home, Resource Types, Environments, Applications, Resources, and Recipes. The main area has a header "Environments" and a sub-header "Displaying environments where applications can be deployed". Below this is a breadcrumb trail: Home / Environments. A filter dropdown is set to "customer-1". The main content is a table titled "Environments" with columns: NAME, RESOURCE GROUP, and KIND. The data rows are:

NAME	RESOURCE GROUP	KIND
aws-eu-central-1	customer-1	AWS
aws-eu-west-1	customer-1	AWS
az-northeurope	customer-1	Azure
az-westeurope	customer-1	Azure
cz1-ktis	customer-1	Kubernetes
it1-arezzo	customer-1	Kubernetes
it2-arezzo	customer-1	Kubernetes
it3-bergamo	customer-1	Kubernetes
it4-rome	customer-1	Kubernetes



How do developers define their application resources across different cloud platforms?

How do developers work across multiple cloud platforms seamlessly?

How do control what resources are used in the underlying cloud environment?

How do developers manage their apps and application resources across multiple cloud platforms?

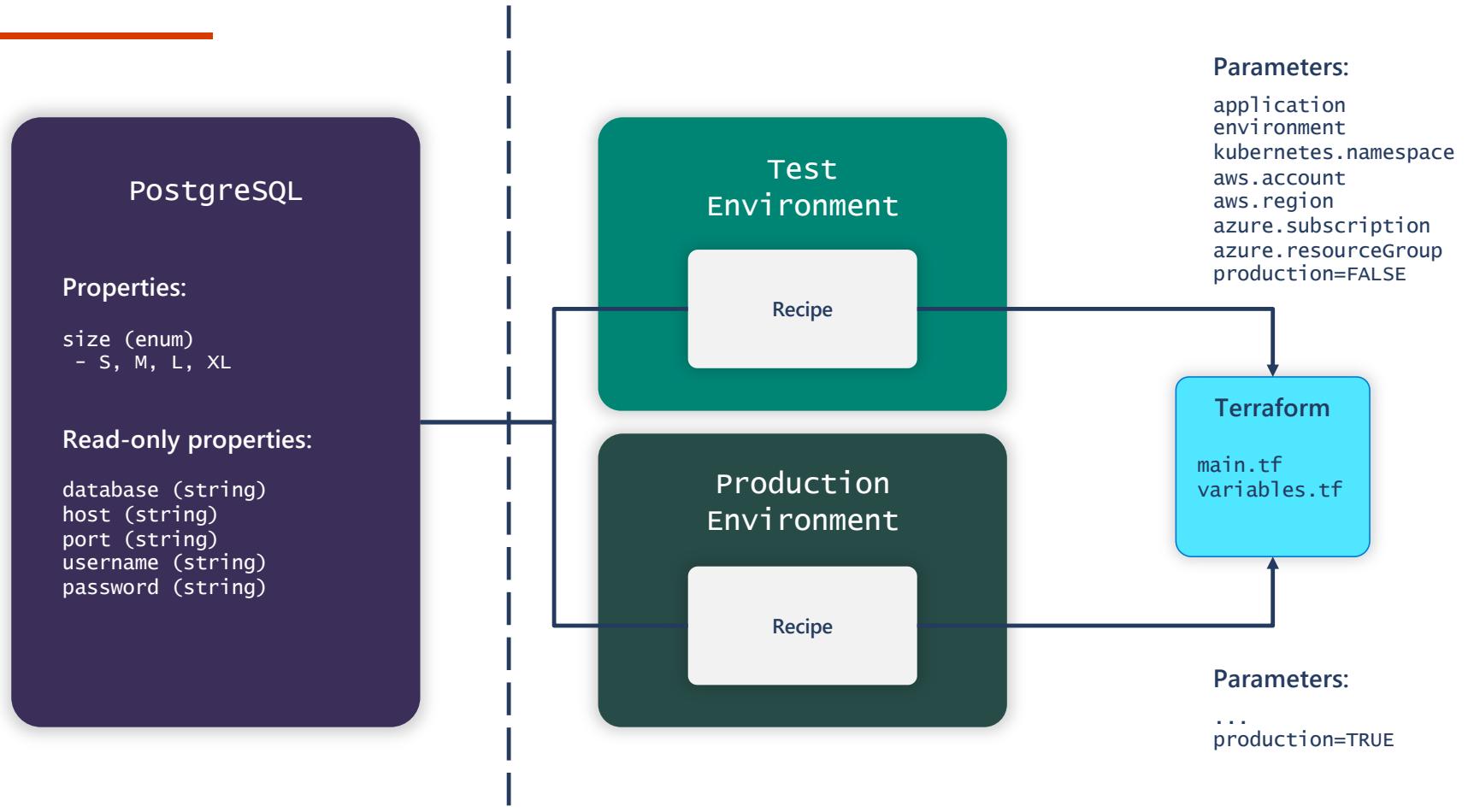


Infrastructure Recipes

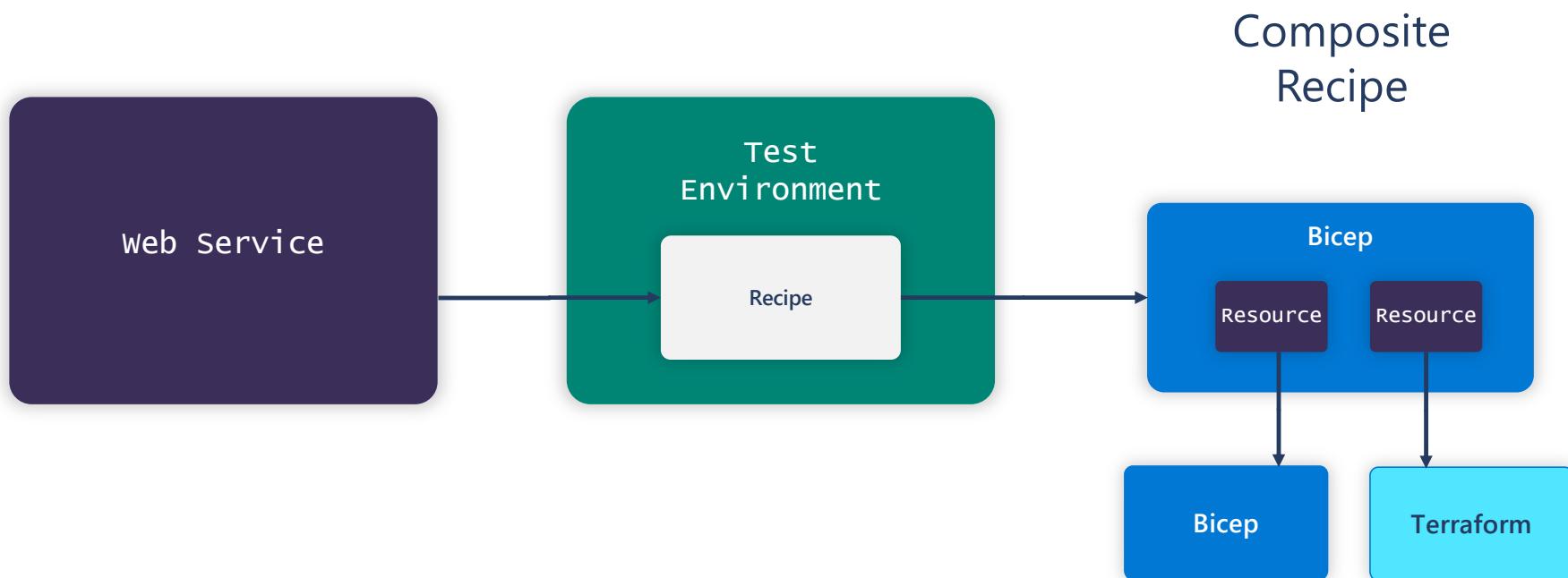
Define platform-specific configurations using existing IaC tools that enforce security and cost best-practices



Recipes



Recipes



How do developers define their application resources across different cloud platforms?

How do developers work across multiple cloud platforms seamlessly?

How do developers control what resources are used in the underlying cloud environment?

How do developers manage their apps and application resources across multiple cloud platforms?

>_

CLI, Dashboard, & IDE Integration

Radius CLI deploys any cloud platform via the Radius control plane. Works with VS Code IntelliSense.

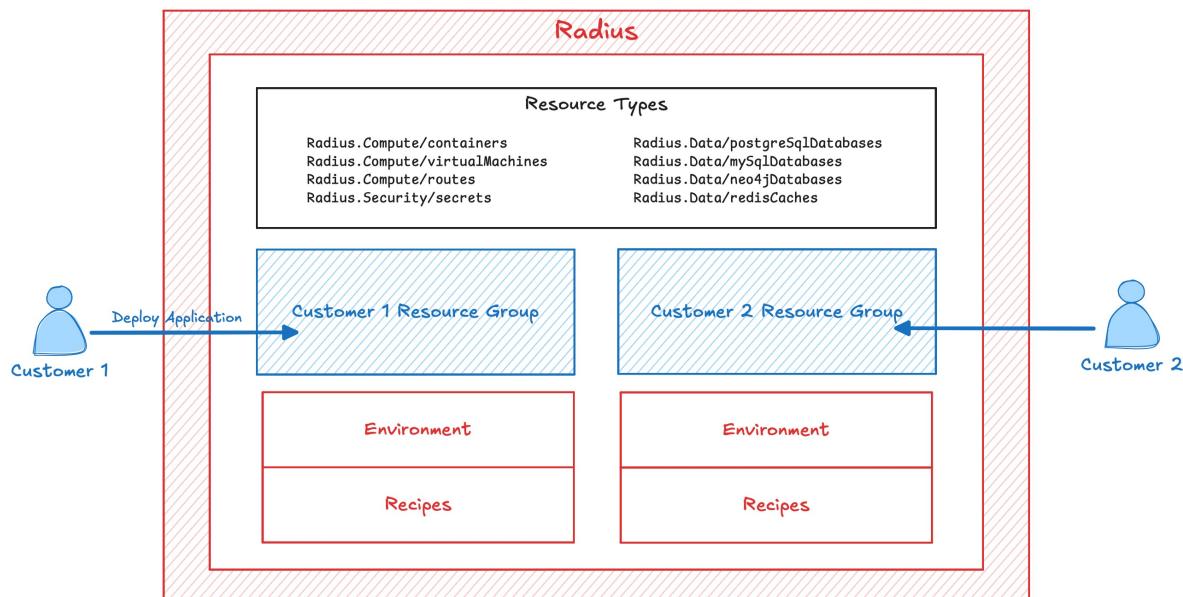
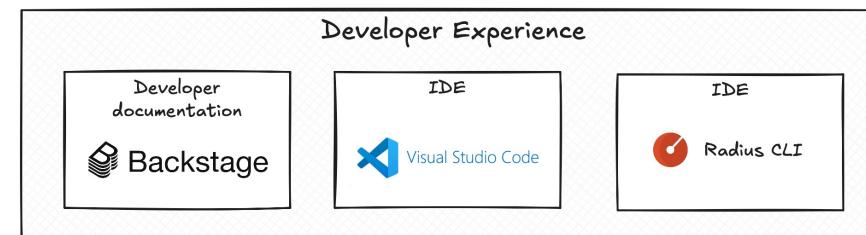


DEMO

Deploying PostgreSQL



Architecture



Get Started with Radius

The screenshot shows a web browser displaying the [radius docs](https://docs.radapp.io) website. The page title is "Tutorial: Create a Radius Application". The left sidebar contains navigation links for Home, Getting started, Installation, Tutorials (with a "Create a new application" section), How-to guides, Concepts, Reference, Contributing, and Community. The main content area shows a breadcrumb path "Tutorials / Create a new application". Below this is a heading "Tutorial: Create a Radius Application" and a sub-heading "Learn how to define, deploy, and manage a new Radius Application". A "Categories" section includes a "Tutorial" link. The "Overview" section describes the tutorial's purpose and steps: 1. Define and deploy a Radius Environment and Application, 2. Add a container to your application and customize that container, 3. Add a MongoDB database to your application and connect it to your container, 4. Add a second container and connect it to the first container, 5. Expose your application through a Gateway. It also states that by the end of the tutorial, the user will have created and deployed a new Radius Application. A "todolist Application" diagram is shown, illustrating the architecture with components like gateway, demo Container, mongoDB, and backend Container.

<https://docs.radapp.io/tutorials/new-app/>

The screenshot shows a web browser displaying the "Radius Roadmap" page. The top navigation bar includes links for Home, Blog, and GitHub. The main content area is titled "Radius Roadmap" and shows a grid of work items categorized by status: Proposed, Accepted, Designing, and Developing. Each item has a small circular icon indicating its status (blue for Proposed, green for Accepted, yellow for Designing, orange for Developing). The "Proposed" column contains items like "Remove extenders", "Policy management with Rego", and "End-to-end developer experience in Radius dashboard". The "Accepted" column contains items like "Dry-run deployments", "Radius usability", and "Radius Terraform provider". The "Designing" column contains items like "Customizable Terraform backend", "Deploy to multiple Kubernetes clusters", and "External data store for Radius". The "Developing" column contains items like "Compute platform extensibility", "Install Radius in an air-gapped environment", and "Advanced Terraform settings". A search bar at the top right allows users to search for specific items.

<https://aka.ms/radius-roadmap>





Stay up to date

Get involved on Discord
<https://aka.ms/Radius/Discord>

Follow the Radius blog
<https://blog.radapp.io/>

Get release notes and community call invites
https://groups.google.com/g/radapp_io

Follow Radius on social media

- in <https://www.linkedin.com/company/radius-project>
- X <https://x.com/radius-project>
- @ https://www.threads.net/@radapp_io
- https://www.youtube.com/@radapp_io



Q&A

