Unit 3
Platform as a Service

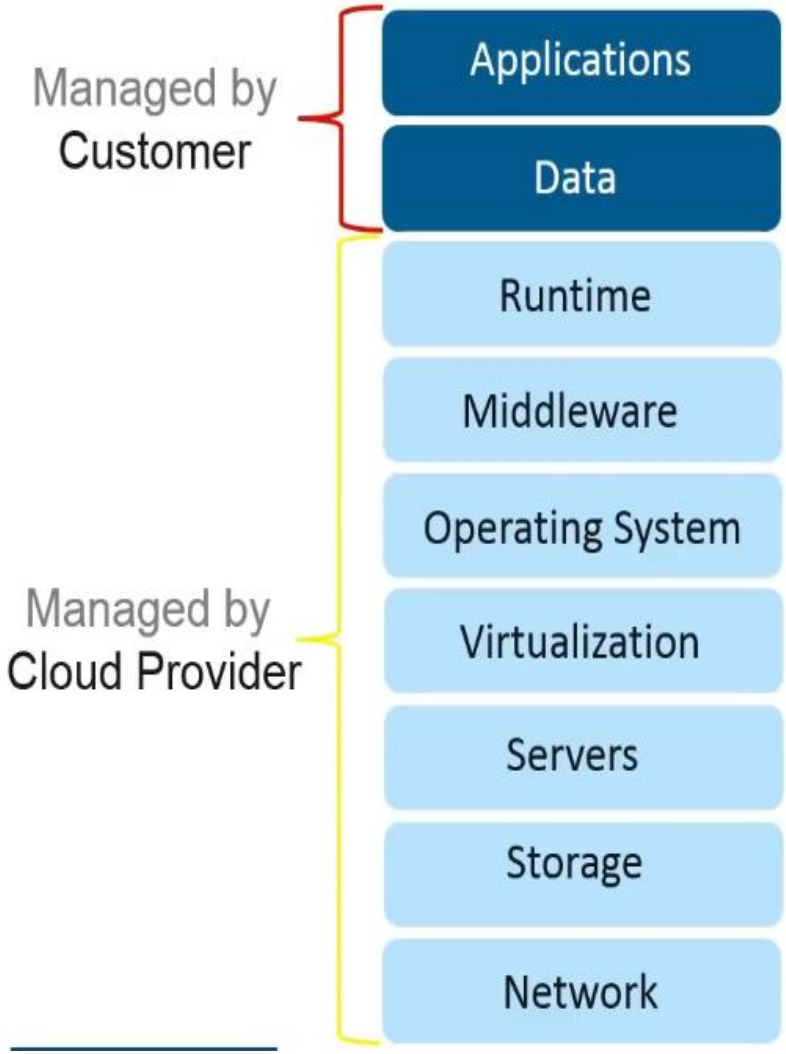
What is PaaS?

- Platform as a service (PaaS) is a cloud computing model in which a third-party provider delivers hardware and software tools , usually those needed for application development to users over the internet.
- A PaaS provider hosts the hardware and software on its own infrastructure.
- It is an abstracted and integrated cloud based computing environment that supports the development, running and management of applications
- PaaS is a category of cloud computing that provides a platform and environment to allow developers to build and deploy applications and services on the internet.
- An abstraction layer between your cloud application and your IaaS provider.

What is PaaS?

- PaaS services are hosted in the cloud and accessed by users simply via their web browser.
- Fundamentally provides elastic scaling of your application.
- You need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application.
- Platform as a Service (PaaS) is a way to rent hardware, operating systems, storage and network capacity over the Internet.
- The service delivery model allows the customer to rent virtualized servers and associated services for running existing applications or developing and testing new ones.

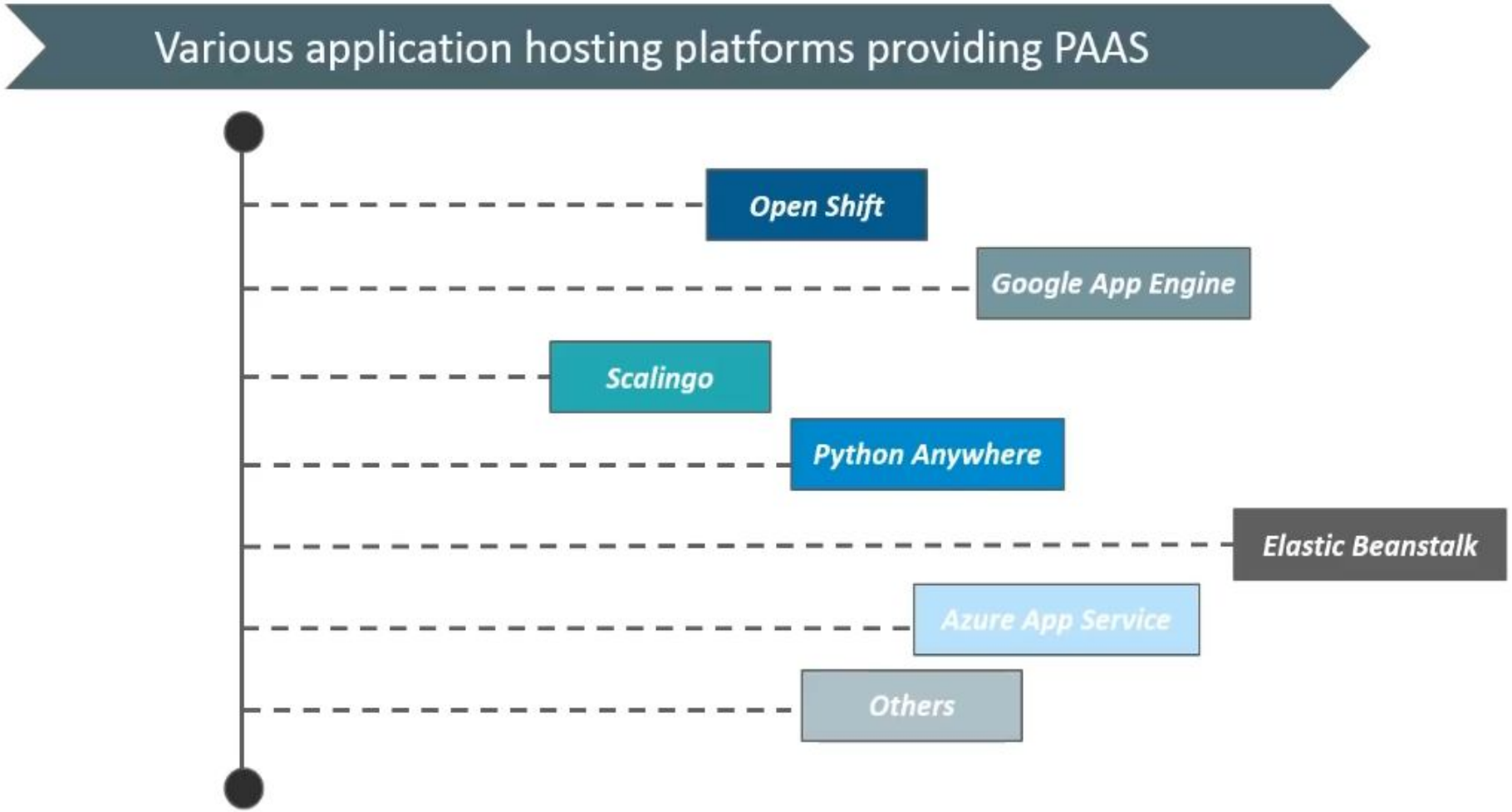
PAAS(Platform-As-A-Service)



Features for PaaS offering

- Operating system
- Server-side scripting environment
- Database management system
- Server Software
- Support
- Storage
- Network access
- Tools for design and development
- Hosting

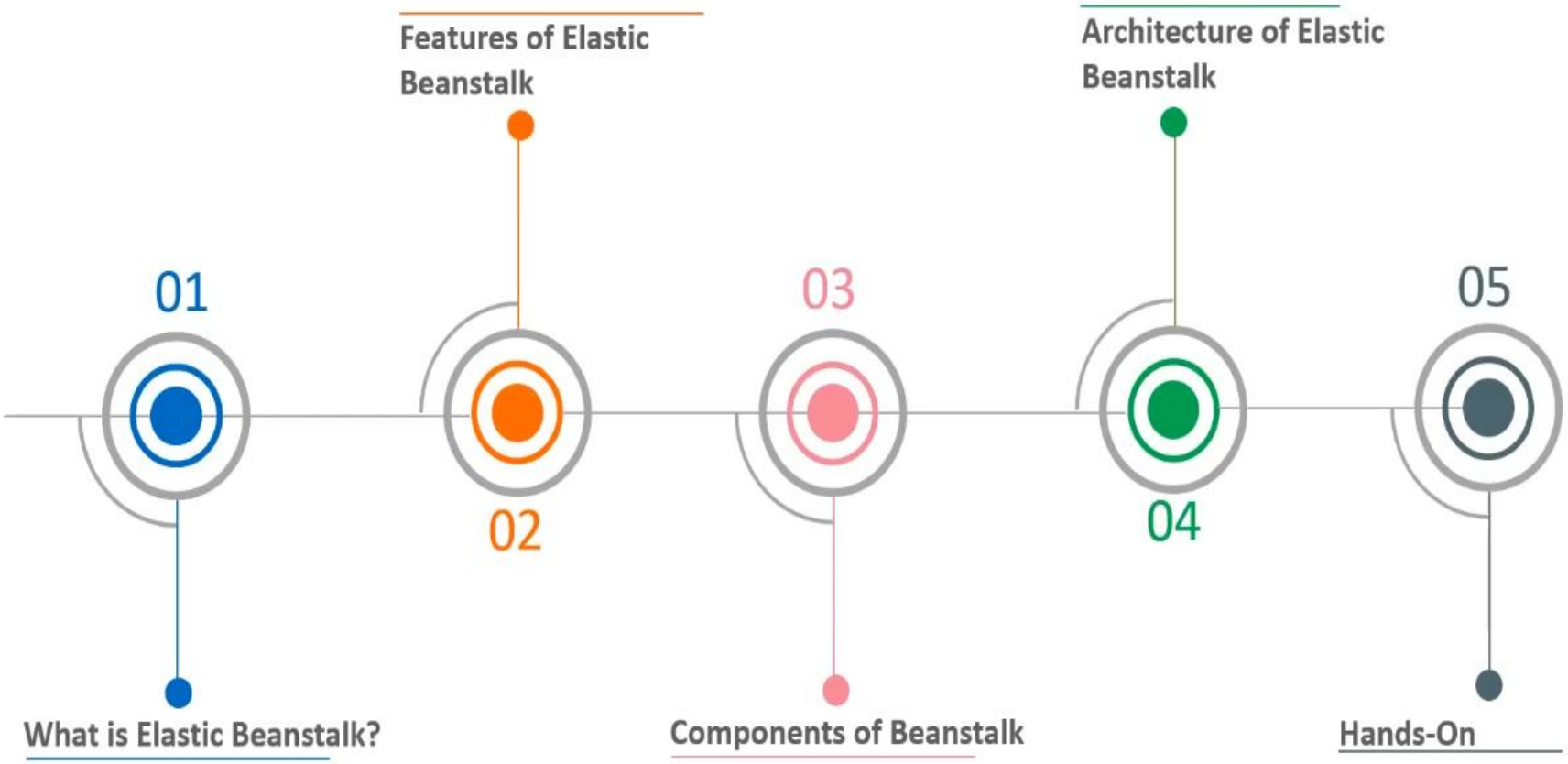
Web Hosting Platforms



Elastic Beanstalk

Tamal Dey
MCA,PESU

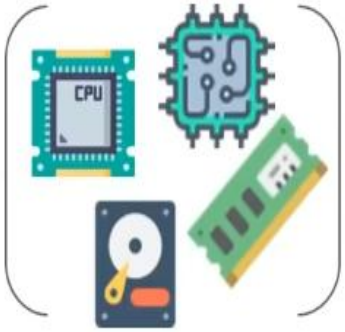
Outline



AWS Elastic Beanstalk is an PAAS service used for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js etc on familiar servers such as Apache, Nginx, Tomcat, and IIS.

Scenario: A Computer

Option A



You can go to a computer ware house and buy different components according to your requirement & assemble them.

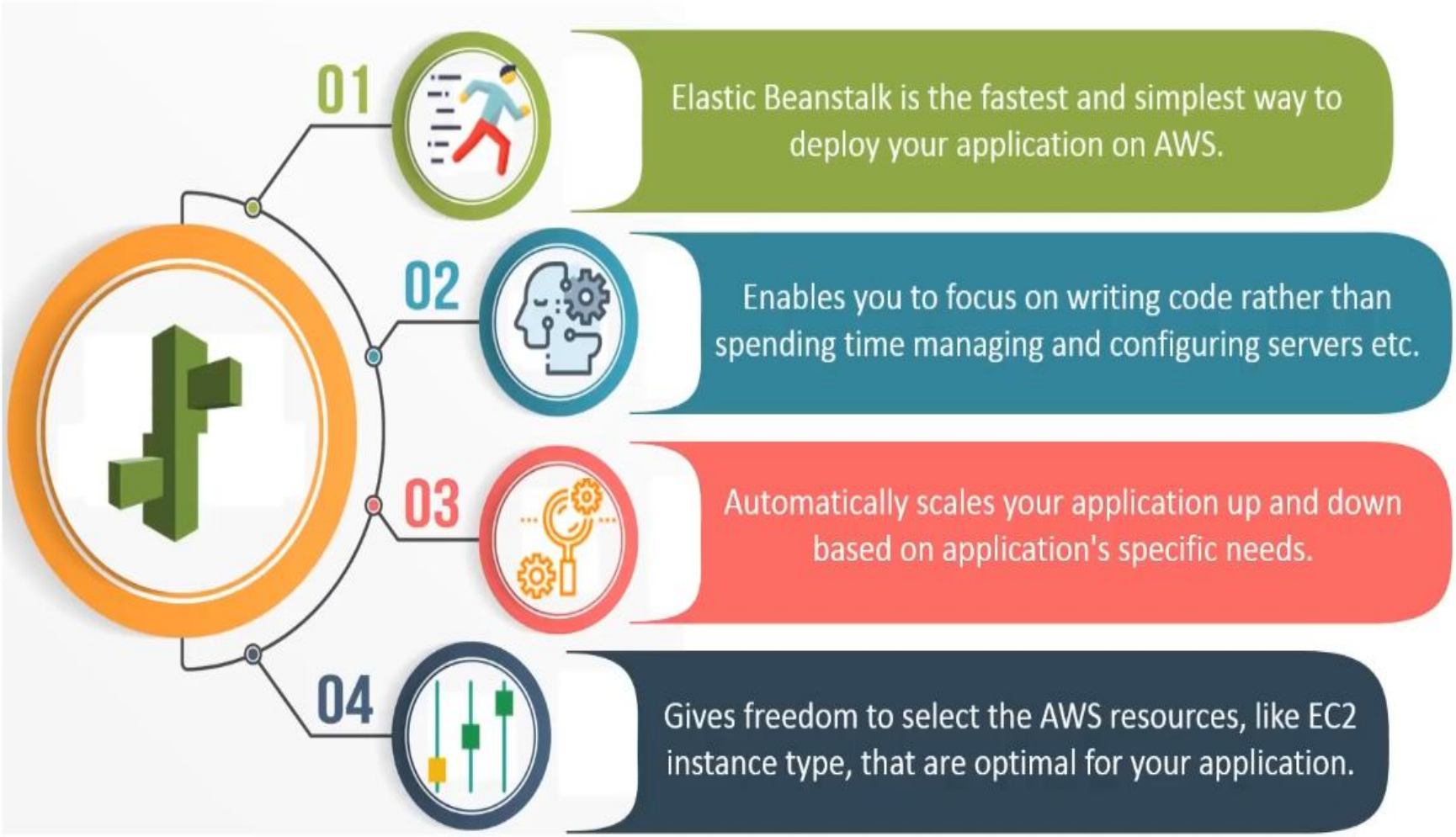
Deploying an application without using Elastic Beanstalk

Option B

You can visit a electronic retail store and buy a computer that fits your requirements.

Deploying an application using Elastic Beanstalk

Features of AWS Elastic Beanstalk



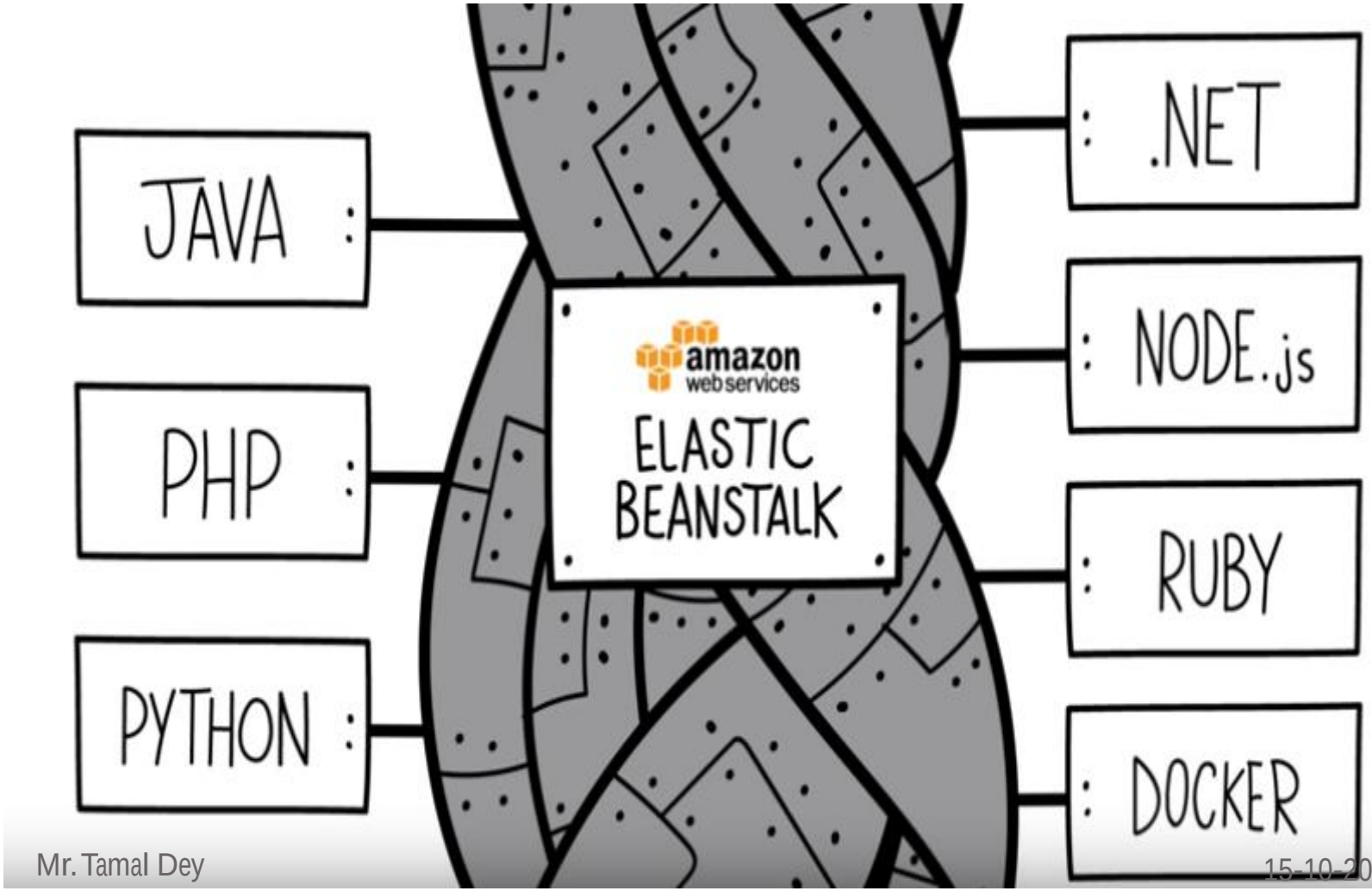
Beanstalk

- AWS provides PaaS solution called Elastic Beanstalk. It is mature PaaS than other PaaS service provider.
- Developers no need to worry about infrastructure to launch an web app. Developers just upload the application as binary file such as ***projectname.war***
- Developers can use application versioning to switch between previous version of application by using ***swap*** URL environment option.
- It is possible to use an existing instance instead of provisioning a new environment. You can create AMI of your existing instance and use your AMI to create an instance.

Beanstalk

- Whenever we want to host an application, we have configure system, manage servers, databases, scaling, load balancing etc.
- Most of the time the programmers have to concentrate on these things
- Beanstalk helps in reducing all these activities and helps to host/deploy your application very fast
- Beanstalk is a service that deploys, manages and scales the web application for the users

Manage Applications



Manage Servers (Apache http, Apache tomcat, NginX, IIS)

Components Of Elastic Beanstalk

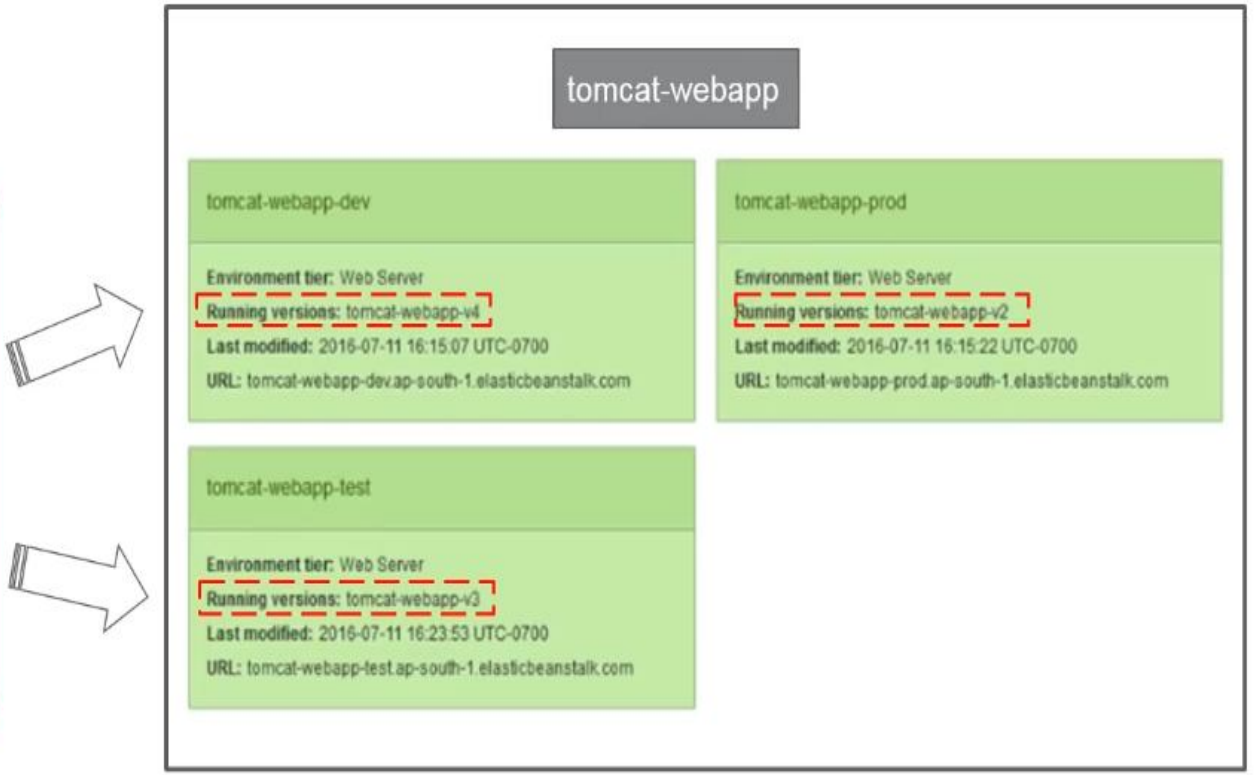
- An application is a collection of components including ***environments, versions*** and ***environment configuration.***
- An application in Elastic Beanstalk is conceptually similar to a folder.
- Ex : **tomcat-webapp**

01 Application

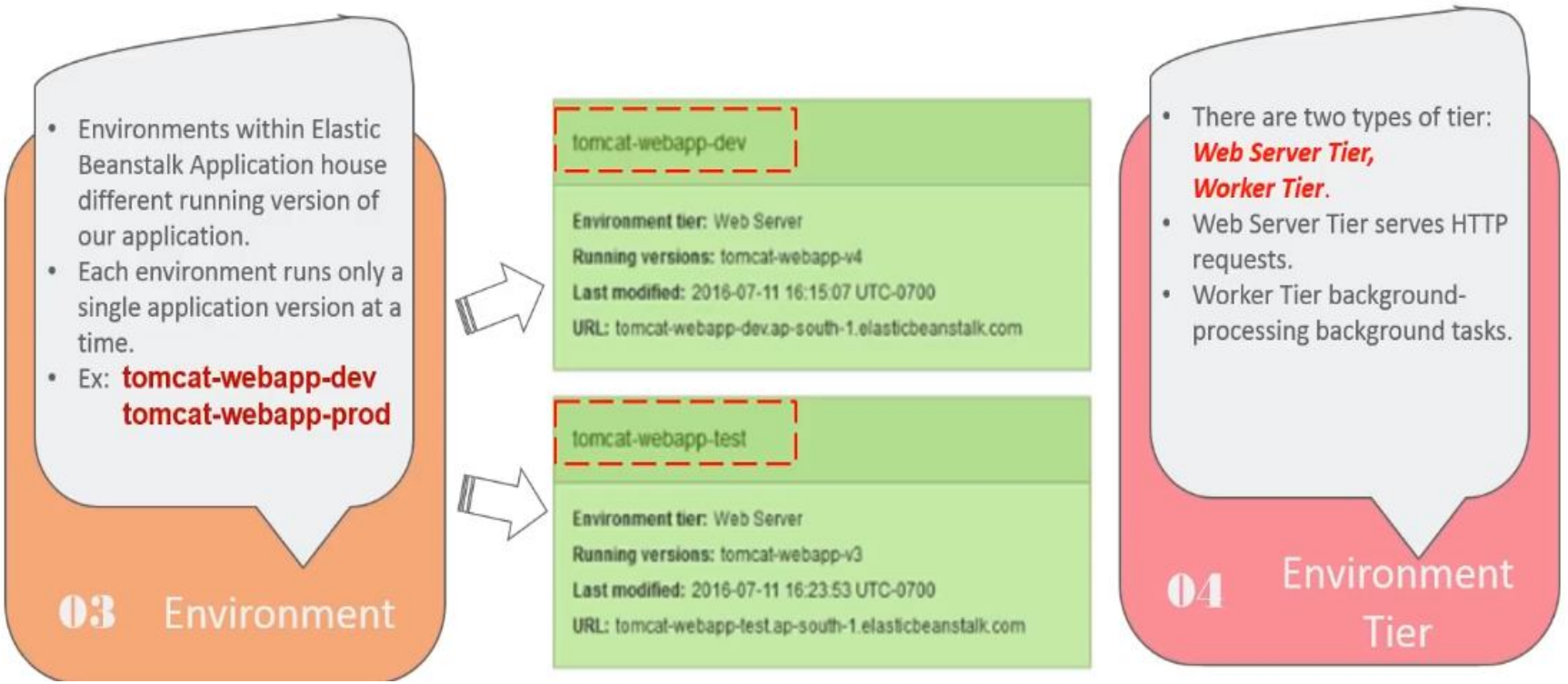
Components Of Elastic Beanstalk

- An *application version* refers to a specific, labeled iteration of deployable code for a web application.
- An application version points to an Amazon S3 object that contains the deployable code such as a Java WAR file.

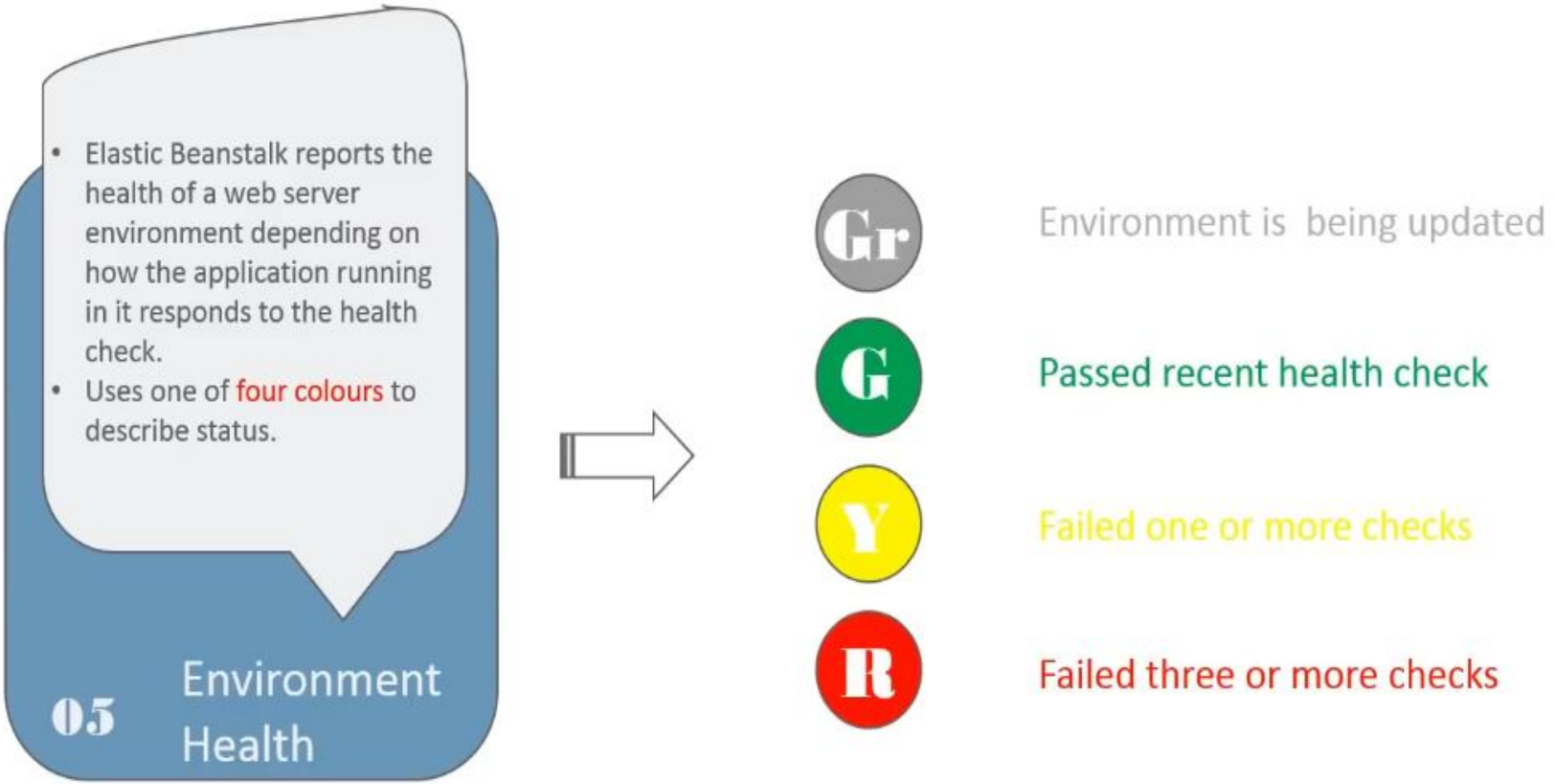
02 Application Version



Components Of Elastic Beanstalk

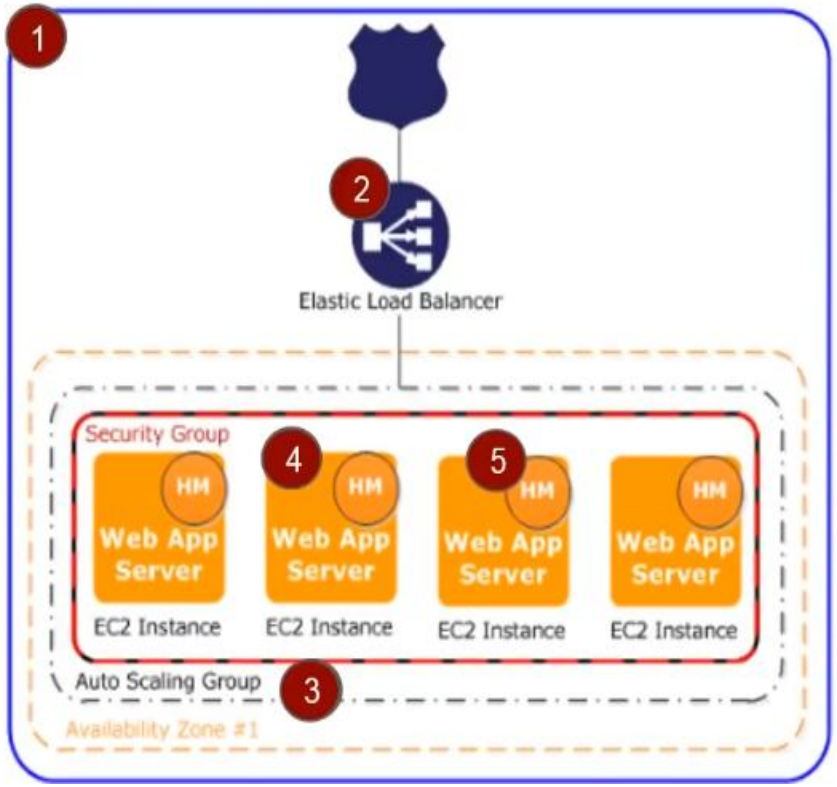


Components Of Elastic Beanstalk



Web Server Environment

Web Server Environment Tier handles HTTP requests.



01 Beanstalk Environment

02 Elastic Load Balancer

03 Auto Scaling Group

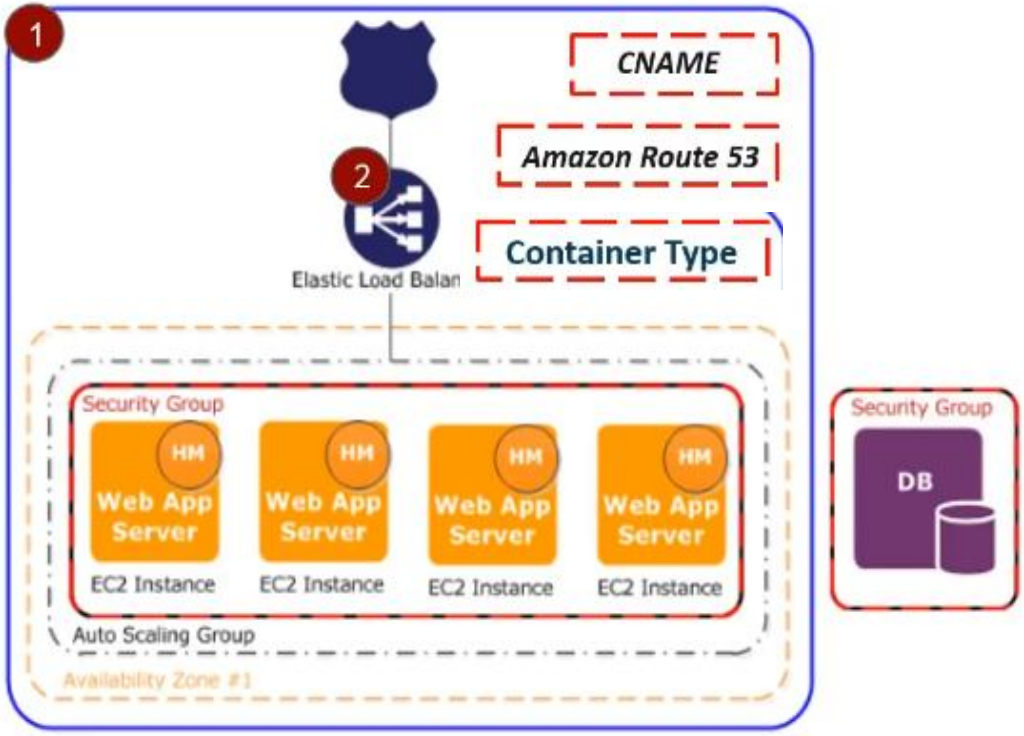
04 EC2 Instances

05 Host Manager

06 Security Groups

Web Server Environment

Web Server Environment Tier handles HTTP requests.



01 Beanstalk Environment

02 Elastic Load Balancer

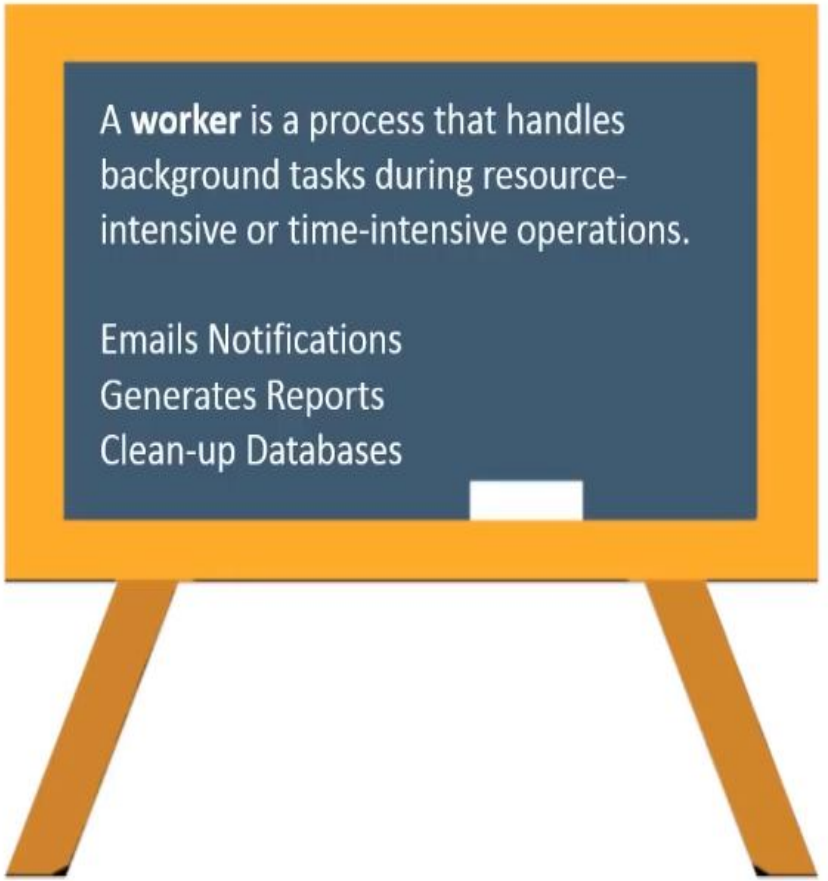
03 Auto Scaling Group

04 EC2 Instances

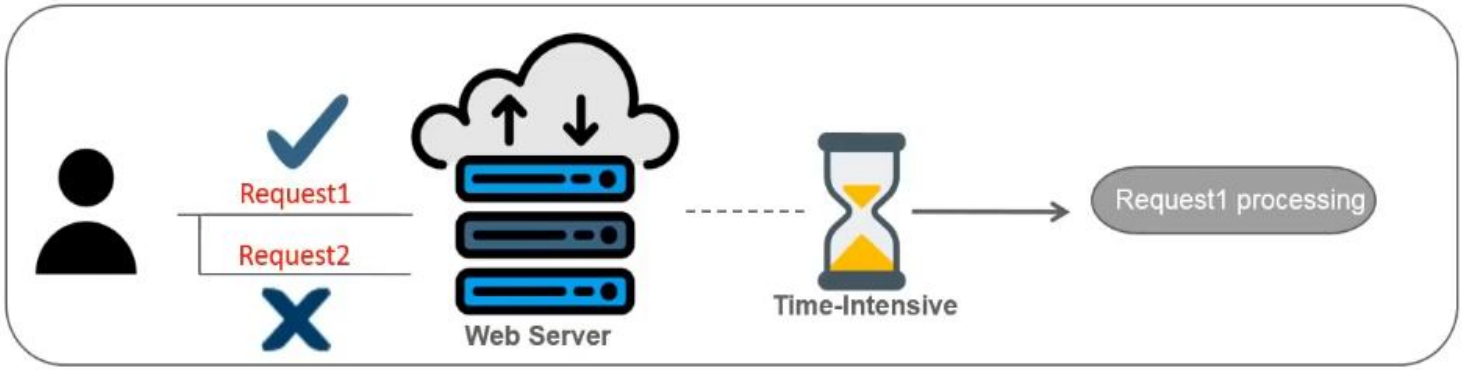
05 Host Manager

06 Security Groups

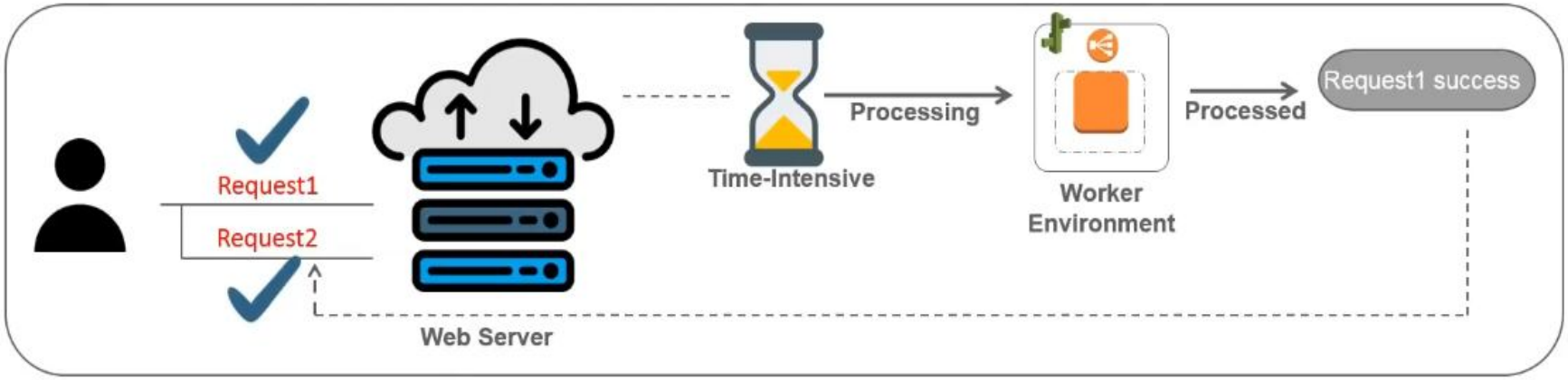
Worker Environment



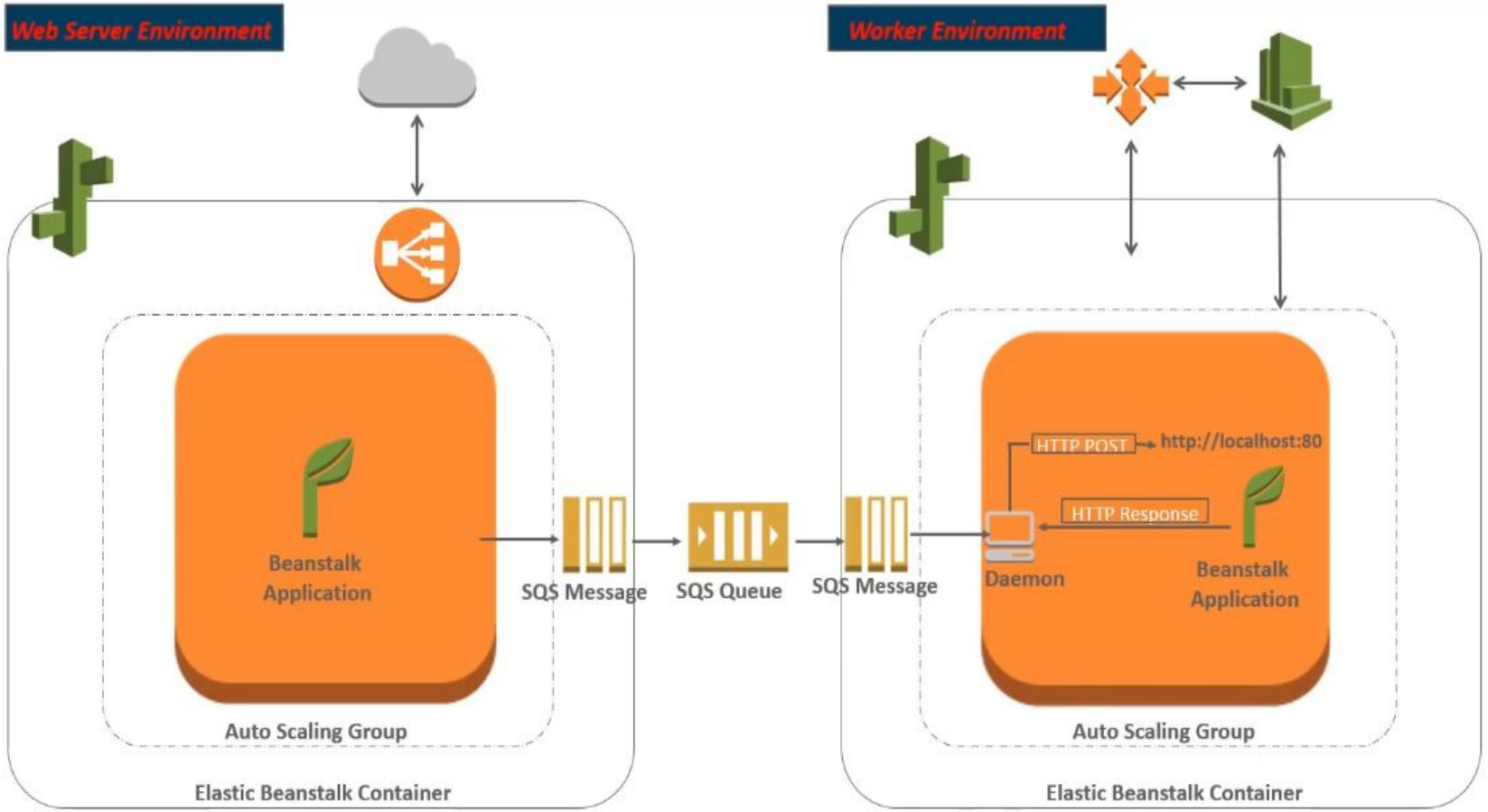
Why Do We Need Worker?



Performance and Requests Accepted

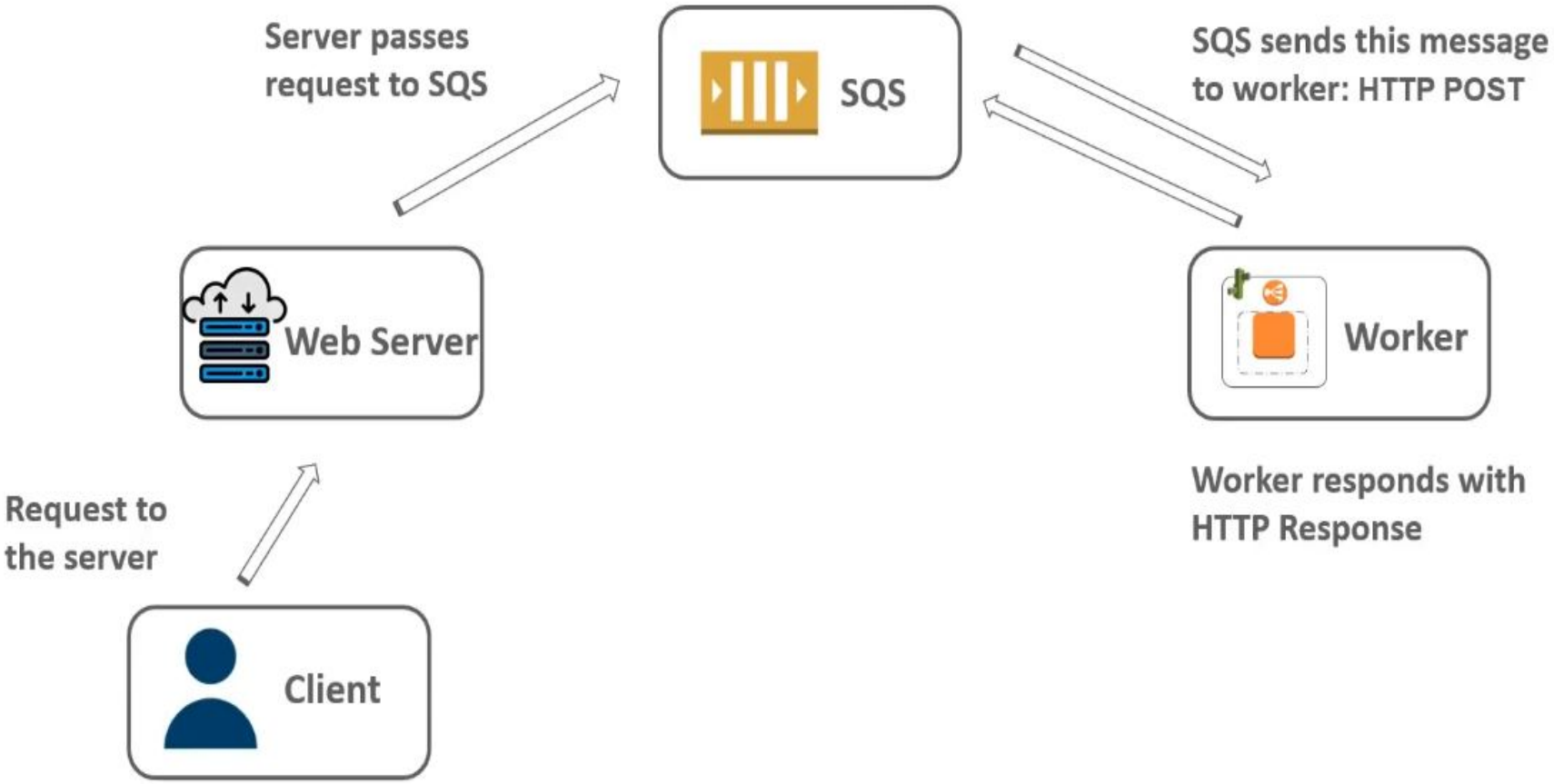


How Do These Two Environments Communicate?

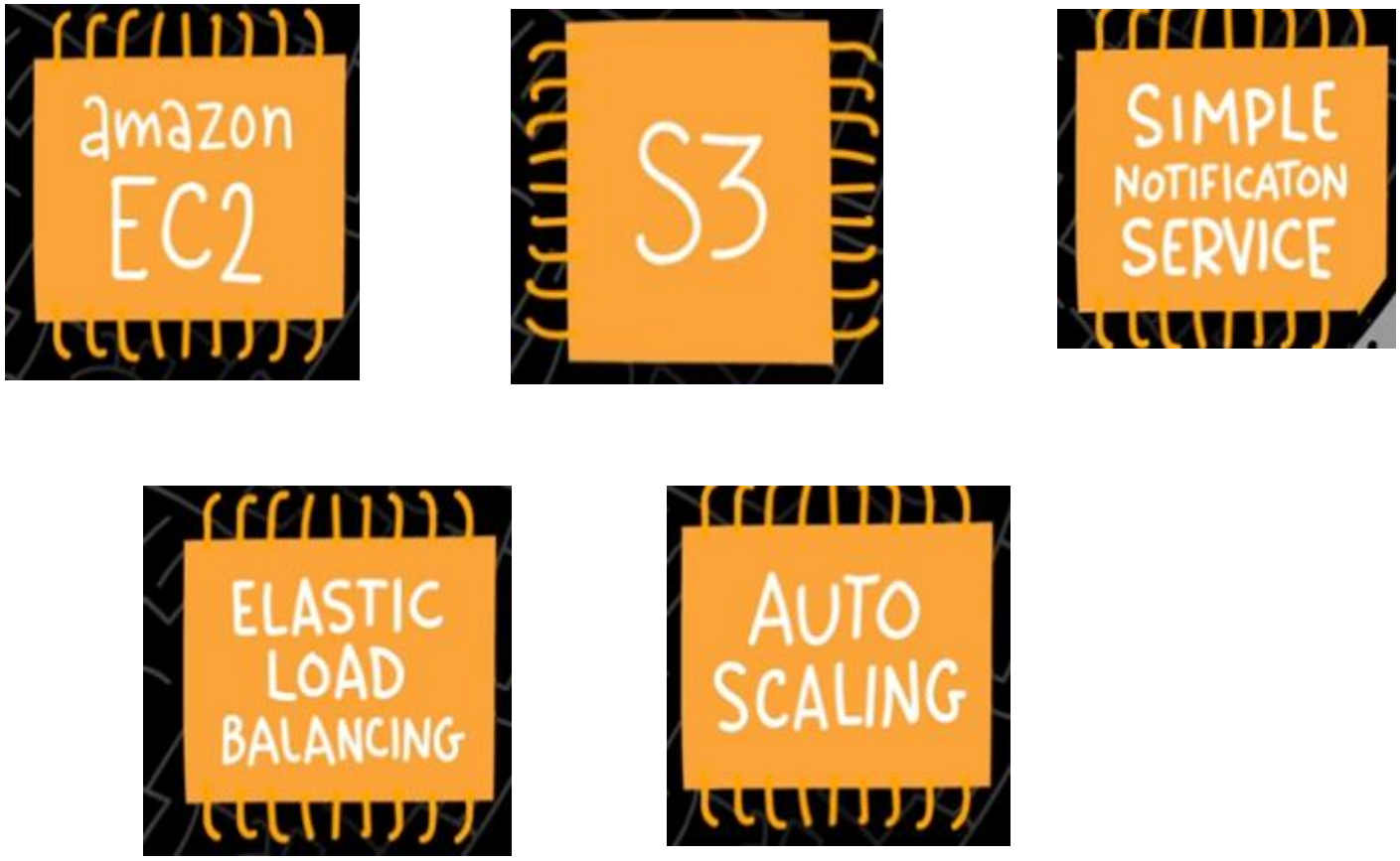


- Amazon *Simple Queue Service* (**SQS**) is a **message** queuing service that enables FIFO queues that are designed to guarantee that **messages** are processed exactly once that they are sent.

How Do These Two Environments Communicate?



Communication to other AWS services



Demo-How To Deploy An Application Using Beanstalk

Steps to Deploy Application

- Go to AWS management console
- Under Compute choose Elastic Beanstalk
- Click to
  - Give the application name
- Configure the environment
  - Choose env. tier : web server
  - Platform: PHP
- Application Version
  - Sample application / Your application code

- Environment information
  - Give the domain name
- Additional resources
  - Create an RDS instance
- Configuration details
  - T2.micro
  - Select your ec2 key pair
- Click save and launch
- Wait for some time to get activated
- Click on the application url on the page.
- Add your application foldername/index.php in the url

- **For Static website**

  Service -> Elastic beanstalk -> Get Started -> Give application name -> Choose platform as your file type -> select upload your code -> In local file choose file -> upload -> create application -> Click on URL

- **For dynamic website**

  Service -> Elastic beanstalk -> Get Started -> Give application name -> Choose platform as your file type -> select upload your code -> In local file choose file -> upload -> Configure more option -> in Instances select Modify -> Under EC2 security groups select default -> Save -> In security -> Select key-pair -> save -> In Databases -> Give user Name and password -> Save -> Create application -> Click on URL

- Connect to database using workbench.

Summary of Steps

- Log into the AWS console and go to the compute section and select Elastic beanstalk
- Click on the Get started button
- Next you need to choose the type of Application code that needs to be deployed.
- Choose the sample application and click on "Create Application" or Choose for upload your code and upload your zipped folder.
- As the application gets created, you will be able to see the different events as the infrastructure gets created
- You should be able to see the sample URL located at the right hand of the screen. Click on the URL to view the application.

- If you make any changes to the file, save the file and then right click in Windows explorer and choose on Send to->Compressed(zipped) folder
- Now go to Elastic beanstalk and click on Upload and Deploy
- The changes will start getting deployed
- And now to the URL and see the new changes to your application.

Complete Project Hosting

- Open Elastic Beanstalk
- Create new Application
- Sample name and choose a DNS name
- Choose PHP application
- Upload Web Hosting Project
- Go to Configuration and choose PHP
- Upload the project (webhosting.zip)
- Once created go to configuration add a ec2 key in Security.
  - Create the key before hand
- Go to database and create a user name and password
  - Wait for 10 Minutes
  - Go to enpoint link created under database and view copy the endpoint link for future access.

RDS Configuration

- Click the **security group** link in RDS and Edit the inbound rule
  - Add SSH and Check TCP(Default)
    - Modify the access for both protocols -> Connect Anywhere
- Connect to EC2 instance via the keypair (Putty/SSH)
  - cd /var/app/current
  - sudo chmod 777 –R webhosting
  - cd webhosting
  - vi config.php
    - DB Server- 'RDS Endpoint Address'
    - DB Username- '____'
    - DB Password- '____'
    - DB name- 'demo'
  - Save the file Escape key + :wq

DB Creation

- Connect to database from your EC2 Instance
  - Mysql –h ‘RDS enpoint address’ –u root –p
  - Create and use database demo
  - Create user table (refer query.sql file in webhosting directory)

- Go to browser- BeanStack URL/webhosting/login.php

- You are good to go
- In case of any problem refer the RDS setup video in last slide

Video References

- **AWS Elastic Beanstalk**
  - https://www.youtube.com/watch?v=96DJ2Og90hU
- **AWS Elastic Beanstalk - Part 1**
  - https://www.youtube.com/watch?v=IUZnlCDQilY
- **Connect AWS Elastic Beanstalk Using SSH- Part 2**
  - https://www.youtube.com/watch?v=jqxwCppc97s
- **Configure RDS**
  - https://www.youtube.com/watch?v=KbdWjETGoew&t=44s
  - https://www.youtube.com/watch?v=NheohB3adV8
  - https://www.youtube.com/watch?v=I-xaBOISR2s&t=2s