|  |  |
| --- | --- |
| **git** | **github** |
| Git is a version control system used to track changes in files over time | GitHub is a platform where Git repositories can be stored and shared |
| It runs locally on our computer | It is a cloud-based service |
| Git can be used offline, as it operates locally on our machine. | GitHub requires an internet connection because it is hosted on the web |

**Version Control System A version control system helps track and manage changes to files over time, making it easy to save versions, revert changes, and collaborate with others.**

[**Git repository**](https://www.geeksforgeeks.org/what-is-a-git-repository/) is a storage system that holds project files and tracks their changes over time.

**merge conflict error** in Git happens when two people make changes to the same part of a file in different branches, and Git doesn’t know which change to keep. You need to manually decide how to combine the changes before completing the merge.

Top of Form

**Docker** is a tool that packages applications with their dependencies into containers, ensuring they run the same in any environment.

**Container=**is a lightweight, standalone package that includes everything needed to run an application, such as code, libraries, and dependencies. It ensures consistency across different environments by isolating the application from the host system

**Image=**is a read-only template that contains all the necessary code, libraries, and dependencies to run an application. It’s used to create Docker containers. Think of it as a blueprint for containers

**Volume=**is used to store data outside of containers, allowing persistent data storage. Volumes are independent of containers, meaning data can persist even if a container is deleted or recreated. It is useful for databases and other applications that require durable storage.

**Docker-Hub=** is a cloud-based registry that stores and distributes Docker container images, allowing users to share and download pre-built applications. **Hypervisor=** is software that enables virtualization by creating and managing multiple virtual machines on a single physical machine. **Virtualization**=is the process of creating virtual versions of physical resources, like servers or storage, to improve flexibility.

Linux= **Linux** is a free and open-source operating system based on the Linux kernel.

**What is terraform**=Terraform is an open-source tool created by Hashi Corp for automating the provisioning and management of infrastructure. **Use=**1. it is use in Automate and manage infrastructure with code.2. Works with various cloud providers (AWS, Azure, GCP) and on-prem systems.

**What is ansible =**Ansible is an open-source automation tool used for configuration management, application deployment, and task automation. It uses simple, human-readable YAML files to define tasks, making it easy to automate IT processes.

**What is Jenkins =**Jenkins is used for automating software development tasks, particularly CI and CD. It automates the process of building, testing, and deploying code, ensuring faster and more reliable software delivery.

**Kubernetes** =Kubernetes is an open-source container orchestration platform. It helps you deploy, manage, and scale containerized applications across clusters of machines in an efficient, automated way.

Shell= Shell scripting is writing a series of commands in a file to automate tasks in a Unix/Linux shell.

**I worked on** **US Visa Approval Prediction Project and the Problem Statement** was Predict whether a US visa application will be approved or not based on features such as **country, education, experience, salary, and employment details**.

**In Solution**: We implemented a **machine learning classification pipeline** to: 1. Loaded and preprocess data.2. Performed **EDA** and **feature engineering**.3. Trained classification models 4. Selected the best model based on metrics for deployment.

In Deployment :1. **We Created a Docker file to package the application**

**2. Created AWS Resources such as Amazon EC2 and ECR**

**3. we used GitHub Actions for CI/CD:**

We havetwo main jobs in GitHub Actions:

* + **Continuous Integration** includes Builds, tags, and pushes the Docker image to Amazon ECR.
  + **Continuous Deployment** includes Pulls the Docker image and runs it on the EC2 instance.

4. **We Added secrets to GitHub** for secure access such as ACCESS\_KEY, SECRET\_KEY, REGION, ECR\_REPO

5. we created Self**-**Hosted Runner on GitHub to manage deployment workflows.

**6.** Job will be Automatically triggered on every push to the main branch.

**Steps** of CI/CD workflow are:

* 1. Checkout the latest code.
  2. Configure AWS credentials.
  3. Build and push the Docker image to **Amazon ECR**.
  4. Pull the Docker image from **ECR** and run it on **EC2**.

**Sensor-Fault-Detection.**

I worked on a project, Sensor-Fault-Detection to identify if a vehicle fault is due to the Air Pressure System (APS).

* In **Kafka**, the producers are the vehicle sensors that send data. The data goes into topics, where the data is kept. Then, the consumers take the data from these topics and store it in a database. Using **MongoDB** as Database

**Machine Learning Pipeline**

1. **Data Ingestion:** we collect the sensor data and store it for processing.
2. Data Validation: we validate our data
3. **Data Transformation:** We clean and preprocess the data.
4. **Model Training:** We use machine learning model.
5. **Model Evaluation:** we evaluate the accuracy of the model
6. **Model Pusher:** we send model to the deployment pipeline.

In Deployment :1. **We Created a Docker file to package the application**

**2. Created AWS Resources such as Amazon EC2 and ECR**

**3. we used GitHub Actions for CI/CD:**

We havetwo main jobs in GitHub Actions:

* + **Continuous Integration** includes Builds, tags, and pushes the Docker image to Amazon ECR.
  + **Continuous Deployment** includes Pulls the Docker image and runs it on the EC2 instance.

4. **We Added secrets to GitHub** for secure access such as ACCESS\_KEY, SECRET\_KEY, REGION, ECR\_REPO

5. we created Self**-**Hosted Runner on GitHub to manage deployment workflows.

**6.** Job will be Automatically triggered on every push to the main branch.

**Steps** of CI/CD workflow are:

Checkout the latest code.

Configure AWS credentials.

Build and push the Docker image to **Amazon ECR**.

Pull the Docker image from **ECR** and run it on **EC2**.