Cloud Computing

* Abstraction of IT infrastructure
* IT infrastructure available over the internet
  + Web Servers
  + Databases
  + Networks
* IaaS Infrastructure as a Service
  + Providing it hardware over the internet
    - Computers
    - Memory
    - Storage Disks
    - CPU
      * Examples
      * EC2 (A virtual computer)
      * EBS Elastic Block Store (Virtual hard drive)
* PaaS Platform as a Service
  + Hardware + software for you to develop an application on
  + Not a completed product
    - Examples
      * RDS (Relational Database Service)
      * S3 (Simple Storage Service)
* SaaS Software as a Service
  + Providing a completed application over the web
    - Examples
      * Gmail
      * Netflix
      * YouTube

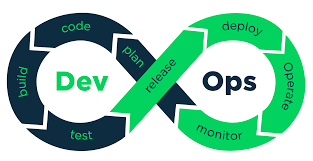
EC2

Elastic Compute Cloud

* It Is a virtual computer
  + Pick the OS
  + The amount of memory and processing power
  + Pick the size of the hard drive
* Because it is available over the internet
  + Anyone could access it if you configure the computer’s settings
  + Therefore it can be used to make web servers

DevOps

* Portmanteau of **Dev**elopment and **Op**erations
* Continuous feedback loop between the development and operations of your applications
  + Development
    - Design of the app
    - Implementation
    - Testing
  + Operations
    - Deployment (making it available online for other to use)
    - Monitoring (checking to see if the application has not crashed)
    - Metric (how much traffic seeing and when)



Setting up pipelines

* We want make development as fast and easy as possible
* We want software developers to quickly be able to test and deploy their code
* We want software developers to easily see if their code works with other code written by team members
* Continuous integration
  + Every developer will work on their local machine
  + They will push code to a shared repository
  + They work on their own branch
  + They can make a pull request
  + Whenever a developer makes a push to the repository the application will be automatically built and test
  + Fail fast and get feedback as soon as possible
* Continuous Delivery
  + Extension of CI
  + The build process create a “Deliverable”
    - Some sort of artifact/piece of software
    - Something.jar
    - Something.war
    - Something.zip
  + This deliverable is often put into some sort of staging
* Continuous Deployment
  + Extension of CD
  + If your deliverable passes all of the tests (Junit tests and other automated tests)
  + Your application is immediately deployed for your end users to access and use
  + You do not need human approval for the code to be deployed
    - This can make deployments incredibly fast and responsive
    - Amazon does some 20,000 deployments a day

Jenkins

* Popular tool for build automation and Continuous integration
* Used to create pipelines
* Steps to create a Jenkins project
  1. Install Jenkins
  2. Log into Jenkins
  3. Create a Jenkins job ( a project)
  4. Link your Jenkins job to a remote repository (like git hub)
  5. Write a build script that will call maven to build your application
  6. The script will take the completed .war file to the tomcat webapps folder

Maven

* Build automation and dependency management tool
* Java Specific
* Pom.xml (project object model) stores all information about your Java project
  + Name
  + Java Version
  + Dependencies
* Maven goals
  + Mvn package (builds the application)
    - Also automatically runs your Junit tests
  + Mvn test (will run anything in the tests folder)
  + Mvn clean (removes what is in the target folder)

AWS

* Amazon Web Services
  + The largest source of Cloud Computing/ infrastructure in the world
  + Competitors
    - Google Cloud
    - Azure (Microsoft) popular with .net
    - IBM
    - Oracle
    - Alibaba
* Regions
  + Geographic locations where their web servers are available
    - Spread out the servers over the world
    - Geographic proximity means the connection is faster
* Availability Zones
  + The actual data centers within a region
    - Northern East Virginia has 4 or 5 AZs
    - They provide redundancy to ensure that your services are always available
    - If one AZ goes down the others are up and running

AWS Services

* IAM (Identity access management)
  + Security for your AWS account
  + NOT security for the applications you create
  + Users
    - Humans
  + Roles
    - Machines
  + Policies
    - What a human/machine is allowed to dp
* User Steve
  + Policies
    - Can create RDS
    - Can Create EC2 less than 0.25 cents an hour
* Role MyEC2
  + Policies
    - Can read from RDS
    - Can create more EC2s

EC2 (Elastic Compute Cloud)

* Virtual machine in the cloud
  + Can be specialized for things like memory or processing power etc..
* AMI (Amazon Machine Image)
  + Initial software loaded onto an EC2
  + OS and possibly other development software
* EBS (Elastic Block Store)
  + The virtual hard drive attached to an EC2

RDS (Relational Database Service)

* A relational database available over the cloud
  + Oracle
  + MySQL
  + MariaDB
  + Aurora

Route 53

* DNS (Domain Name Service) provided by AWS
* Allows you to buy a domain name and have that traffic redirect to where you want.
  + Buy Ranieri.com and have it redirect to your website

S3

* Simple Storage Service
* Allows you to store files (objects) on the internet
* Files are stored in buckets
* Immensely popular service
* Can be used to perform static website hosting

Serverless Computing

* Developers want to be able to deploy and develop applications as easily as possible
* They often do not have the time or experience to set up complex web servers
* They do not want to waste time managing them
* AWS provides services which can abstract and perform different parts of your application
* Serverless application can scale incredibly well. You pay for exactly what is being used and nothing more

Full stack Serverless Application

* Front end- (s3 bucket)
* API layer url endpoints - (API Gateway)
* Middleware business logic (AWS Lambda)
* Database (DyanmoDB)

Security Group

* Virtual firewall to protect your AWS EC2’s RDS and any other instances you create
* Restrict access based on the port and IP address
* 50% of security is restricting access to your instances
* Minimum necessary access is the standard
  + If people should only be able to access port 8080 then only allow access to that one port

VPC (Virtual Private Cloud)

* LAN (Local Area Network in the cloud)
* Emulates the idea of having your cloud resources in a building physically connected to each other

Scaling

* Scaling refers to the cloud computing ability to create or remove resources to meet demand
* You only create and pay for what you need
* You can commission more resources when you get traffic spikes

Horizontal vs Vertical Scaling

* For web applications horizontal scaling tends be more efficient\*\*\*\*\*
* Some web applications its more the amount of requests than requests require a lot of processing power to handle
* Horizontal scaling benefits
  + Very fast to spin up a small ec2
  + Fast to decommission
  + Creates redundancy and makes the application fail safe

AWS ELB(Elastic Load Balancer)

* Performs horizontal

Sonar Cloud

* Tool for static code analysis
* Reads your code for code smells (bad practice)
  + Code smells
    - Bad naming
    - Overly complex
    - Poor encapsulation
    - Bad security
* Sonar Cloud does not run any tests