Page No.
Date / /

What is compute?

compute resources can be consumed in different quantities for different lengths of time across a wide range of categories, but typically processing billions of instructions

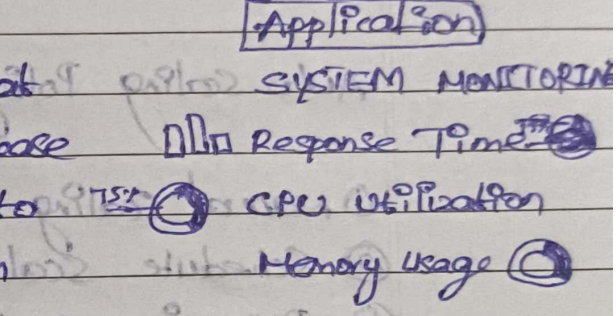
Amazon EC2: To be used to continuously for months or years

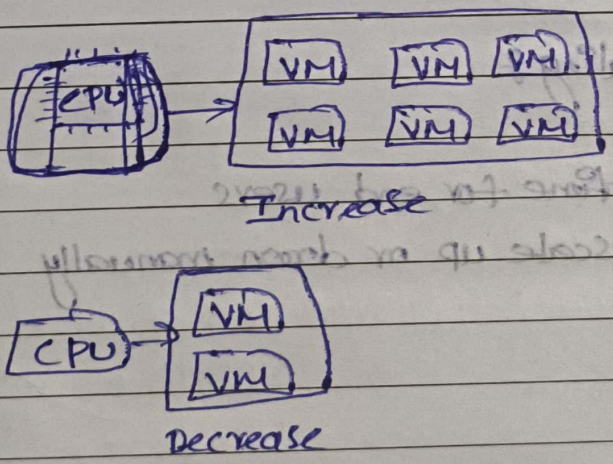
AWS Lambda: ↓

A few milliseconds of compute resources may be utilized within a serverless AWS Lambda function before relinquishing that compute power.

EC2 Auto Scaling:

Auto Scaling is a mechanism that automatically allows you to increase or decrease your EC2 resources to meet the demand based off custom defined metrics and thresholds.





Components of EC2 Auto Scaling:

① Create a launch configuration
② Create an Auto Scaling group

Page No.
Date / /

① Create a launch Template:

click on [EC2] → left hand side [Instances] → [Launch Templates]
→ click [Create Template] → [Create new Template]

② Create launch configurations:

Auto Scaling → launch configurations → Create launch config
Quick start → Amazon

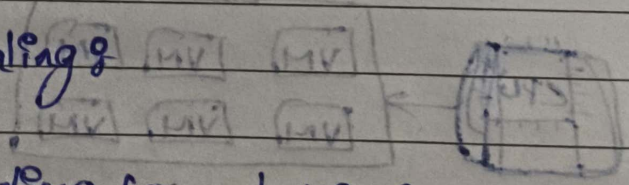
③ Auto scaling:

EC2 → Auto Scaling → Auto scaling groups → Launch Template

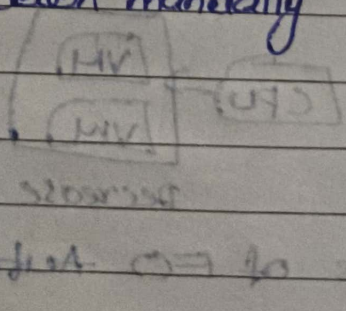
Auto scaling Policy Types:

① Manual scaling
② Schedule Scaling
③ Dynamic scaling
④ Predictive scaling

① Manual scaling:



Reduce downtime for end users
Flexibility scale up or down manually



Page No.
Date / /

What is Aws Elastic Beanstalk?

Allows you to upload code of your web application along with environmental configurations.

Automatically provision and deploy the appropriate and neccessary resources required with AWS to make the web application operational.

The resource can include of the AWS services and features, such as EC2, Auto scaling applications health monitoring and Elastic Load Balancing in addition to capacity provisioning.

⇒ To deploy the correct infrastructure to run the uploaded code.

Compatible with:

* Packer Builder
* Single container Docker
* Multicontainer Docker
* Java with Tomcat
* PHP
* Python
* Preconfigured Docker
* GO
* Java SE
* .NET on WS Server with IIS
* Node.JS
* Ruby

⇒ It's a free

Application Version: A very specific reference to a section of deployable code.
Environment: To an application version that has been deployed. AWS
Env. config: Parameters, settings, resources.
Env. Tier: Elastic Beanstalk, designed, Web or Workerstier.
Conf. Template: Baseline, new, unique, environment config.
Platform: components os, Programming lng, components of Elastic Beanstalk
Applications: A collection of diff. environments, config and application versions.

Page No.
Date / /

Environment Tiers:

HTTP Requests — Web Server Environment
SQS Queue — Worker Environment

Web Server Tier:

Direct web traffic to right servers → Route 53

Automatically distributes incoming application traffic & scales resources to meet traffic demand → Elastic Load Balancer

Manage the capacity planning of your app based on the load received. → Auto Scaling

Part of the auto scaling gp → EC2 Instances

Allows port 80 to be open to everyone. → Security Groups

You can develop and add your own Elastic Beanstalk configuration files within your applications source code.

* YAML
* JSON

Deployment Options:

Deployment Policy: All at once
Rolling
Rolling with additional Batch
Immutable

Page No.
Date / /

Configuring AWS Elastic Beanstalk:

Dashboards
config
Logs
Health
Monitoring
Alarms
Managed updates
Events.
Tags.

→ Search → Elastic Beanstalk.
→ Create a Web app → Application Server Info.
→ App Tags → Base configuration → App code → Sample code → upload your code. → config presets → low cost → High availability → custom configuration → Modify Software → S3 storage → cloud watch logs.
→ Root volume → EC2 security groups → Modify capacity → Auto scaling
→ scaling triggers → Modify load Balancer → Access Log files.

Monitoring Health checks:

Best Elastic Beanstalk health agent?,
A daemon process that runs on each EC2 instance in your Environment, which monitors the operating sys and app level health metrics and reports issues to Elastic Beanstalk.

Page No.
Date / /

An Overview of Aws lambda

Maintaince of EC2

- Installation
- Patching
- Scaling
- Storage Mangement

AWS. Responsibility

- Coding
- Deployment

customer Responsbility

Aws lambda

function example(input1, input2)
//code
return result;

lambda functions are made up of:

* code
* Permissions
* Environment Variables
* CPU/Memory

Memory (MB)

your function is allocated cpu power that proportional to the amount of memory you select.

creating a lambda Function Demo

lambda_function.py

```
import json
import urlib.parse
import boto3
s3= boto3.client('s3')
  def lambda_handler(event, context)
try
```

event exception
print(e)

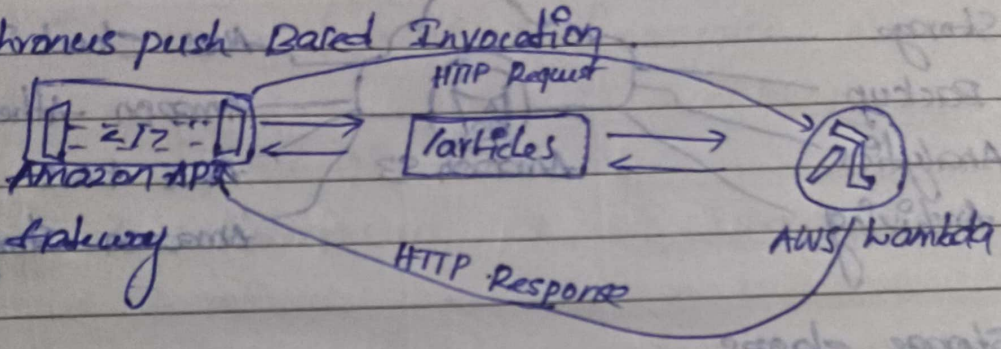
Page No.
Date / /

Invoking Lambda function

Aws Mangement console
AWS CLI
AWS SDKs
Triggers

This API provides three different invocation models.

Sychroneus push Based Invocation.



Stream (Poll Based) Usecase: Processing messgs from a stream or a queued
Ad: Message filtering.

Synchronous: use case: if your app needs to wait for a response.
Advantages: Helps maintain order

Asynchronous: Use case: if your function runs for long periods of time and does not need to wait for responses.
Adv: Offers automatic retries, a built-in queue and a dead letter queue for failed events.

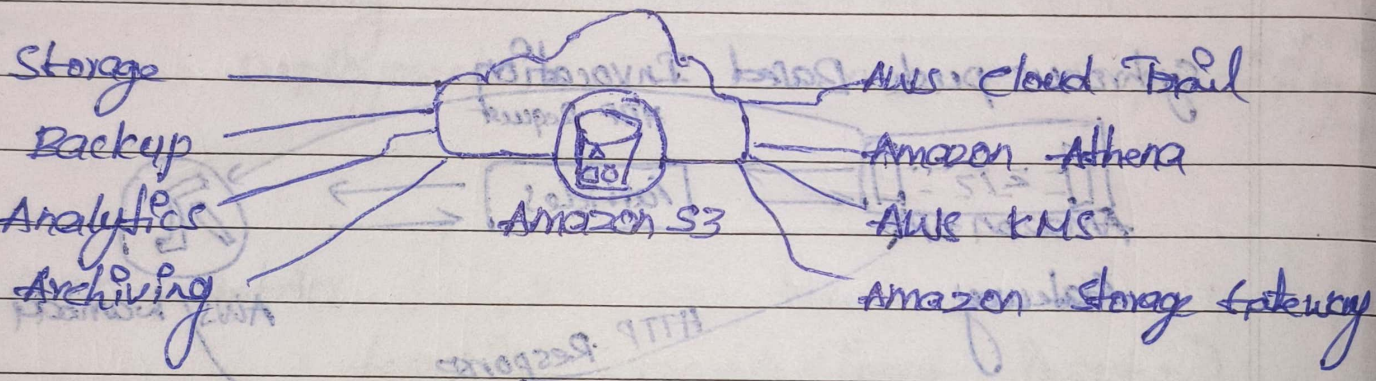
Aws service:
→ If an AWS Service invokes your function, you cannot select an invocation type.
The AWS service selects the invocation method for you

Page No.
Date / /

Storage:

Amazon Elastic Block Store (Amazon EBS)
Amazon Elastic File System (Amazon EFS)
Amazon S3
Amazon Glacier

Use cases: Integration



Storage classes

S3 Standard
S3 INT
S3 S-IA
S3 Z-IA
S3 Glacier
S3 G-DA

Server-side Encryption Mechanisms

Page No.
Date / /

Developer Tools:

What is CI?

Continious Integration:-

CI helps to verify that code is one step closer to production quality code.
Its only continious integration if developers are continiously integrating their code.

Creating a Development Environment

Mirror production is best.
Vgrant + Virtual Box.

Version control

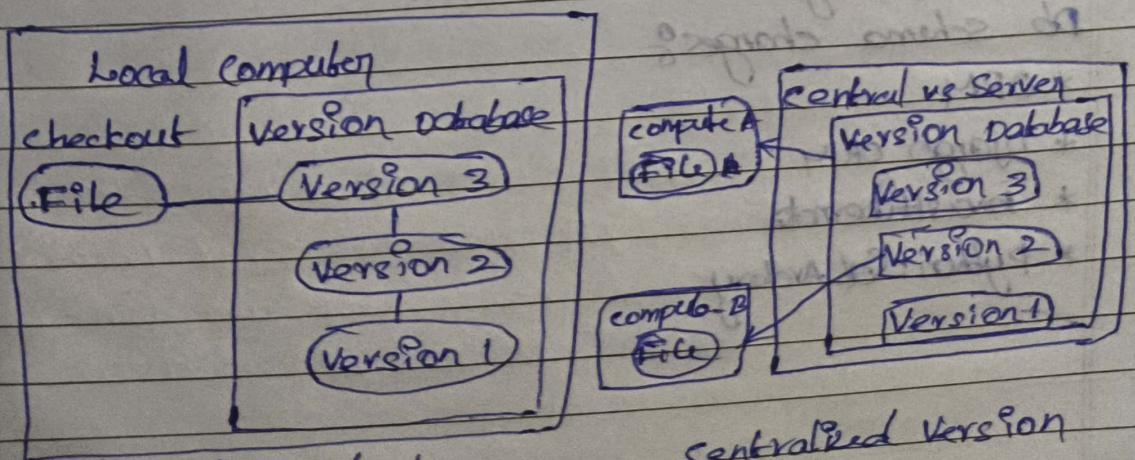
Version control . Source control . Revision control

Version control

Version control is about tracking the changes to one or more files over time files can be anything.

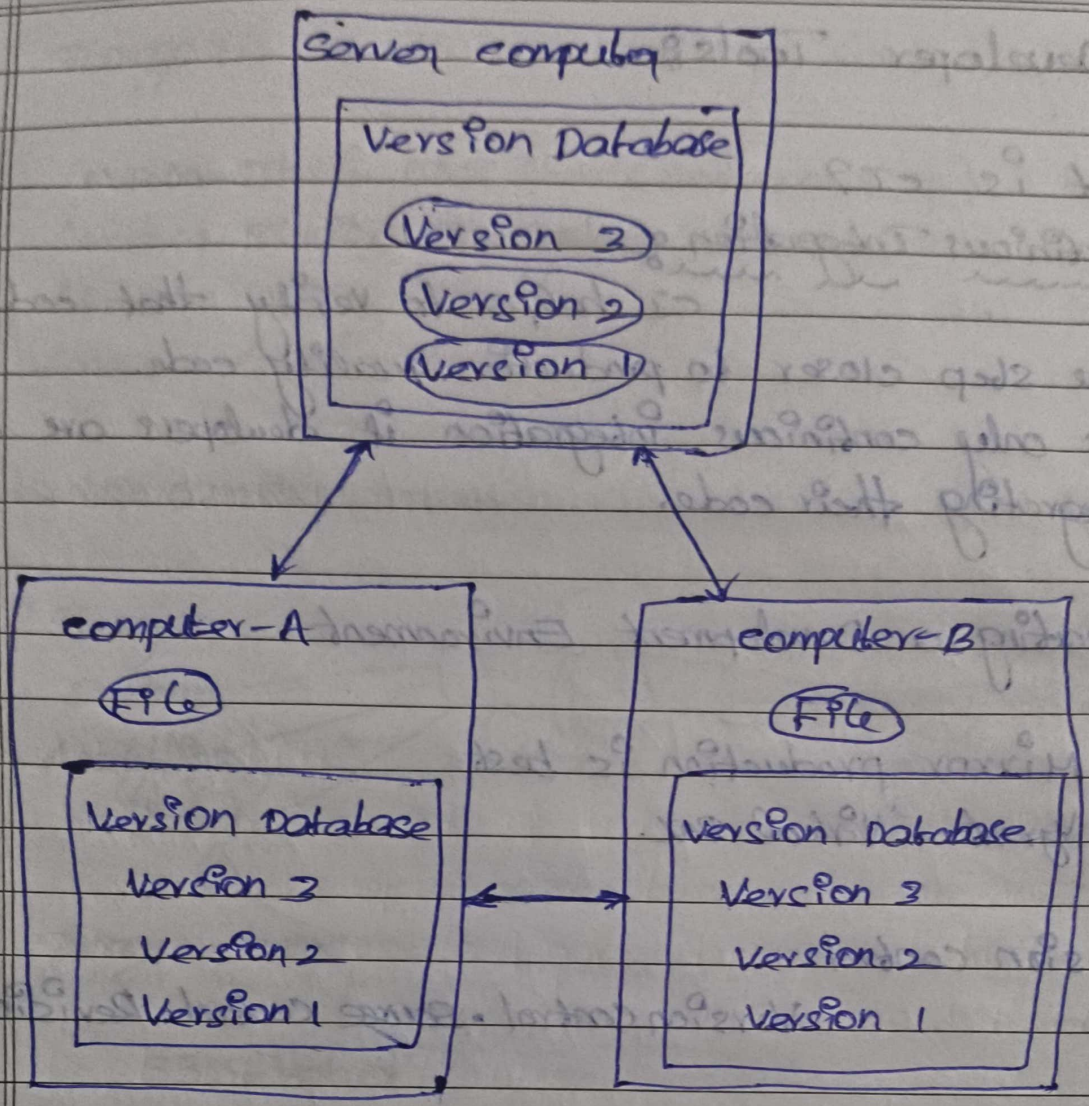
Types of version control

Local Version control
Centralized Version control
Distributed Version control



Local Version control

Centralized Version



Distributed Version control.

Testing

CI Testing

* To test code works correctly
* Unit test
* Linters
* code coverage

Db schema changes

* ORM
* Django ORM
* Java: Hibernate
* Python: Sql Alchemy

Page No.
Date / /

→ CI with Jenkins
→ Putting All Together

We need to deliver quickly software.

Break s/w down into pieces
- Search API
- User Management
- File processor
- Payment API

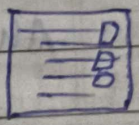
→ Dependency conflicts
→ Logic conflicts
→ Version control
→ Handle merge quickly
→ The term build
→ Testing environment
→ unit test commit
→ code ship.

Continuous Delivery

continuous delivery is a way of building s/w such that it can be deployed to a specified environment wherever you want to

Jenkins

As well as more depth
Security Audits
* Deploy only the highest quality version to production.
* Improved cycle time
* CI/CD
* Better customer feedback loops.



Date / /

coding for continuous Delivery

OWSAP-10
SQL Injection
ORM injection

→ Inversion control
→ Secure coding practicis.

Summary

→ AWS Amplify
Provide a set of tools and services for full stack application development on AWS.
Allows you to configure, deploy and host applications that use AWS services and run on ios, Android, React Native, or the Web.
Includes an open-source framework with libraries UI components, and a CLI.
Supports authentication, analytics, offline data, and push notifications.

AWS cloud 9 :

The Environment
The IDE

The Environment :

Main types of development environment
EC2-based : php. Python, javascript, Docer, Git
SSH : Create, manage, stop, and delete the instance your self. Download. Install

Page No.
Date / /

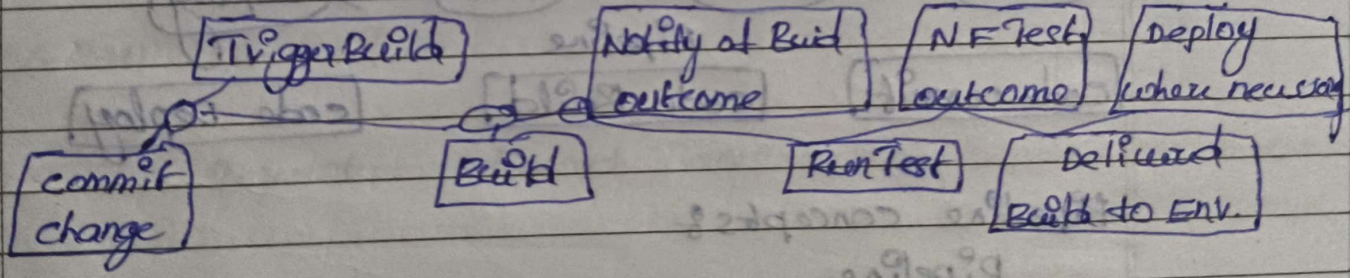
The IDE has the functionality of local desktop-installed IDE as it includes

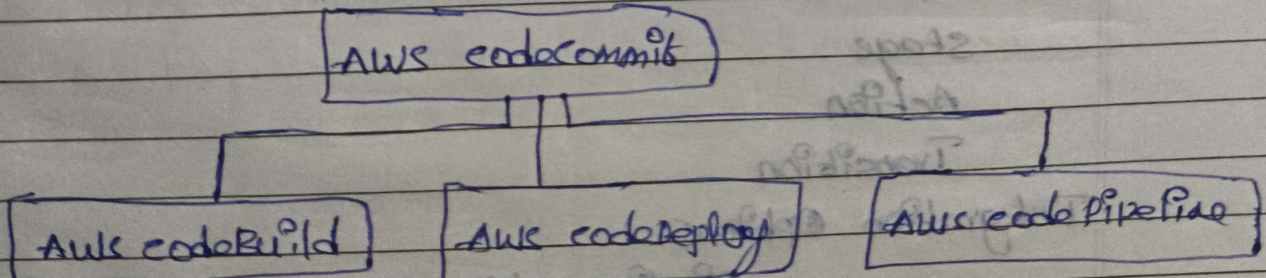
* A code editor
* A section to manage project files
* Environment Members
  Read, Write
* AWS Lambda
* A terminal
* A debugger.

AWS cloudshell :

→ Browser-based Linux shell experience that provide secure, Pre-authenticated access to AWS CLIs and other tools.
→ Makes it easy to interact with AWS services without needing to set up a local development environment first.
→ Security is pre-configured based on the credentials of user currently logged in to the AWS console.
→ Access in any supported region by clicking [>] icon in top navigation
→ Pre configured with AWS cli and CLIs for Elastic Beanstalk, Ecs and SAM

Introduction to AWS codecommit





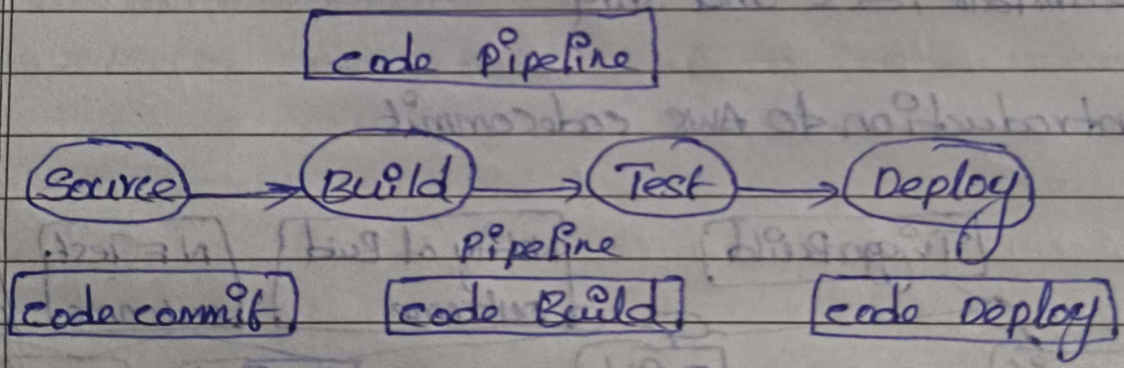
Date / /

AppSpec File for EC2

⇒ hooks
⇒ files
Appspec for lambda
hooks: Runs lambda functions
Resources: where you specify lambda.

Introduction to AWS code Pipeline

Automate
Track
Learn
Automation of build, test and release
Manual Approvals
Pipeline History Reports
Pipeline stages Visualization



Pipeline concepts:
Pipeline
Stage
Action
Transition
Integration Options
Pipeline Action Types
Invoke lambd
codepipeline with

Page No.
Date / /

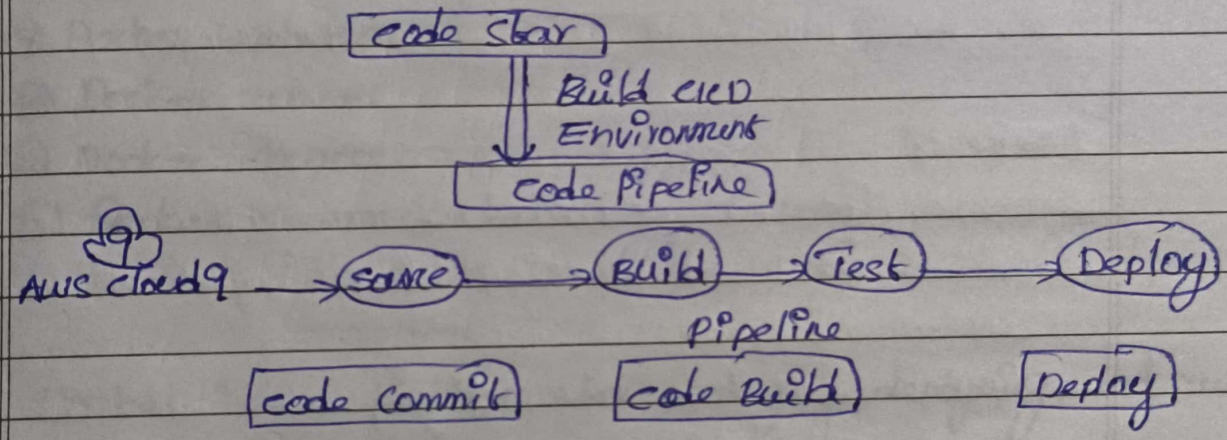
AWS code star :

Create New applications
Work Across your team securely
Manage slw delivery easily

Features :

Pre-configured CICD Workflow Templates
Dashboard visualization
Team Membership Management
Issue and Ticket Tracking Integration.

CICD Environment



X-Ray Data Flow

Trace Request - Record Traces View Service Map ~~Analyze~~ Issues