**AWS Cloud Practitioner Full Course on Edureka**

**Written by M. Jatin Kishore**

**Agenda :**

1. Introduction to cloud computing

2. How to become a AWS cloud practitioner

3.What is cloud computing

4. AWS basics

5. Instances in AWS

6.AWS Lambda

7.AWS Elastic Beanstalk

8.Cloud Storage on AWS

9.AWS S3

10.Networking Fundamentals

11.AWS CloudFront

12.Amazon CloudWatch

13.AWS CloudFormation

14.Autoscaling & Load Balancer

15.Cloud Security

16.AWS IAM

17.Amazon Redshift

18. AWS Kinesis, Gateway & SES Services

19.AWS IoT , Amazon Polly & AWS Recognition

20.DevOps on AWS

(i) AWS Code Pipeline (ii) Kubernetes on AWS

**1. Introduction to cloud computing :**

Cloud computing is an important skill in IT sector.

Every IT sector runs in cloud . So its an essential skill to be mastered.

**Who is this certification for?**

It’s a foundational knowledge ,a light weight of solution architect. In special , Its has a strong focus on billing on AWS.

So its often preferred by :

* Sales & Management Professionals
* Professional who want basic knowledge in AWS
* People who need to learn business-centric concepts & billing in AWS

**Why should you get in Certification?**

* Its one of the easiest AWS certification to get & it adds to your certifications.
* It gets you familiar with your test center and its environment
* It makes the solution architect exam easy to attend.

**How long do you study for this certification?**

* **Developer -** 8hrs
* **Bootcamp Graduate –** 15hrs
* **Sales and Management Professionals –** 20hrs

**Exam Details:**

* **Cost of Exam –** $100
* **Exam Duration –** 90min
* **No. of Questions :** 65
* **Passing Score :** 70%
* **Validity of Certificate :** 3 years

**Why Become a An AWS Cloud Practioner?**

**Job Opportunities:**

* India – 12,470+

Bangalore - 400+

* USA – 15,200+

California – 1000+

**Salary:**

**India –** ₹11,02,000

**USA –** $ 110k

**What is an AWS Cloud Practitioner?**

A certification for individuals who are looking to build and validate overall understanding of the AWS Cloud.

Organization are hiring them for :

* Designing
* Planning
* Managing
* Maintaining
* Supporting

**Roles and Responsibilities of the Cloud Practitioner:**

* Supervise the architecture and deployment of applications
* Adoption, management and monitoring of cloud platforms
* Designing and Building applications on the Cloud
* Executing Unix commands and perform containerization with Docker
* Working on AWS environment for applications building and deployment
* Handling complex project management tools like Jira.

**Cloud Practitioner Skills (**as per Tech Mahindra**):**

* Handling multiple AWS accounts
* AWS cloud platform design and hybrid performance solutions
* Design and implementation of workload in AWS in public cloud
* AWS security
* Microservices architecture
* DevOps tools
* Serverless application
* Participated/handed RFP and RFI’s

**Common Skills Required are as follows:**

* Cloud Infrastructure
* Operating System – Linux
* Virtualization – Docker
* Programming Language – Java, Python
* Storage and Networking
* Database
* Testing and Automation – Selenium
* Strategic Planning
* DevOps tools – Jenkins , Cradle etc.

**INTRODUCTION TO CLOUD COMPUTING:**

**What is Cloud Computing?**

cloud computing is the delivery of computing services—including servers, storage, databases, networking, software, analytics, and intelligence—over the Internet (“the cloud”) to offer faster innovation, flexible resources, and economies of scale.

**Before Cloud Computing:**

* Buy a stack of servers
* Traffic is not constant.
* Monitoring and maintaining the servers is difficult.

These troubles are overcame using cloud computing.

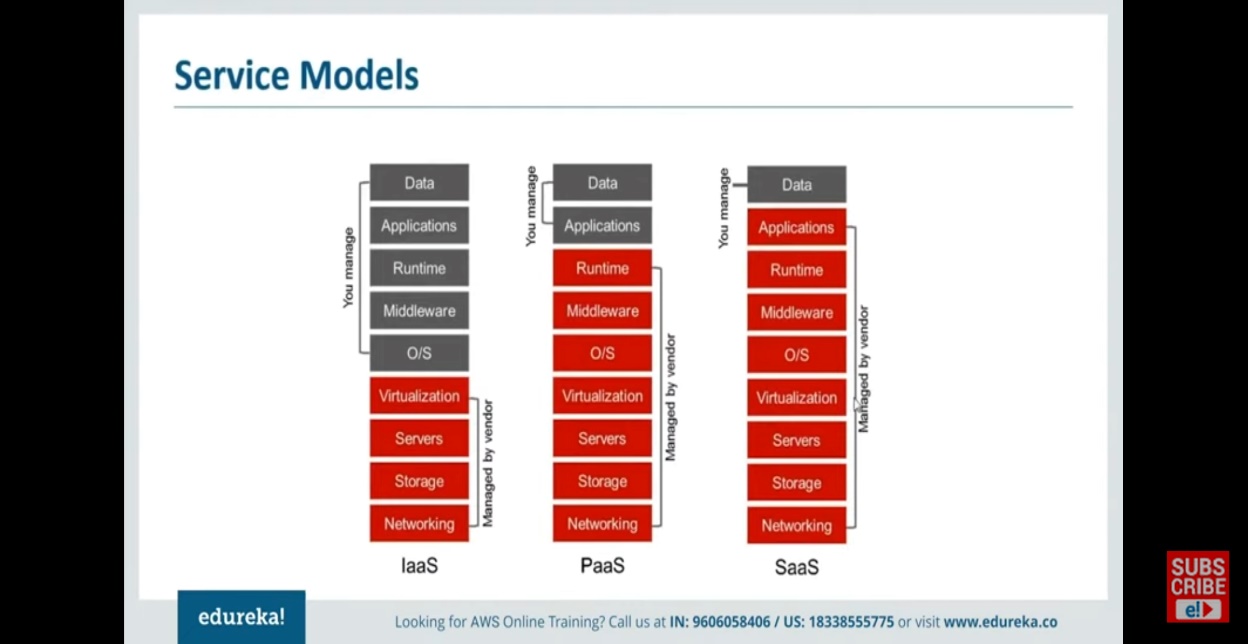
**Cloud Computing:**

Local computer with limited storage -> moving all the data -> cloud with unlimited storage

* Storing data/applications in remote servers
* Processing data/ application from servers
* Accessing data/application via internet

**Service Models in Cloud:**

* + 1. **SaaS:**
* Software as a service
* Consuming a service already made by someone
* We need not need to maintain it.
* Eg. gmail
  + 1. **PaaS:**
* Platform as a Service
* We can create our own application and run our application there. Where other users can use our application.
* But no control over the underlying architecture such as OS, Storage, Services etc.
* Eg. Google Engine
  + 1. **IaaS:**
* The whole infrastructure is given. So, we can create our own application with our required OS, storage and so on.
* Provides virtualized computing resources over the internet.
* No worries about the underlying physical machine

****

**DEPLOYMENT MODELS:**

* **Public Cloud**
  + A service provider who makes resources available for everyone in public via internet
  + Easy and inexpensive set-up because hardware, application and bandwidth costs are covered by the provider.
  + No wasted resources ,also you pay for what you use.
* **Private Cloud:**
  + Offers service to a limited number of people behind firewall to minimize the security concerns.
  + Private cloud give companies a direct control over their data.
* **Hybrid Cloud:**
  + It’s a mix of public, private and third-party cloud services
  + Helps to utilize the best of both the worlds.

**CLOUD PROVIDERS:**

* Among all the cloud providers AWS(Amazon) provides alone 6 times more computing than all other cloud providers. The second one is Azure (Microsoft) followed by Google Cloud Platform (cheapest among the market since they use less power consumption than any other using their analytics technology).

**INTRODUCTION TO AWS:**

**What is AWS?**

Amazon Web Services, Inc. is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered, pay-as-you-go basis. Clients will often use this in combination with autoscaling.

**Use Case:**

* Manufacturing Organization
* Architecture Consulting
* Media Company
* Large Enterprise.

**Advantages of AWS:**

* Flexibility
* Cost Effective
* Scalability
* Security

**AWS Architecture**

The infrastructure is divided in to 2 categories

* 1. **Availability Zones:** 
     + - Availability Zones are distinct locations within an AWS Region that are engineered to be isolated from failures in other Availability Zones. They provide inexpensive, low-latency network connectivity to other Availability Zones in the same AWS Region.
  2. **Regions:**
     + - A region consists of multiple availability zones.
       - There are several regions across the world, user can choose their region. from anywhere across the world.
       - If one region gets down the data is shared with other regions.

**DOMAINS OF AWS:**

1. **Compute:**

* **EC2(Elastic Cloud Compute):**

Let you have a resizable compute capacity.

It provides a clean slake that we can install everything new or work on a completely new system.

* **Elastic beanstalk:**

Similar to EC2, here most of them are prewritten like libraries and so on

1. **Migration:**

We can perform both and physical and virtual migration of data to other data centers using AWS.

* **Snowball:**

A storage system that helps us to migrate data.

1. **Security and Compliance:**

The are various process performed such as authentication, verification and so on.

1. **Storage:**

* **S3:**

A bucket object system. The storage space is called bucket and the files we store is called objects.

* **CloudFront:**

Storing content in every network.

* **Glacier:**

Usually used to store archive data since they cost only so minimum price.

1. **Networking:**

We have services like **VPC** (Virtual Network), **Route 53** (DNS), **Direct connect** (Least internet connection on AWS).

1. **Messaging:**

We have **Cloud Drawer** and **Amazon Simple Notification Service** is a notification service provided as part of Amazon Web Services since 2010. It provides a service for sending messages. **Amazon SNS** acts as a single message bus that can message to a variety of devices and platforms

1. **Database:**

* Storage – store executable files
* Database – it’s a storage but stores only data that / can be queried using a query language.
* **Aurora:**

Its similar to SQL but 5 times faster than SQL.

* + - * + **DynamoDB:**

It’s a NoSQL database

1. **Managing Tools:**
   * + - * **CloudWatch:**

A monitoring tool we have set alarms and so on using this.

**CREATING AN AWS ACCOUNT:**

Amazon cloud account free tier provides 12 months of free access and you can use it to access most of the services provided by AWS.

It asks your credit card details while creating an account just for verification purpose. Then it will take 2 rupees but give it back in 2 or 3 days.

**INSTANCES IN AWS**

**What is an Instance?**

An instance is a virtual server for running applications on Amazon EC2. It can also be understood like a tiny part of a larger computer, a tiny part which has its own Hard drive, network connection, OS etc. But it is actually all virtual in nature.

A server consists of lots of instances made virtually to run independently.

**What is EC2?**

**Resizable** compute service in the cloud. It is

* + - * + Scalable
        + Cost efficient
        + Flexible

**Types of Instances:**

* **General Purpose Instance:**

For applications require a balance of performance and cost. Where you need a prompt response and less processing at a low cost.

E.g. Email

* **Compute Instances:**

For application that require a lot of processing from the CPU.

E.g. Streaming

* **Memory Instances:**

For Applications that are heavy in nature and therefore require a lot of RAM.

* **Storage Instances:**

For applications that are huge in size or have a data set that occupies a lot of space.

* **GPU Instances:**

For applications that require some heavy graphics rendering.

**Instance Pricing Models:**

1. **On-Demand**

An temporary instance paid only for a fixed time and terminate organization ensuring that their data is secured and not shared with others. Its Costlier.

1. **On Spot**

A Spot Instance is an instance that uses spare EC2 capacity that is available for **less than the On-Demand price**. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly.

Only suggested for volatile memory

1. **Reserved**

Amazon EC2 Reserved Instances (RI) provide a significant discount (up to 72%) compared to On-Demand pricing and provide a capacity reservation when used in a specific Availability Zone.

**Instances based on Functionality:**

1. **Burstable:**
   * It starts with a base power utilization if the traffic is heavy it burst the CPU performance to 100%.
   * In order to use it the user is given certain credits. If he don’t use it , the credits will be remained for later usage
2. **EBS Optimized**
   * For processing high data continuously
   * They provide High input / Output process
3. **Cluster Networking**
   * A single cluster perform a particular functionality like a cluster for high performance and another for storage purpose.
4. **Dedicated**
   * It is for data security.

**Use Case:**

Take the example as Edureka.

1. analysis of customer’s data – Burstable (customer data is not constant)
2. Auto Response E-mail System – EBS optimized with iops (I need high performance but can’t pay high)
3. Search Engine and Browsing – Cluster Network (Since performing two different works)
4. Confidential Data Processing – Dedicated

**Hands On – Instances (1:11:16 – 1:25:29)**

* 1. Go to AWS Management Console
  2. Choose EC2
  3. There are several instances like Microsoft, Linux and so on
  4. You can create Amazon Machine Images (AMIs) a template of created instances.
  5. You can choose type of Instance: if ‘**micro**’ you get 1 vCPU and 1GB of Memory and EBS backed up. Low to moderate network performance
  6. Add a root storage (It is a constant storage)
  7. Add a tag
  8. Configure Security group
  9. Launch instance if needed add rules
  10. Generate a key pair for more security
  11. Download the key pair
  12. Make sure keep the key pair, we can’t access the instance if we lost the key pair
  13. It takes some minutes for some security and status checking purposes
  14. Go to EC2
  15. Running Instances
  16. We can start/stop a instance.
  17. While creating a AMI we need to stop the instance at the moment and take AMI/snapshot
  18. After using an instance, we can terminate them
  19. We can change instance type also… to higher versions if needed
  20. To launch a instance select the instance and click connect.
  21. Click get password…browse the downloaded key pair in it.
  22. Copy the password
  23. Now you can launch the remote desktop connection.
  24. It will ask for the password . Give the copied password
  25. The instance will be launched( here we choose windows OS instances)
  26. These are virtual machines running in cloud
  27. We can take AMIs also
  28. Go to AMIs under Images
  29. Go to actions Create Snapshot
  30. Also we can copy a snapshot and save it to another instance. Go to Modify Volume under the Actions. We can take a snapshot and we can detach the volume .
  31. And paste the snapshot in another instance. By creating a snapshot in the particular instances.
  32. We need to terminate the instances after using them or else we would be charged.

**AWS LAMBDA:**

* AWS Lambda is a ‘serverless compute’ service. Where developer don’t have to worry about which AWS resources to launch to manage them.
* The user needs to put a code of lambda it will get executed.
* Save time
* Used to execute backend code.
* When there is an Event Source the lambda functions are triggered.
* Example: we are uploading images as object in the S3 bucket .. Where the S3 bucket sends a notification to Lambda and this acts as an invoke call for the Lambda Function. The Execution role in lambda is known as IAM(Identity and access permission) for giving access to AWS resources.
* AWS Lambda is compatible with Node.js, Python and Java
* All we need to do is upload the Zip file and define an event source.

**AWS Lambda Vs AWS EC2:**

* + In EC2 we need to manage Load Balance , Create Virtual Machine and so on .
  + But in Lambda we need to insert only a plain code where the rest will be taken care by the AWS itself.
  + But if there is continuous Stream of Data we need to go with EC2 . Because The architecture of Lambda has some restrictions like if your running a small scale company with random emails receiving everyday low or high then lambda can be your choice but If your running a Large scale company where the number of emails receiving is endless then you need go for EC2.

**Pricing in AWS Lambda:**

* + It’s a pay as you go model.
  + The parameters are,

1. The number of requests that you make to your lambda function
2. The duration for which your code executes (It takes the time from which your code starts and get return or terminated)

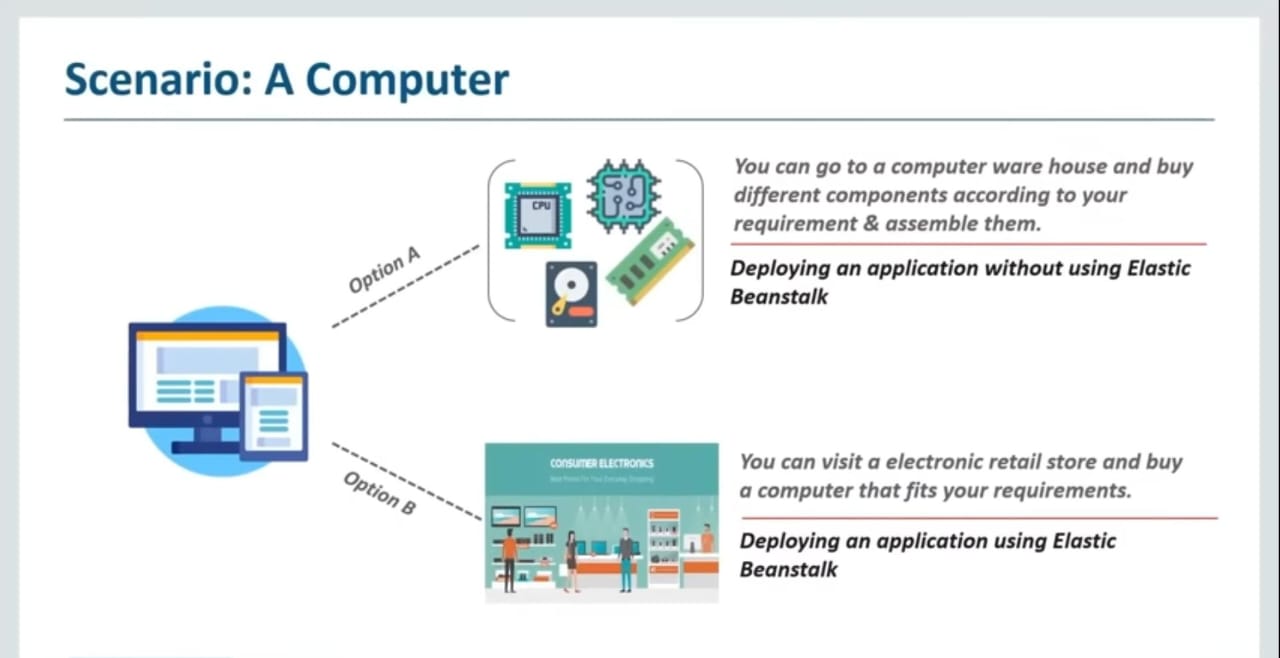
**Hands On – Lambda (1:32:20 - ):**

* 1. Under compute section click lambda
  2. In lambda click functions .(note: its take some time to show up the recently uploaded functions)
  3. Actions can be performed only on functions that are already present.
  4. Under Actions you have View Details, Test, Delete.
  5. Click create function.
  6. In create function you have 3 options like Author from scratch, Use a Blueprint, Browse serverless app repository
  7. Choose from scratch and give the function a name (here we give as hello world)
  8. Choose runtime(here we choose python 3.8)
  9. Permissions(by default AWS upload logs to CloudWatch Logs(an monitoring and managing service that provides data and insights to hybrid and on-premise applications). We can customize it)
  10. Under Execution Role choose Create a new role with basic Lambda Permission.
  11. Now Create function
  12. It will redirect to the lambda function hello world page..
  13. Here we have 3 elements

1. **Configuration** 
   * Where we have designer, we can add triggers and destinations.
   * Under trigger we need to choose a service and set the trigger you need from that service. Click the Hello World function from the configuration element. Under designer. Where you will get a code editor to write the code. After writing the code and deploy it. Layers are resources we have libraries and other dependencies.
   * Add other layers to separate function code from other dependencies. Under Add Layer we have 1.AWS layers to select some layers respective to our function, 2.Custom Layers to use previously created layers and 3. Specify an ARN(Amazon Resource Name) to use layer shared by another account or to run a function that does not run with our dependencies.
   * Add destination are used to perform invocation after a success or failure.
2. **Permission**
   * + As we give permission to Store logs in amazon CloudWatch logs.
     + We can add some other permission also if needed.
3. **Monitoring.**

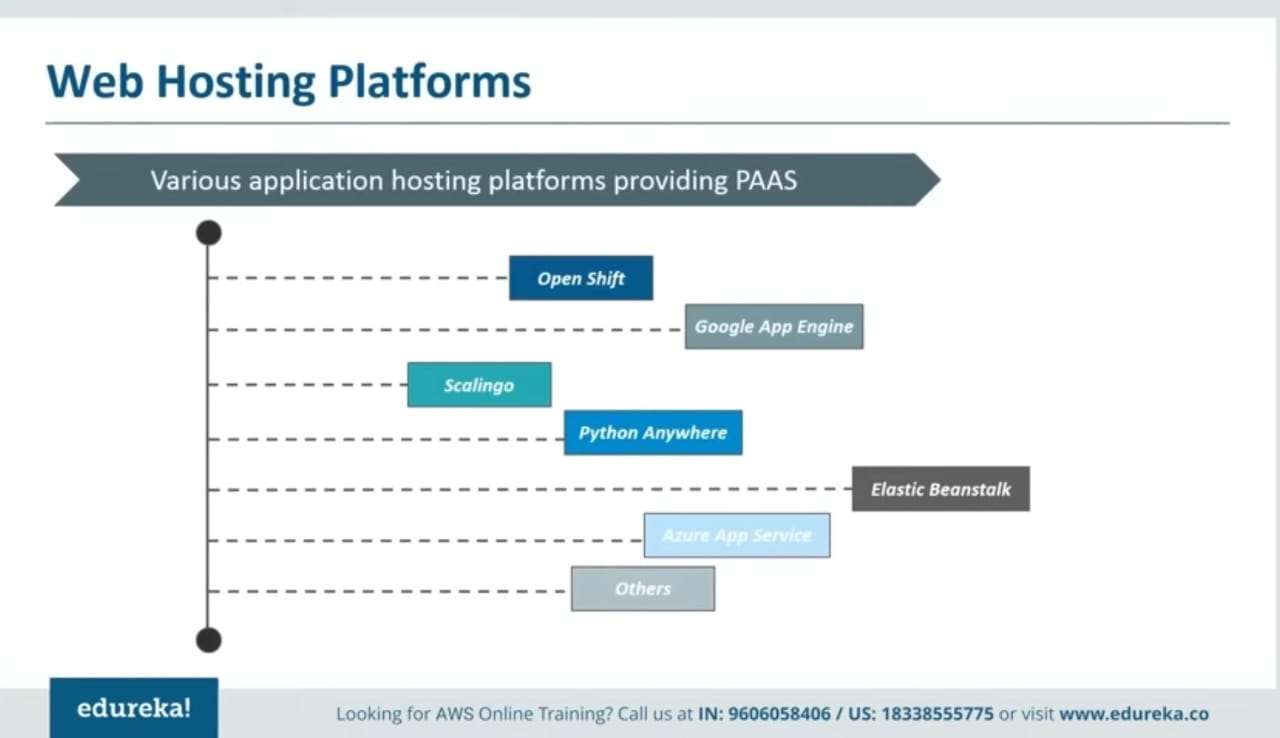
* Used to monitor the function, trace and debug.
  1. Create a test event and name it. By clicking configure test event.
  2. In order to run the code, click test.
  3. Now we can monitor that our code have run successfully in the monitoring element.
  4. Now we can add a trigger to check no. of times our code it executed to do this we use ‘API Gateway Service’.
  5. While configuring the trigger select HTTP request, Security -> open(but not in real world scenarios).
  6. You will get a API endpoint link via API Gateway service , open the link in a new tab
  7. Once you done with your lambda function make sure that you have deleted it under actions drop down list.

**AWS ELASTIC BEANSTLAK:**

* AWS 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 .
* Example:
* ****
* In Elastic Beanstalk all our work is to concentrate on the code other than the EC2 instances, storage and all other works are done by the Elastic Beanstalk.
* **Applications:**

1. Quicker Development – Every EC2 instances, load balancing , storage and everything will be take care by Elastic Beanstalk
2. Simplifies Operations
3. Cost Effectiveness – if we deploy the application instead of PaaS, we need to pay a lot for additional software for testing and security purposes but all these software are given as a package by using elastic Beanstalk.
4. Multi-tenant Architecture – Make it useful for users to share their application with multiple users with high security.
5. Better User Experience. – you can collect feedback on different stages of application development and we can make improvement.

* **Various other Platform as a Services:**

****

* **Features of Elastic Beanstalk:**
* **Components of Elastic Beanstalk:**
  1. Application:
     + - Its is similar to a folder. In Elastic Beanstalk it is a collection of Environments, versions and environment configurations.
       - E.g. Tomcat-webapp
  2. Application Version:
     + - When ever we make an update in our code the Beanstalk makes the copy of the file and save it as different versions.
       - So that we can deploy which version we need to use.
  3. Environment:
     + - Each environment can run only one single version of application at a time.
       - But Elastic Beanstalk make it possible to make multiple environments and make it possible to run multiple versions of application at a time for different stages of applications like production stage, deployment stage and so on.
  4. Environment Tier:
     + - There are two types of tiers:
         * Web Server Tier – serves HTTP requests
         * Worker Tier – processing background tasks.
  5. Environment Health:
     + - Report the health of web server
       - Uses four colors to describe health status:
         * **Grey** - Environment is being updated (when trying to upload different versions)
         * **Green** – Passed recent Health check
         * **Yellow** – Failed one to three checks
         * **Red** – Failed three or more checks
* **Architecture of Elastic Beanstalk (**We have 2 tiers**)**
  1. **Web Server Environment: -** it handles HTTP requests
     + - * **Beanstalk Environment –** place to run application & runs only one version at a time. Beanstalk allows you to create multiple environments.
         * **Elastic Load Balancer** – It distributes requests among all the instances. So that all the requests are handled and no request is denied. The requests are shared as a URL called **CNAME.**
         * **Auto Scaling Group –** If traffic is high, it automatically install few EC2 instances and if traffic is low it terminates several EC2 instances.
         * **EC2 Instance –** Beanstalk will install the EC2 instances but the necessary files for an instance like OS, servers and other software are installed through **container type.**
         * **Host Manager-** monitor using CloudWatch or others services and gives report about servers and so on.
         * **Security Groups –** Not anybody can access the system. Client can access the application only via port 80. You can also add other security groups.