# Cloud Computing, PaaS, AWS

23/01/19

# Cloud Computing

Cloud computing is the *on-demand* delivery of

- Compute power
- Database storage
- Applications
- Other IT resources

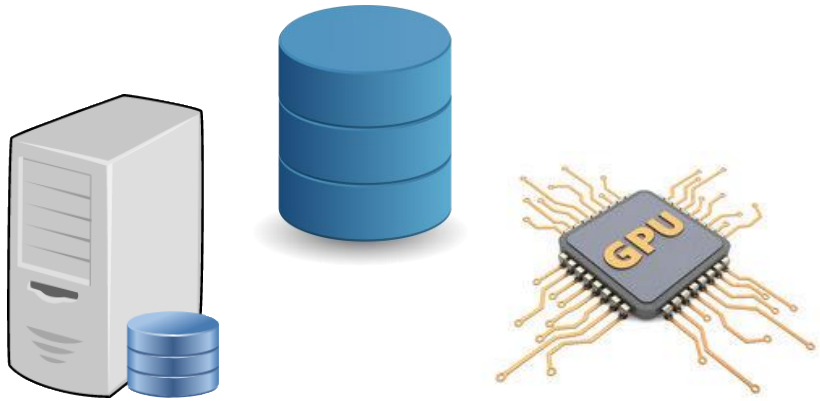
through a cloud services platform via the internet with pay-as-you-go pricing

# Cloud Computing

Cloud computing is the *on-demand* delivery of

- Compute power
- Database storage
- Applications
- Other IT resources

through a cloud services platform via the internet with pay-as-you-go pricing

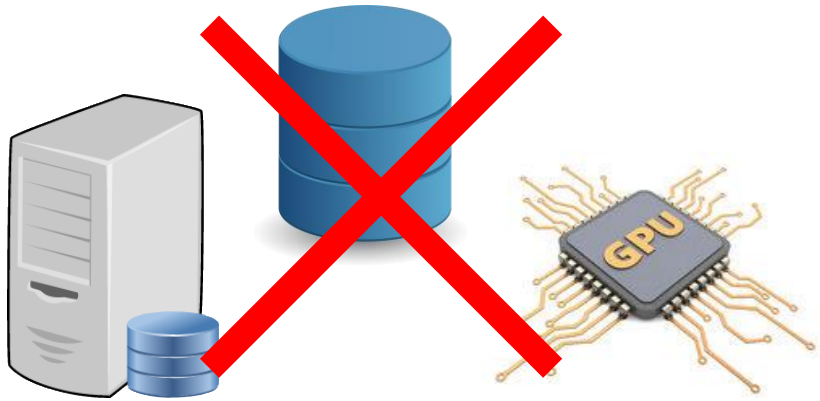


# Cloud Computing

Cloud computing is the *on-demand* delivery of

- Compute power
- Database storage
- Applications
- Other IT resources

through a cloud services platform via the internet with pay-as-you-go pricing

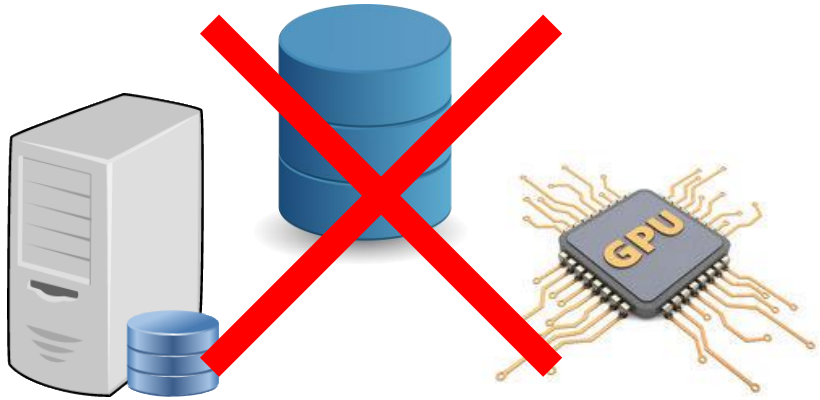


# Cloud Computing

Cloud computing is the *on-demand* delivery of

- Compute power
- Database storage
- Applications
- Other IT resources

through a cloud services platform via the internet with pay-as-you-go pricing




Azure

AWS

# AWS: Amazon Web Services

AWS is a service that offers many cloud-based products

# AWS: Amazon Web Services

AWS is a service that offers many cloud-based products

- Compute power
- Storage
- Database
- Analytics
- Networking
- Mobile tools
- Developer tools
- Security applications
- Management tools
- IoT
- Enterprise applications

# AWS: Amazon Web Services

AWS is a service that offers many cloud-based products

| | | |
|---|---|---|
| Compute power | Networking | Management tools |
| Storage | Mobile tools | IoT |
| Database | Developer tools | Enterprise applications |
| Analytics | Security applications | |

These services help organizations move faster, lower IT costs, and scale

# AWS: Amazon Web Services

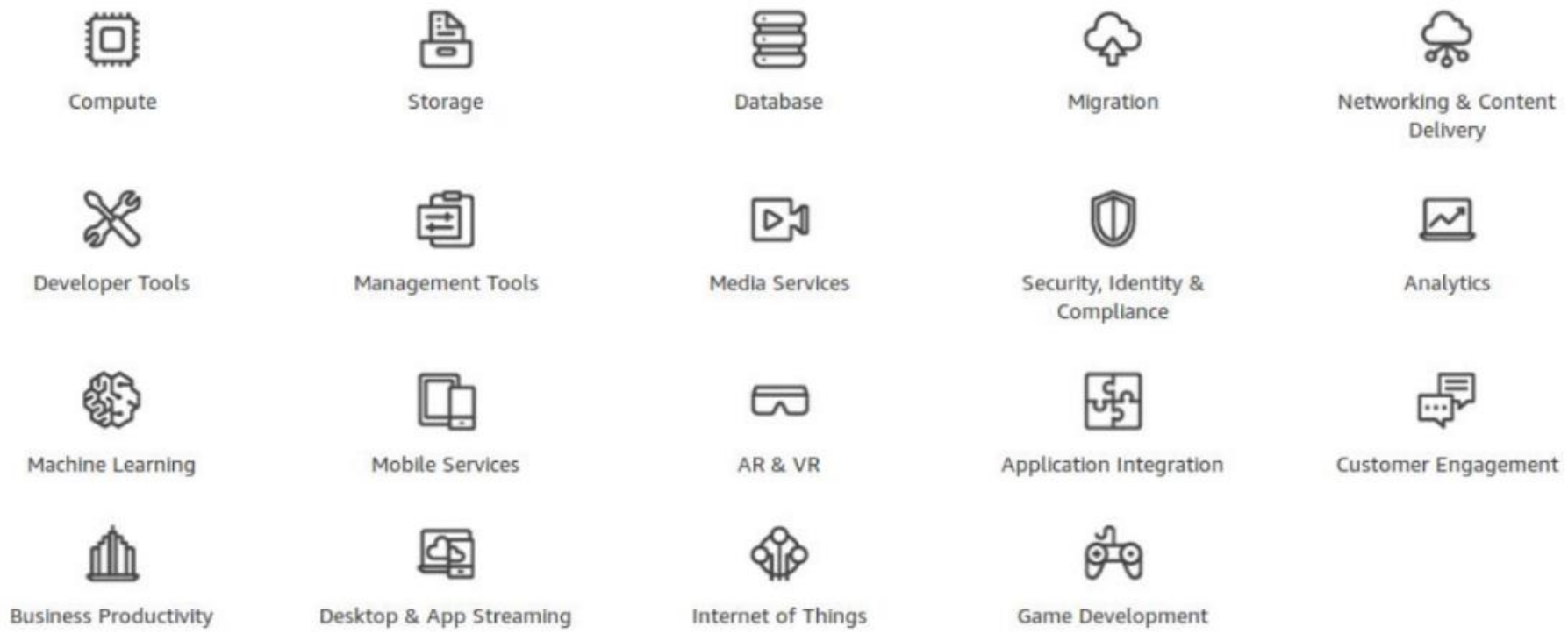
AWS is a service that offers many cloud-based products

| | | |
|---|---|---|
| Compute power | Networking | Management tools |
| Storage | Mobile tools | IoT |
| Database | Developer tools | Enterprise applications |
| Analytics | Security applications | |

These services help organizations move faster, lower IT costs, and scale

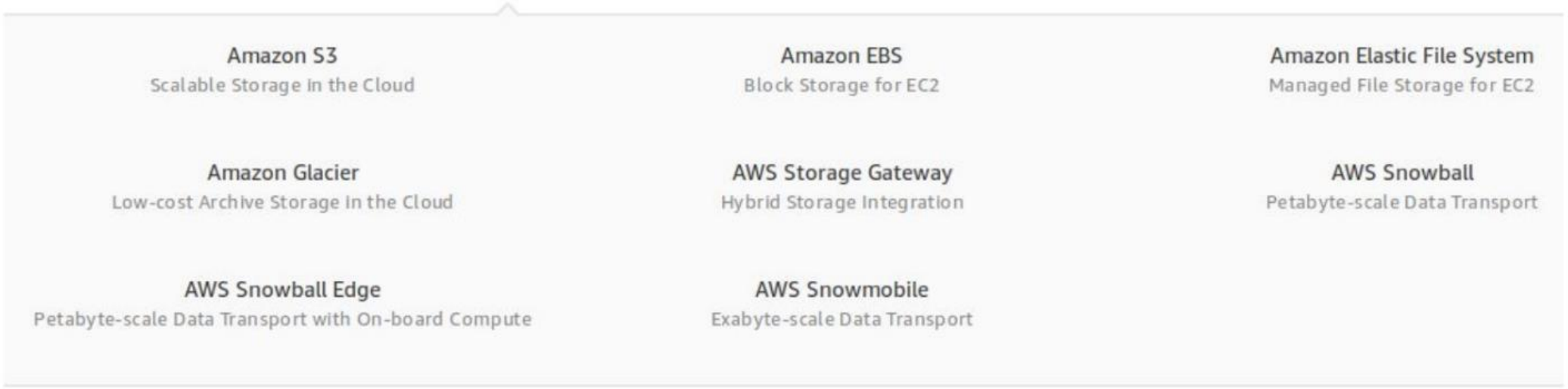
Used for web and mobile applications, game development, data processing, etc.

# AWS Services



# AWS Services: Compute

# AWS Services: Storage



# AWS Services: Security

**Amazon CloudWatch**
Monitor Resources and Applications

**AWS Auto Scaling**
Scale Multiple Resources to Meet Demand

**AWS CloudFormation**
Create and Manage Resources with Templates

**AWS CloudTrail**
Track User Activity and API Usage

**AWS Config**
Track Resource Inventory and Changes

**AWS OpsWorks**
Automate Operations with Chef and Puppet

**AWS Service Catalog**
Create and Use Standardized Products

**AWS Systems Manager**
Gain Operational Insights and Take Action

**AWS Trusted Advisor**
Optimize Performance and Security

**AWS Personal Health Dashboard**
Personalized View of AWS Service Health

# AWS Services

There are many more

# AWS Services in the ERASMUS Course

**EC2:** Virtual Servers in the Cloud. Manage instances (Linux, Windows)

**VPC:** Isolated Cloud Resources. Manage networking and security

**IAM:** Manage User Access and Encryption Keys

**CloudWatch:** Monitor Resources and Applications

**Cost Explorer:** manage the use and cost

**Billing:** manage bills

**CloudFormation:** Create and Manage Resources with Templates

# PaaS: Platform as a Service Introduction

When building an application, developers must

# PaaS: Platform as a Service Introduction

When building an application, developers must

Find data centers for data storage

# PaaS: Platform as a Service Introduction

When building an application, developers must

Find data centers for data storage

Purchase computing power

# PaaS: Platform as a Service Introduction

When building an application, developers must

Find data centers for data storage

Implement measures of security

Purchase computing power

# PaaS: Platform as a Service Introduction

When building an application, developers must

Find data centers for data storage

Implement measures of security

Obtain servers to host the application

Purchase computing power

# PaaS: Platform as a Service Introduction

When building an application, developers must

Find data centers for data storage

Implement measures of security

Obtain servers to host the application

Purchase computing power

And more

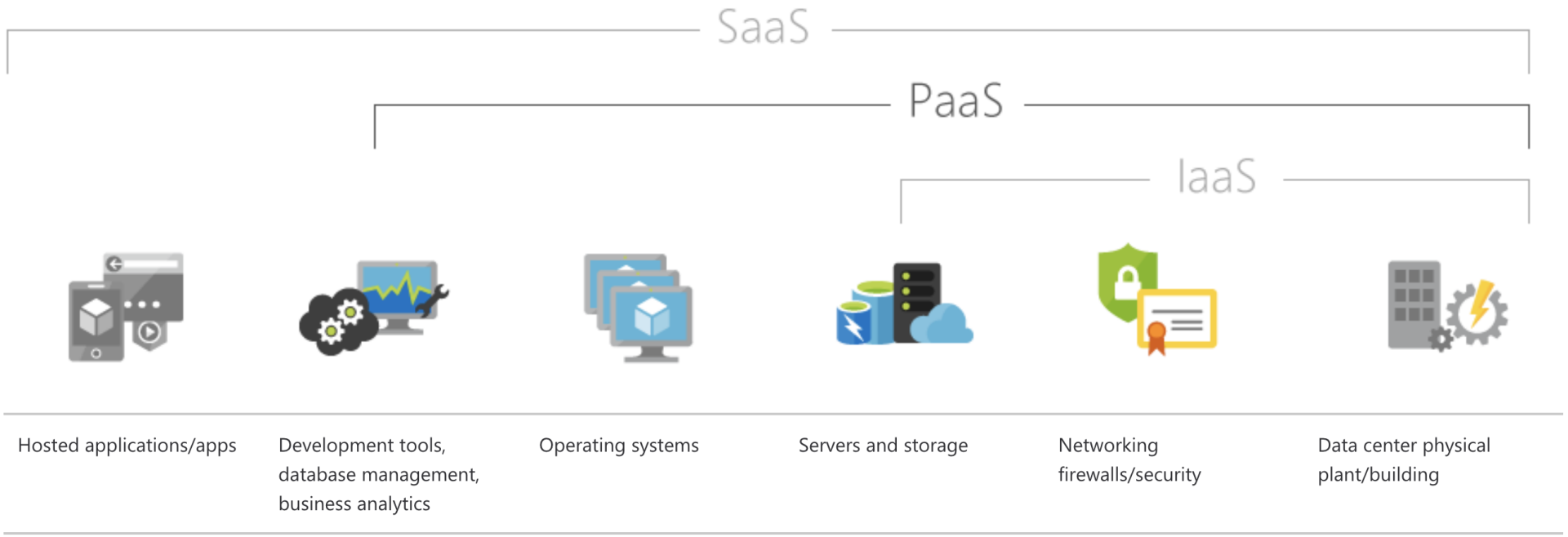
# What is PaaS?

# What is PaaS?

PaaS is a *cloud computing model* that allows the users to *develop, deploy, run and manage* applications without taking care about the underlying layers

# What is PaaS?

PaaS is a *cloud computing model* that allows the users to *develop, deploy, run and manage* applications without taking care about the underlying layers



# What is PaaS?


Traditionally, the developer handles everything

| | |
|---|---|
| Servers | Compute power |
| Storage | Network |
| Security | Etcetera |

# What is PaaS?


Traditionally, the developer handles everything

| | |
|---|---|
| Servers | Compute power |
| Storage | Network |
| Security | Etcetera |

With PaaS, an external provider handles the general services

# What is PaaS?


Traditionally, the developer handles everything

| | |
|---|---|
| Servers | Compute power |
| Databases | Network |
| Security | Etcetera |

With PaaS, an external provider handles the general services so that the developer can focus on the custom details specific to their application

# Advantages of PaaS

Developers only focused on **developing applications**

Developers **don't care about the underlying layer** (infrastructure), saving time and resources

Developers can **code applications more efficiently** by taking advantage of pre-coded components such as workflow, directory services, security features, search, etc.

Developers only pay for what they use under the **pay-as-you-go model**

Developers can **work together from remote locations** because the development environment is accessed over the Internet

# Examples of PaaS

AWS, Google App Engine, Microsoft Azure, Oracle Cloud Platform

Heroku, OpenShift

# Examples of PaaS

AWS, Google App Engine, Microsoft Azure, Oracle Cloud Platform

Heroku, OpenShift

Differences in autonomy / flexibility, pricing, and features

# OpenShift Introduction

Application are traditionally implemented as a **single set of libraries and configuration files**

They are **implemented in an operating system** (on physical or virtual servers) with a set of services running (web, database, ...)

**Disadvantage 1:** O.S. updates can interrupt the application.

**Disadvantage 2:** If several applications are running on the same system, the library updates of one of them can affect other applications.

**Possible solution: High availability systems that minimize the downtime of an application.**

Can be accomplished via **containers**

# Containers

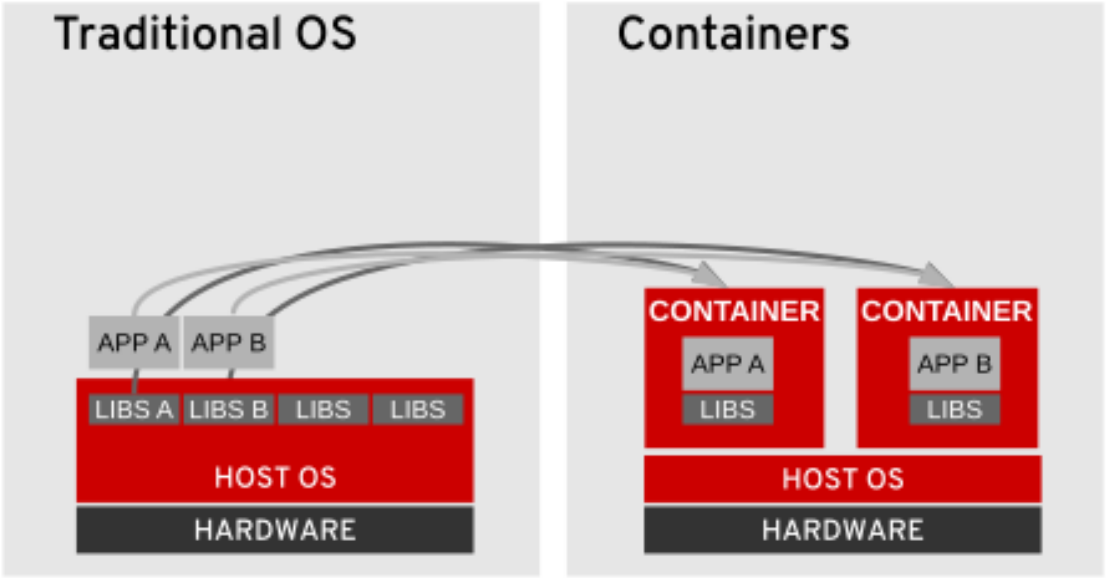
A container is a partition isolated within a single operating system

They consist of an entire runtime environment (application, dependencies, libraries, etc.)

# Containers

A container is a partition isolated within a single operating system

They consist of an entire runtime environment (application, dependencies, libraries, etc.)



# Benefits of Containers

Require fewer hardware resources than virtual machines, but offer similar benefits

Quicker to start and stop

Environment isolation

Efficient deployment

Components reused

The impact of errors/changes can be minimized

# OpenShift

Service for PaaS developed by Red Hat

Focus on application development

Uses Docker (CRI-O) and Kubernetes internally

Allows developers to deploy applications in different environments (dev, prod, etc.)

Facilitates continuous integration

Interact with OpenShift via web application, CLI or API REST

# Docker

Virtualization system that make it easier to create, deploy, and run applications by allowing developers to work with application containers

Containers are self-sufficient, so they don’t need to depend on other containers to function

Multiple containers share the same core, but each container can be restricted to using only a defined amount of resources such as CPU, memory and I/O

# Kubernetes

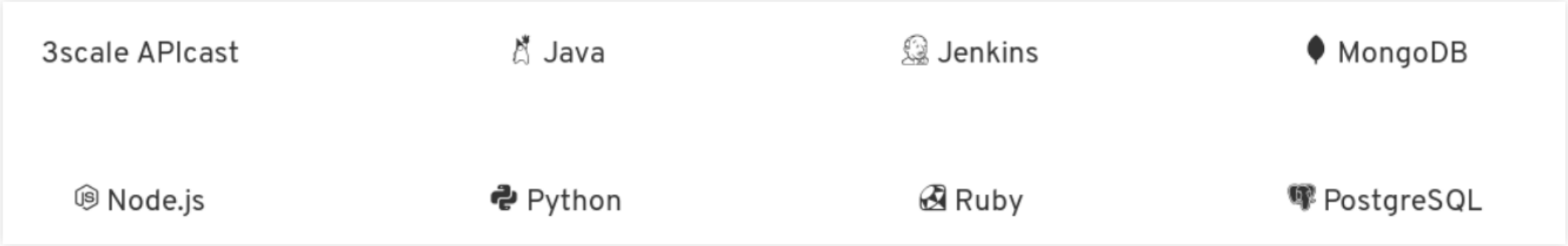
Open-source system that allows developers to automate deployment, scaling, and management of application containers

Manages a set of servers where the containers are executed

The real applications are formed by several containers

Offers developers the mechanisms to manage the containers: implementation, service assurance, error tolerance, scalability, continuous updates, integration and automatic deployments, load balancing, monitoring

# Available Technologies on OpenShift



# Example of PaaS

http://myscheduly.herokuapp.com/welcome

# Example of PaaS

| Me | Heroku |
| --- | --- |
| Application Code | Deployment |
| Application Features | Computing Services |
| Data | Data Storage |
| Images | Network |
| | Server |
| | Security |