Git

- What is git?
    - git is a distributed software version control system

- What is version control?
    - It keeps track of every change ever made to a project during its lifetime
    - Provides a collaboration tool when working in groups

- Why do we need git repository hosting platforms?
    - As git is a distributed software version control system, we must have local repositories sync up with remote repositories

- What are some git repository hosting platforms?
    - GitHub
    - GitLab
    - BitBucket
    - & more

- List the git commands you know and what they do
    - `git config`: Modify configuration details such as username and email
    - `git init`: Creates a new repository
    - `git clone`: Clones an existing remote repository
    - `git fetch`: Downloads commits and files to local repository
    - `git pull`: Downloads commits and files and immediately merges with local
    - `git add`: Adds files to the staging area
    - `git status`: Displays the state of the working directory and the staging area
    - `git commit`: Commits the files to the repository
    - `git remote add`: Sets up the remote repository
    - `git push`: Sends the commit from local to remote
    - `git revert`: "Safe" rollback, undoes changes and recommits
    - `git reset`: "Dangerous" rollback, resets pointers, can orphan commits

- How would you prevent a file from being tracked by git?
    - `git remove <path to file>`: Removes the file from the staging area
    - .gitignore file: Specifies paths to intentionally not track

- What is a branch?
    - A branch is a separate version of the main repository. It is used to make changes in the code without altering the main branch.

- What are some common branching strategies?
    - Trunk-based: Commit to main unless the feature needs several days
    - Feature branching: Branches for each new feature, merged back into main when completed
    - git flow: Main and develop branches, hotfix for bugs, feature for new features to be added to develop branch, push to main for major releases

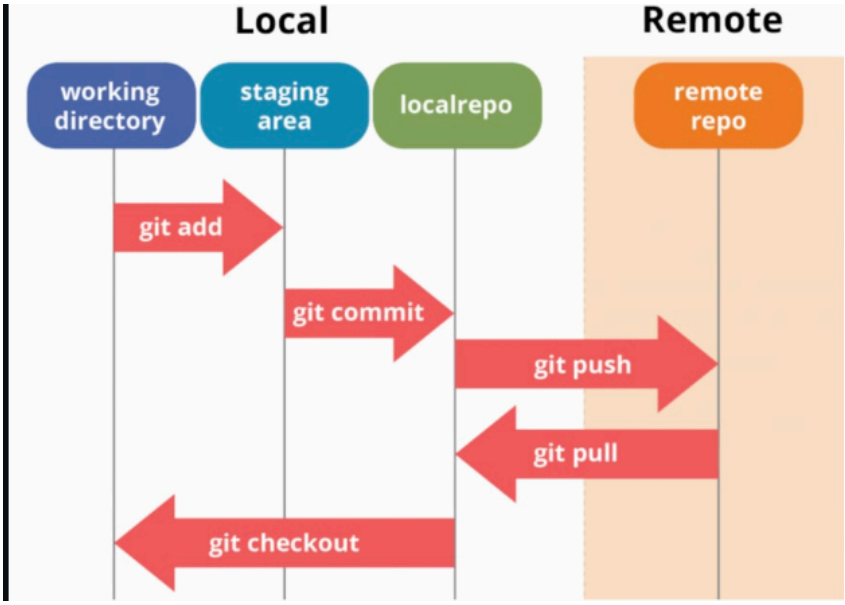
- How to create a new branch?
    - git branch <name> or git checkout -b <name>

- What is a merge conflict?
    - A merge conflict is when the version-control software cannot automatically resolve code differences between commits, such as the same line of code being altered or a file being deleted and added.
- How do you prevent merge conflicts?
    - We can avoid merge conflicts by
        - making smaller commits
        - standardizing development rules
        - communicating frequently
- How can merge conflicts become resolved?
    - In order to resolve the merge conflict, one or more conflicting files must be modified, either through command line or altering the files directly.

- What is a GitHub pull request?
    - A pull request is a request to pull changes from a branch into the main branch of the repository. It is an opportunity for collaboration before a feature is fully implemented. After review, if the code is agreed to, it is merged into the main.

- What is the git workflow for editing code and saving changes?
    - git pull: Get the most recent version of main
    - git branch: Move to a new branch (optional but recommended)
    - Make changes
    - git add <files>: Stage the changed files for commit
    - git status: View changes, add or remove files if necessary
    - git commit -m: Commit the staged files with a commit message
    - git push: Push the changes to the remote repository

- What is a commit?
    - A commit is a snapshot of changes to code, a new “version” of the software



- How would you go back in your commit history if you make a mistake?
    - git reset or revert

AWS

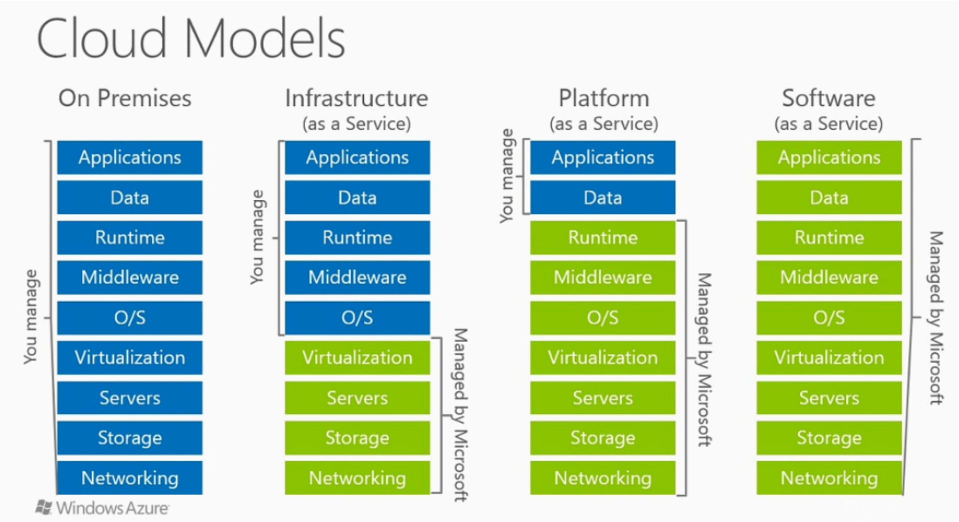
Cloud / AWS Overview

- How would you describe AWS? What is "the cloud" or "cloud computing"?
    - Cloud computing is the on-demand delivery of compute power, database storage, applications and other IT resources through a cloud services platform via the Internet with pay-as-you-go pricing.
    - The cloud just refers to a distributed collection of servers that host infrastructure and applications.

- Why is cloud computing so popular now?
    - It is popular because it allows companies to outsource many of the issues of on-site servers, such as scalability, security and redundancy.

- Define Infrastructure, Platform, and Software as a Service
    - IaaS
        - Outsources
            - Networking
            - Storage
            - Servers
            - Virtualization

        - Remain responsible for
            - OS
            - Middleware
            - RTE
            - Data
            - Applications.
        - Examples include AWS EC2, Digital Ocean
    - PaaS
        - Outsources
            - OS
            - Middleware
            - RTE
        - Remain responsible for
            - Data
            - Applications
        - Examples include Microsoft Azure, Google App Engine
    - SaaS
        - Outsources
            - Data
            - Applications
        - Examples include Dropbox, Salesforce



- What's the difference between a Region and an Availability Zone (AZ)?
    - Regions represent highly-available AWS data centers. There are 33 regions in total. Those regions are broken down into 105 availability zones that provide redundancy and protect against failure.

- How are you charged for using AWS services? Does it vary by service?

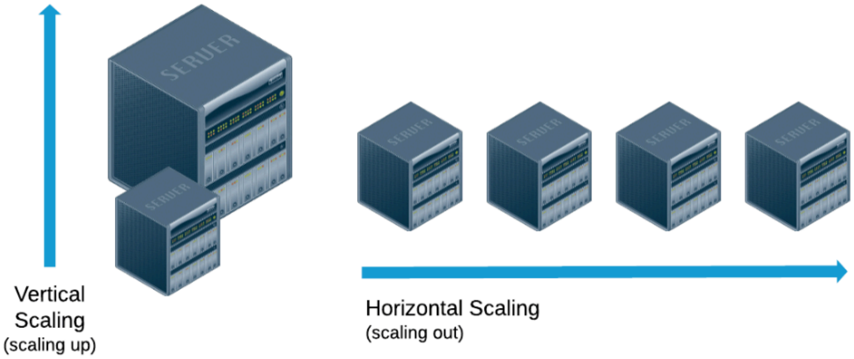
    - Pay-as-you-go: Charged for the services you consume
    - Can also commit to a certain size of storage over a period of time for a discount
    - Varies by service

- Different ways to interact with AWS services?
    - Command line
    - AWS console
    - Programmatic access

EC2 / Elastic Cloud Compute (*IaaS)*

- What is AWS EC2?
    - A web service offered by Amazon that provides resizable compute capacity in the AWS cloud
    - Enables you to manage a Linux/UNIX and Windows server instances in Amazon's Data Centers
    - Provides complete control of computing resources, which one can scale as per the requirement
- What is an EC2 instance?
    - A virtual server used for running applications on Amazon's EC2 is an instance
- What are the configuration options for EC2?
    - Preconfigured templates for your instances, Amazon Machine Images (AMIs)
- What are the different EC2 instance sizes/types?
    - General Purpose (e.g., t2, m5): Balanced compute, memory, and networking resources.
    - Compute Optimized (e.g., c5): High-performance computing.
    - Memory Optimized (e.g., r5): High memory-to-CPU ratio for memory-intensive applications.
    - Storage Optimized (e.g., i3): High I/O performance for storage-intensive workloads.
    - Accelerated Computing (e.g., p3, g4): GPU or FPGA-equipped instances for graphics rendering, machine learning, etc.
- Once you create an EC2, how to connect to it?
    - You can connect to an EC2 instance using SSH (for Linux instances) or RDP (for Windows instances) using the public IP address or DNS name of the instance, along with the private key or password associated with the instance.
- What are Security Groups? When defining a rule for a security group, what 3 things do you need to specify?
    - Security groups are sets of rules that act as virtual firewalls for EC2 instances, controlling inbound and outbound traffic. Source/destination, IP range and protocol
- What's the difference between scalability, elasticity, and resiliency?
    - Scalability is a characteristic of cloud computing through which increasing workload can be handled by increasing in proportion the amount of resource capacity. It allows the architecture to provide on demand resources if the requirement is being raised by the traffic.

    - Elasticity is the concept of commissioning and decommissioning of a large amount of resource capacity dynamically. It is measured by the speed by which the resources are coming on demand and the usage of the resources.
    - Resiliency refers to the ability to recover from a disaster or outage
- What is autoscaling?
    - Autoscaling is the process of dynamically scaling the amount of resources available depending on usage. The old model was "vertical scaling", or getting a bigger/faster server room as demand increased. This put a heavy burden on the company to procure and install these servers on-site. Horizontal scaling allows companies to use multiple smaller distributed servers to scale per demand.



    -
- Ways of paying for EC2?
    - On-Demand Instances: Pay for compute capacity by the hour or second with no long-term commitments.
    - Reserved Instances: Make a one-time payment for a discounted hourly rate in exchange for a commitment to use the instance for a 1- or 3-year term.
    - Spot Instances: Bid for unused EC2 capacity, potentially saving up to 90% compared to On-Demand prices.
    - Dedicated Hosts: Physical servers dedicated to your use, offering visibility and control over instance placement.

RDS / Relational Database Service

- What's an RDS?
    - "web service that makes it easier to set up, operate, and scale a relational database in the AWS Cloud."
    - "It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks."
- Which vendors are supported?
    - Amazon, PostgreSQL, MySQL as examples, 15 in total
- Why use RDS?
    - Outsource database management tasks such as
        - Server provisioning
        - Patching
        - Setup

        - Configuration
        - Backups
        - Recovery

S3 / Simple Storage Service

- What kind of data would you store on S3 vs a database?
    - S3 is meant for file storage. A database is meant for recording data.
    - S3 files are stored in “buckets” that correspond to root-level folders
    - Buckets are created in a region and store objects, which consist of a key/value pair and metadata
- Are there any limits on S3?
    - Individual objects can range from 0B to 5TB
- What are the rules for bucket naming?
    - Has to start and end with a letter or number
    - Certain characters or strings are disallowed
    - Globally-unique
- What are the different storage tiers?
    - S3 Standard: Frequently accessed data
    - S3 Intelligent-Tiering: Different tiers with different prices for more and less-frequently accessed data
    - S3 Glacier: Lower-cost for archival data
- Can you use S3 to host a front-end or back-end of an application?
    - Front-end

Docker

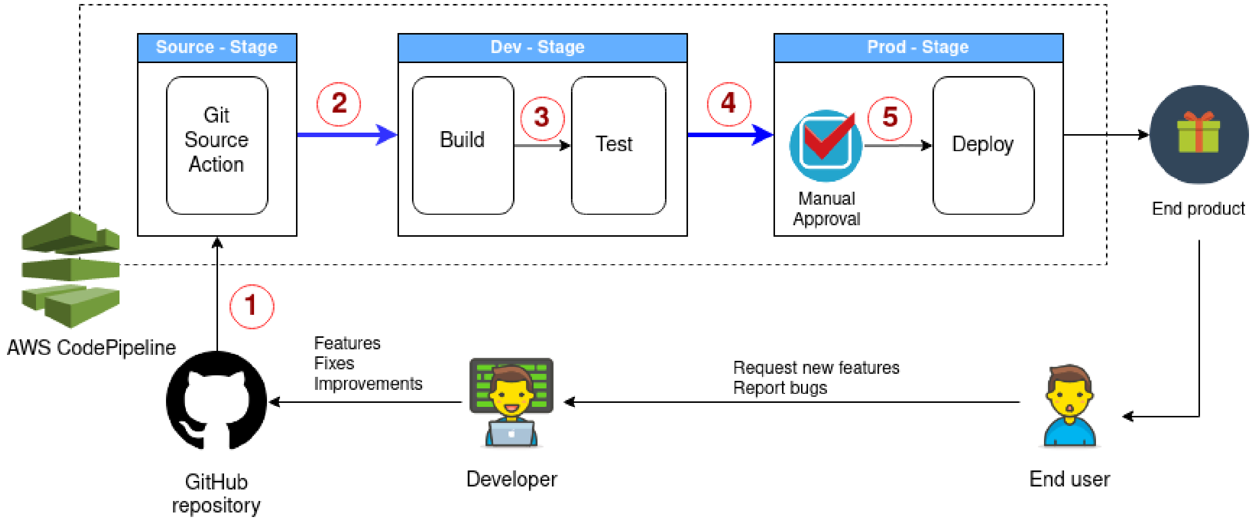
- What is a container? How is it different from a VM?
    1. A container is a lightweight, standalone, executable that includes everything needed to run software
    2. Often more lightweight and interact with the host OS’ kernel ***in contrast to VMs***
- Virtual Machine / VM
    1. operates like a separate computer within the host computer.
- What is the Docker Daemon?
    1. A background process that manages Docker containers. It handles container lifecycle operations including starting, stopping, and deletion of containers
- What is a Docker image? Container?
    1. Docker images are read-only files which contain the instructions to create a Docker Container. Containers are runnable forms of images that run in isolation of each other and their host system.
- How is a Docker image different from a Docker container?
    1. An image is a static, immutable snapshot of an app and its dependencies whilst a container is a runnable form of an image. Containers can be created, started, stopped, deleted, and more while images can only be changed through explicit modifications or rebuilt. Think blueprint vs building made off the print

- List the steps to start Docker, create a Docker image, and spin up a container
    1. Write a Dockerfile
    2. Build the image `docker build <pathToDockerfile>`
    3. Spin up the container `docker run <imageName>`
- What is the relevance of the Dockerfile to this process? List some keywords in the Dockerfile.
    1. The Dockerfile is essential for defining the configuration and dependencies of the application image. It contains instructions for building the image, such as installing dependencies, copying files, setting environment variables, and defining the entry point for the application.
    2. FROM: Specifies the base image.
    3. RUN: Executes commands in the container during the build process.
    4. COPY / ADD: Copies files from the host into the container.
    5. ENV: Sets environment variables.
    6. CMD / ENTRYPOINT: Specifies the command to run when the container starts.
    7. EXPOSE: Exposes ports to the host machine.
    8. VOLUME: Mounts a directory from the host into the container as a volume.
- What are some other Docker commands?
    1. docker pull: Pulls an image from a registry.
    2. docker push: Pushes an image to a registry.
    3. docker ps: Lists running containers.
    4. docker stop: Stops a running container.
    5. docker rm: Removes a container.
    6. docker rmi: Removes an image.
- What is a container registry? How would you retrieve and upload images to DockerHub?
    - Holds images, either public or private
    - Use `docker pull` and `docker push` to upload/download images
- If you want to store state for a container, how does Docker recommend doing that?
    - Use a volume or connect to external state management service
    - Volumes are file systems mounted to a container and exist on the host independent of the container

DevOps / Development and Operations

- What is DevOps?
    - Combination of philosophies, practices and tools into a continuous process
- What is the goal of various DevOps processes?
    - Allows organizations to deliver applications at a fast pace
    - Better serve customers
    - Compete in the market
- Explain CI/CD. What is the difference between Continuous Deployment and Continuous Delivery?
    - Continuous Integration: Continuously develop new code, achieved through a version-control system
    - Continuous Delivery: Automate testing, building and staging for deployment

  - Continuous Deployment: Automate deploying new code to production environment
- What tools have you used to achieve CI/CD? Explain how you've setup and used them
  - CI: git and GitHub for version control
- What is a DevOps pipeline? Explain the steps to setting one up



-
- What is a Jenkinsfile?
  - It's needed to create a custom pipeline. It consists of agents, stages, and steps.
  - An agent tells Jenkins where and how to execute the pipeline.
  - Stages consist of multiple steps that are performed by the Jenkins pipeline.
  - Steps are specific actions the Jenkins Pipeline does.
- What is a Jenkins Job?
  - Jenkins job is any automated process in Jenkins, such as building source code from git.
  - "Traditional way of configuring build jobs in Jenkins using the web interface." (CHATGPT ANSWER)
- How is it different from a Jenkins Pipeline?
  - A Jenkins Pipeline can use Jenkinsfiles so that the Job runs automatically (someone pls check - idk if this is right)
  -
  - Jenkins Pipeline: Allows defining build pipelines as code using a Jenkinsfile, enabling version control, reusability, and automation of the entire pipeline.(CHATGPT ANSWER)
- How to set up a Jenkins CI/CD Pipeline?
  - Configure your jenkinsfile
  - Navigate to your Jenkins server and select 'New Item' from the Jenkins menu.
  - Provide a name for your new item (i.e. "Pipeline-Example")
  - Click "Add Source" to choose the type and details for your repository
  - Click "Save". You can now build your project using this new pipeline
- What is a "build"? What is the end result of a build? What is the build tool you've used for Java projects?
  - A build is created when a job is triggered.
  - Their status is blue: success, yellow: unstable, red: failure, and grey: no build or aborted
  - We've used Maven in Java

- How can you check what caused a build to fail?
    - Build failures can be checked by examining the build logs, which provide information about the build process, errors, and warnings encountered during compilation, testing, and packaging.
- What is SonarQube / SonarCloud? Explain some of the features of it
    - SonarQube and Sonar Cloud are code quality analysis tools which increase the readability, security and maintainability of code.
    - Sonar Cloud is a cloud-based code review option that can review code in a cloud repository.
    - Code Quality Analysis tools are programs specifically designed to expose errors as well as code smells, which try to find vulnerabilities, bugs, and maintainability issues.

SDLC / Software Development Life Cycle

- What are the steps in the SLDC?
    - Requirements
    - Analysis
    - Design
    - Development
    - Testing
    - Integration/Deployment
    - Maintenance
- What is the difference between Waterfall and Agile methodologies? Explain the benefits and drawbacks of each
    - Waterfall: All stages are monolithic; once the Requirements stage is complete, you don't return to it. This can be good for smaller teams and projects where requirements are unlikely to change, but does not allow for flexibility and adaptability.
    - Agile: Stages are iterative, meaning the steps are followed through in shorter timeframes and expanded upon in the next iteration. It is suited for larger teams and situations in which customer collaboration is more values. The downside is that more experience is needed on the management side to monitor these iterations.
- Agile Manifesto Principles
    - Individuals and Interactions over processes and tools
    - Working Software over comprehensive documentation
    - Customer collaboration over contract negotiation
    - Responding to change over following a plan

      Basically, interaction and change over plans and documents.

- What specific Agile frameworks exist? What are the main features of each?
    - Scrum is the simplest Agile framework in use.
- What is the Scrum process? Explain each of the Scrum ceremonies
    - Sprint Planning: Plan the work to be done in the upcoming sprint.
    - Daily Standup: Daily meeting to synchronize and plan the work for the day.
    - Sprint Review: Demonstrate and review the increment with stakeholders.

  - Sprint Retrospective: Reflect on the sprint and identify improvements.
  - Artifacts: Product Backlog contains all features, enhancements, and bug fixes, Sprint Backlog contains items selected for the sprint, and Increment is the sum of all completed work from the sprint.
- How long is a typical sprint?
  - Sprints are typically 2-4 weeks.
- What is a "standup" and what should you report about your work?
  - <15 minute meeting at the beginning of the day, team members talk about what they are working on, where they are stuck and plans for the current day
- What is the role of a "Scrum master" in a project? What about the "Product owner"?
  - A "Scrum master" is a "servant-leader" of a team that facilitates communication between the product owner and the development team.
  - The "product owner" is the customer's representative in discussions and has authority over the project
- Explain the following metrics/charts: sprint velocity, burndown chart
  - Sprint velocity: The sum of story points that have been completed during the sprint
  - Burndown chart: Compares story points completed to story points remaining
- What is a Scrum board? Have you used any software tools for your team's Scrum board?
  - "an agile project management tool that enables teams that aim to work iteratively, to visualize, track, and manage work during a sprint, or fixed period of time" – *Atlassian*
- Name some technologies teams can use to keep track of progress on different projects, tasks, and due dates.
  - Kanban Boards

Event Driven Architecture

- What are Events?
  - Changes in state, or an update, like an item being placed in a shopping cart on an e-commerce site.
- What is Event Driven Architecture?
  - Architecture which utilizes Events to trigger and communicate between decoupled services.
- What are some use cases for using Event Driven Architecture?
  - Real-time data processing
  - Microservices-based systems
  - IoT (Internet of Things) applications
  - Systems that require high levels of concurrency and responsiveness
- What is an Event Stream in Kafka?
  - Event streaming is the practice of capturing data in real-time from event sources like databases, sensors, mobile devices, cloud services, and software applications in the form of streams of events; storing these event streams durably for later retrieval; manipulating, processing, and reacting to the event streams in real-time as well as retrospectively; and routing the event streams to different destination technologies as needed.
- What is the use of Kafka?

    - To publish and subscribe to streams of events, including continuous import/export of data from other systems
    - To store streams of events durably and reliably for as long as you want
    - To process streams of events as they occur or retrospectively
- What is the role of a Kafka Broker?
    - Responsible for transferring events from the publisher to the consumer
- What is a topic in kafka?
    - Events are organized and stored in Topics
    - Similar to a folder in a file system, with Events as files.
    - Topics are multi-producer and multi-consumer
    - Events in Topics are not deleted after consumption, can define how long Kafka should retain your events
    - Topics are **partitioned** and can be **replicated**
- What is a Kafka Producer? Consumer?
    - Producers are client applications that publish (write) events to Kafka
    - Consumers are client applications that subscribe to (read and process) those events
- How have you used Kafka?
    - Personally, no.