**General Training**

* **Project, Word file, Ppts**

# Day 1 Course: DBMS

**What is file system?**

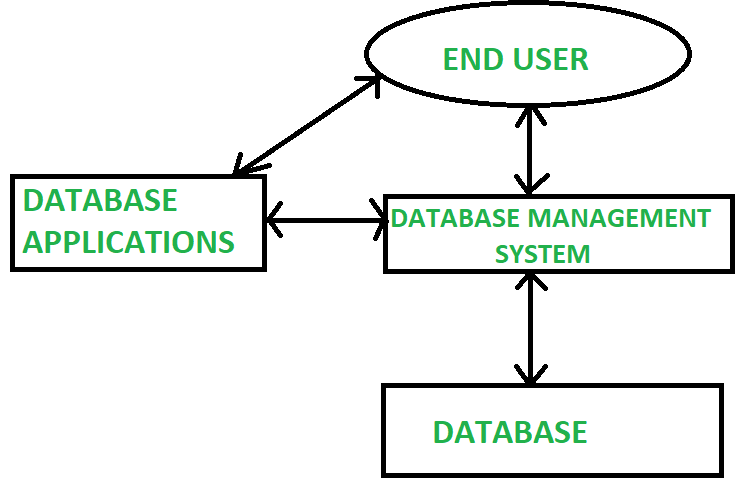


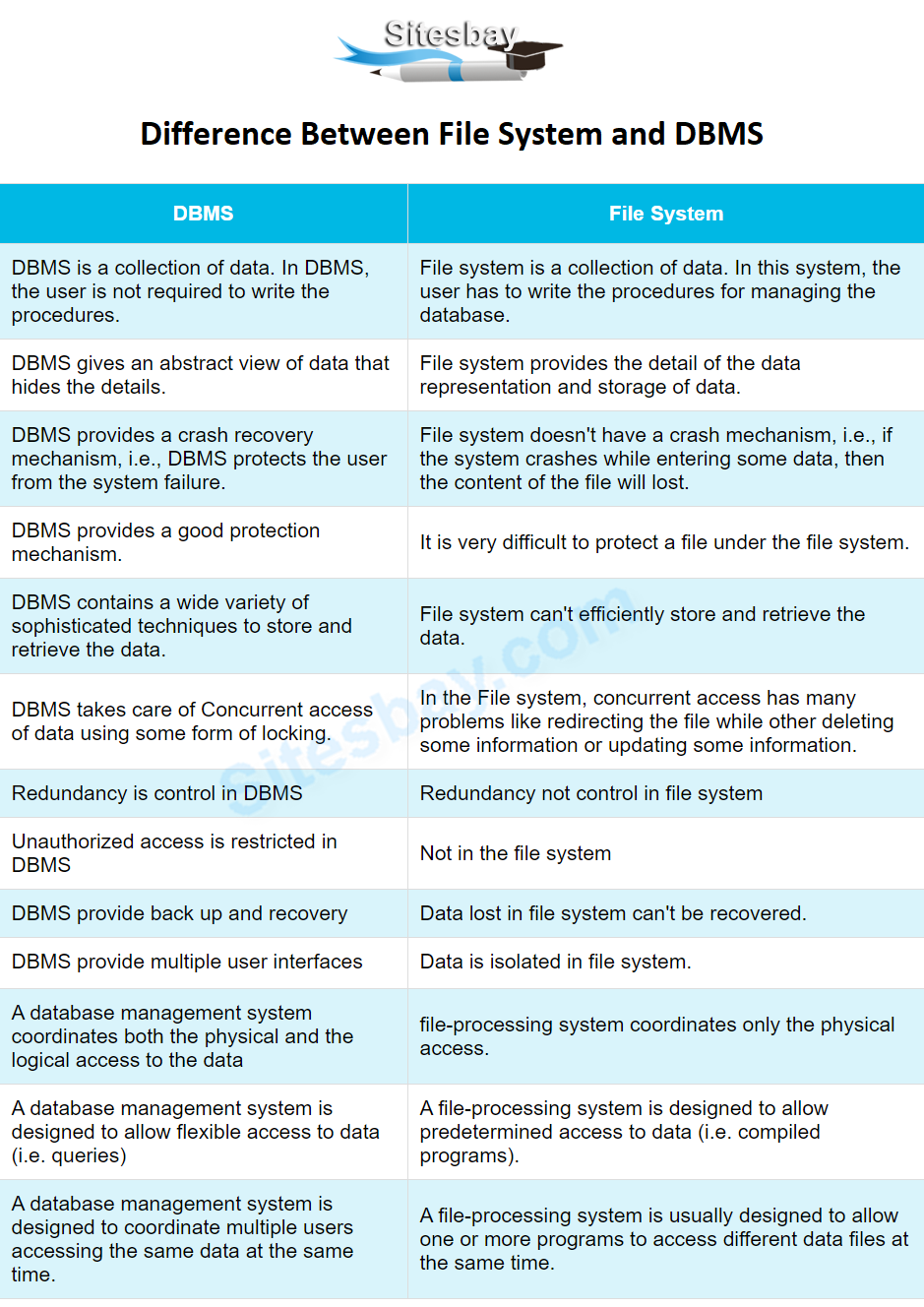
A file system is the method an operating system uses to store, organize, manage, and retrieve data on a storage device. It acts as an interface between the user/applications and the physical hardware, allowing files to be arranged hierarchically in directories and providing the necessary structure to read, write, and delete data efficiently.

The file system is basically a way of arranging the files in a storage medium like a hard disk. The file system organizes the files and helps in the retrieval of files when they are required. File systems consist of different files which are grouped into directories. The directories further contain other folders and files. The file system performs basic operations like management, file naming, giving access rules, etc.

**DBMS ( Database Management System)**

Database Management System is basically software that manages the collection of related data. It is used for storing data and retrieving the data effectively when it is needed. It also provides proper security measures for protecting the data from unauthorized access. In Database Management System the data can be fetched by [SQL](https://www.geeksforgeeks.org/sql/what-is-sql/)queries and relational algebra. It also provides mechanisms for data recovery and data backup.





**What is DDL and DML?**

DDL is a Data Definition Language that is used to define data structures. For example: creating a table, and altering a table are instructions in SQL. DML is a Data Manipulation Language that is used to manipulate data itself. For example: insert, update, and delete are instructions in SQL.

Data Definition Language

* DDL is used to specify a database's structure, which includes its tables, views, indexes, and constraints.
* DDL commands come in the following types: CREATE, ALTER, DROP, RENAME, and TRUNCATE.
* DDL statements only modify the database's schema; they have no direct effect on the data within the database.
* DDL declarations are irreversible and difficult to undo.

Data Manipulation Language

* Inserting, updating, removing, and retrieving data from a database are all possible with DML.
* DML commands come in the following types: SELECT, INSERT, UPDATE, DELETE, and MERGE.
* DML statements have a direct impact on the database's data.
* In the event of an error, data can be recovered thanks to the reversibility of DML statements.

**Sample SQL queries**

Create table People (

name VARCHAR(100) NOT NULL,

age INT,

gender CHAR(1),

location VARCHAR(255)

);

Create table ECOMMERCE (

ORDER\_ID INT,

CUSTOMER\_ID INT,

CITY VARCHAR(100),

STATUS VARCHAR(100),

ORDER\_DATE DATE,

AMOUNT INT

);  
---------------

INSERT INTO ECOMMERCE

VALUES (1, 101, 'PUNE', 'PAID', '2025-01-20', 1400),

(2, 102, 'MUMBAI', 'PAID', '2025-01-08', 2400),

(3, 103, 'DELHI', 'CANCELLED', '2024-07-20', 600),

(4, 104, 'PUNE', 'PAID', '2025-08-29', 900),

(5, 105, 'MUMBAI', 'PAID', '2024-12-19', 1700),

(6, 106, 'PUNE', 'REFUNDED', '2025-05-28', 1000);

---------------------------

INSERT INTO ECOMMERCE

VALUES (1, 101, 'PUNE', 'PAID', 2025-01-20, 1400),

(2, 102, 'MUMBAI', 'PAID', 2025-01-08, 2400),

(3, 103, 'DELHI', 'CANCELLED', 2024-07-20, 600),

(4, 104, 'PUNE', 'PAID', 2025-08-19, 900),

(5, 105, 'MUMBAI', 'PAID', 2024-12-19, 1700),

(6, 106, 'PUNE', 'REFUNDED', 2025-05-28, 1000);

------------------------

DELETE FROM ECOMMERCE

WHERE ORDER\_ID = '6';

-----------------------------

SELECT \* FROM ECOMMERCE

WHERE CUSTOMER\_ID = '104';

-----------------------------------

INSERT INTO ECOMMERCE

VALUES (7, 107, 'MUMBAI', 'PAID', 2024-10-17, 1900);

-------------------------------------

UPDATE ECOMMERCE

SET AMOUNT = '3000'

WHERE ORDER\_ID = '2';

SELECT CITY, SUM(AMOUNT) AS REVENUE

FROM ECOMMERCE

WHERE STATUS = 'PAID'

GROUP BY CITY;

--------------------------------------

SELECT CITY, SUM(AMOUNT) AS REVENUE

FROM ECOMMERCE

WHERE STATUS = 'PAID'

GROUP BY CITY

HAVING SUM(AMOUNT) >2500;

---------------------------------------------

SELECT CITY, AMOUNT AS REVENUE

FROM ECOMMERCE

WHERE STATUS = 'PAID'

--GROUP BY CITY

LIMIT 4;

---------------------------------

INSERT INTO CUSTOMER

VALUES (101, 'RAJ', 'RAJ@MAIL.COM'),

(102, 'MEERA', 'MEERA@MAIL.COM'),

(103, 'ARJUN', 'ARJUN@MAIL.COM');

----------------------------------------------

SELECT E.ORDER\_ID,C.NAME,E.AMOUNT

FROM ECOMMERCE E

JOIN CUSTOMER C ON E.CUSTOMER\_ID = C.CUSTOMER\_ID

WHERE E.STATUS='PAID';

----------------------------------------

SELECT E.ORDER\_ID,C.NAME,E.AMOUNT

FROM ECOMMERCE E

JOIN CUSTOMER C ON E.CUSTOMER\_ID = C.CUSTOMER\_ID

WHERE E.STATUS='PAID';

----------------------------------------

SELECT \* FROM ECOMMERCE

WHERE AMOUNT BETWEEN 1000 AND 2000;

------------------------------------------

SELECT \* FROM ECOMMERCE

WHERE CITY IN('MUMBAI','DELHI');

------------------------------------------

SELECT \* FROM ECOMMERCE

WHERE CITY LIKE 'M%';

------------------------------------------

SELECT E.ORDER\_ID,C.NAME,E.CITY,E.AMOUNT

FROM ECOMMERCE E

INNER JOIN CUSTOMER C

ON E.CUSTOMER\_ID = C.CUSTOMER\_ID;

-----------------------------------------

SELECT E.CUSTOMER\_ID,C.NAME,E.CITY,E.AMOUNT

FROM ECOMMERCE E

LEFT JOIN CUSTOMER C

ON E.CUSTOMER\_ID = C.CUSTOMER\_ID;

-----------------------------------------

SELECT E.CUSTOMER\_ID,C.NAME,E.ORDER\_ID,E.AMOUNT

FROM CUSTOMER C

LEFT JOIN ECOMMERCE E

ON E.CUSTOMER\_ID = C.CUSTOMER\_ID

UNION

SELECT E.CUSTOMER\_ID,C.NAME,E.ORDER\_ID,E.AMOUNT

FROM CUSTOMER C

RIGHT JOIN ECOMMERCE E

ON E.CUSTOMER\_ID = C.CUSTOMER\_ID;

-----------------------------------------

SELECT (ORDER\_DATE) AS ORDER\_DAY,

COUNT(\*) AS ORDERS,

SUM(AMOUNT) AS REVENUE

FROM ECOMMERCE

WHERE STATUS= 'PAID'GROUP BY DATE(ORDER\_DATE)

ORDER BY ORDER\_DAY;

SELECT CUSTOMER\_ID,COUNT(\*) AS PAID\_ORDERS

FROM ECOMMERCE

WHERE STATUS= 'PAID'

GROUP BY CUSTOMER\_ID

HAVING COUNT(\*)>=10

ORDER BY PAID\_ORDERS DESC;

-----------------------------------------

**Rules for using group by and order by**The GROUP BY and ORDER BY clauses in SQL serve distinct purposes and follow specific rules when used in a query.

**Rules for GROUP BY**

* **Purpose:**

The GROUP BY clause is used to group rows that have the same values in one or more specified columns into a summary row. It is typically used with aggregate functions (e.g., COUNT(), SUM(), AVG(), MIN(), MAX()) to perform calculations on each group.

* **Placement:**

It must be placed after the FROM clause and the WHERE clause (if present).

**Rules for ORDER BY**

* **Purpose:**

The ORDER BY clause is used to sort the result set of a query in ascending (ASC, default) or descending (DESC) order based on one or more specified columns or expressions.

* **Placement:**

It must be placed after all other clauses in the SELECT statement, including FROM, WHERE, GROUP BY, and HAVING.

-----------------------------------------

Create table EMPLOYEE (

EMP\_ID INT,

NAME VARCHAR(100),

DEPT\_ID INT,

SALARY INT

);

-----------------------------------------

INSERT INTO EMPLOYEE

VALUES (1, 'JOHN', 101, 60000),

(2, 'ALICE', 102, 24000),

(3, 'BOB', 101, 50000),

(4, 'SARAH', 103, 90000),

(5, 'DAVID', 102, 70000);

-----------------------------------------

CREATE TABLE DEPARTMENT (

DEPT\_ID INT PRIMARY KEY,

DEPT\_NAME VARCHAR(100)

);

-----------------------------------------

INSERT INTO DEPARTMENT

VALUES ( 101, 'IT'),

( 102, 'HR'),

( 103, 'FINANCE');

-----------------------------------------

SELECT \*

FROM EMPLOYEE

WHERE DEPT\_ID IN(SELECT DEPT\_ID FROM DEPARTMENT WHERE DEPT\_NAME IN('IT','FINANCE'));

SELECT DEPT\_ID, MAX(SALARY) AS HIGHEST\_SALARY

FROM EMPLOYEE

GROUP BY DEPT\_ID;

-----------------------------------------

SELECT E.NAME,E.SALARY,D.DEPT\_NAME

FROM EMPLOYEE E

JOIN( SELECT DEPT\_ID, MAX(SALARY) AS HIGHEST\_SALARY

FROM EMPLOYEE

GROUP BY DEPT\_ID) TEMP

ON E.DEPT\_ID = TEMP.DEPT\_ID AND E.SALARY=TEMP.HIGHEST\_SALARY

JOIN DEPARTMENT D ON E.DEPT\_ID=D.DEPT\_ID;

-----------------------------------------

CREATE VIEW EMPLOYEE\_SALARIES AS

SELECT NAME, SALARY

FROM EMPLOYEE;

-----------------------------------------

SELECT \* FROM EMPLOYEE\_SALARIES ;

-----------------------------------------

CREATE VIEW EMPLOYEE\_DEPT AS

SELECT E.NAME, E.SALARY,D.DEPT\_NAME

FROM EMPLOYEE E

JOIN DEPARTMENT D ON E.DEPT\_ID = D.DEPT\_ID;

-----------------------------------------

SELECT \* FROM EMPLOYEE\_DEPT;

SHOW FULL TABLES WHERE TABLE\_TYPE ='VIEW';

DROP VIEW EMPLOYEE\_SALARIES;

-----------------------------------------

CREATE INDEX IDX\_NAME ON EMPLOYEE(NAME);

SELECT \* FROM EMPLOYEE WHERE NAME='SARAH';

-----------------------------------------

**NORMALIZATION**

1 NF – NO MULTIPLE VALUE IN SINGLE COLUMN

2NF – NO PARTIAL DEPENDENCY

3 NF – NO TRANSITIVE DEPENDENCY

BCNF – MUST HAVE CANDIDATE KEY

(CANDIDATE KEY- It refers to a **minimal set of attributes** (columns) in a table that can **uniquely identify each row** (record).  
**Key Characteristics of a Candidate Key:**

* **Uniqueness**: No two rows can have the same value for a candidate key.
* **Minimality**: It contains **no redundant attributes**—meaning if you remove any attribute from it, it will no longer uniquely identify rows.
* **Multiple Possibilities**: A table can have **more than one candidate key**, but only **one** is chosen as the **primary key**.
* **Subset of Super Keys**: Every candidate key is a super key, but not all super keys are candidate keys.)

# Day 2 PL/SQL

**What is PL/SQL?**

PL/SQL (Procedural Language/Structured Query Language) is Oracle Corporation's procedural extension to SQL and the Oracle relational database. It integrates the data manipulation capabilities of SQL with the procedural programming features of a language.

Key aspects of PL/SQL include:

* **Procedural Constructs:**

PL/SQL adds features like variables, constants, conditional statements (IF-THEN-ELSE), loops (FOR, WHILE), and error handling (EXCEPTION blocks) to standard SQL.

* **Integration with SQL:**

It allows the seamless embedding of SQL statements within PL/SQL code, enabling developers to perform database operations and manipulate data directly within the procedural logic.

* **Program Units:**

PL/SQL supports the creation of various program units, including:

* + **Procedures:** Named blocks of code that perform specific actions and can accept parameters.
  + **Functions:** Similar to procedures but designed to return a single value.
  + **Packages:** Collections of related procedures, functions, variables, and other PL/SQL constructs, promoting modularity and reusability.
  + **Triggers:** PL/SQL blocks that automatically execute in response to specific database events (e.g., INSERT, UPDATE, DELETE on a table).

**PL/SQL Introduction**

PL/SQL (Procedural Language/SQL) is Oracle’s extension of SQL that adds procedural features like loops, conditions, and error handling. It allows developers to write powerful programs that combine SQL queries with logic to control how data is processed. With PL/SQL, complex operations, calculations, and error handling can be performed directly within the Oracle database, making data manipulation more efficient and flexible.

PL/SQL allows developers to:

* Execute SQL queries and DML commands inside procedural blocks.
* Define variables and perform complex calculations.
* Create reusable program units, such as procedures, functions, and triggers.
* Handle exceptions, ensuring the program runs smoothly even when errors occur.

![A diagram of a software code

AI-generated content may be incorrect.](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAYABgAAD/4RDgRXhpZgAATU0AKgAAAAgABAE7AAIAAAAHAAAISodpAAQAAAABAAAIUpydAAEAAAAOAAAQyuocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEFyY2hpdAAAAAWQAwACAAAAFAAAEKCQBAACAAAAFAAAELSSkQACAAAAAzE5AACSkgACAAAAAzE5AADqHAAHAAAIDAAACJQAAAAAHOoAAAAIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAyMDE4OjA0OjAzIDEyOjU0OjU1ADIwMTg6MDQ6MDMgMTI6NTQ6NTUAAABBAHIAYwBoAGkAdAAAAP/hCxlodHRwOi8vbnMuYWRvYmUuY29tL3hhcC8xLjAvADw/eHBhY2tldCBiZWdpbj0n77u/JyBpZD0nVzVNME1wQ2VoaUh6cmVTek5UY3prYzlkJz8+DQo8eDp4bXBtZXRhIHhtbG5zOng9ImFkb2JlOm5zOm1ldGEvIj48cmRmOlJERiB4bWxuczpyZGY9Imh0dHA6Ly93d3cudzMub3JnLzE5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMiPjxyZGY6RGVzY3JpcHRpb24gcmRmOmFib3V0PSJ1dWlkOmZhZjViZGQ1LWJhM2QtMTFkYS1hZDMxLWQzM2Q3NTE4MmYxYiIgeG1sbnM6ZGM9Imh0dHA6Ly9wdXJsLm9yZy9kYy9lbGVtZW50cy8xLjEvIi8+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczp4bXA9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8iPjx4bXA6Q3JlYXRlRGF0ZT4yMDE4LTA0LTAzVDEyOjU0OjU1LjE4NTwveG1wOkNyZWF0ZURhdGU+PC9yZGY6RGVzY3JpcHRpb24+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iPjxkYzpjcmVhdG9yPjxyZGY6U2VxIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpsaT5BcmNoaXQ8L3JkZjpsaT48L3JkZjpTZXE+DQoJCQk8L2RjOmNyZWF0b3I+PC9yZGY6RGVzY3JpcHRpb24+PC9yZGY6UkRGPjwveDp4bXBtZXRhPg0KICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICA8P3hwYWNrZXQgZW5kPSd3Jz8+/9sAQwAHBQUGBQQHBgUGCAcHCAoRCwoJCQoVDxAMERgVGhkYFRgXGx4nIRsdJR0XGCIuIiUoKSssKxogLzMvKjInKisq/9sAQwEHCAgKCQoUCwsUKhwYHCoqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioq/8AAEQgBLAHHAwEiAAIRAQMRAf/EAB8AAAEFAQEBAQEBAAAAAAAAAAABAgMEBQYHCAkKC//EALUQAAIBAwMCBAMFBQQEAAABfQECAwAEEQUSITFBBhNRYQcicRQygZGhCCNCscEVUtHwJDNicoIJChYXGBkaJSYnKCkqNDU2Nzg5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6g4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2drh4uPk5ebn6Onq8fLz9PX29/j5+v/EAB8BAAMBAQEBAQEBAQEAAAAAAAABAgMEBQYHCAkKC//EALURAAIBAgQEAwQHBQQEAAECdwABAgMRBAUhMQYSQVEHYXETIjKBCBRCkaGxwQkjM1LwFWJy0QoWJDThJfEXGBkaJicoKSo1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoKDhIWGh4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uLj5OXm5+jp6vLz9PX29/j5+v/aAAwDAQACEQMRAD8A+kaKKKACiiquoapp+k24n1W+trGEttElzMsak+mWIGaALVFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVR/wALH8Ef9Dl4f/8ABpB/8VQB0lFc3/wsfwR/0OXh/wD8GkH/AMVVzTfF/hvWbhbfSPEOlX8zZ2x2t7HKxxyeFYmgDYooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAK8sRxrWt6jq16Fnmjvri0tywyLeOGVotqZ6ZaMsfUnrgKB6nXlWgDNvqP/Ya1P/0unrzMzbVFW7/5nj5w2sOkur/Rl/bRtqXbRtr5zlPk+Ui20bal21y/jXx7pPgSG0k1iK6m+1syxpaorN8uMk7mHHIqo05TfLFal06MqkuWCuzo9tG2uC8P/Gnwr4h1iDTYlvrOa4YJE13EoV2PRcqzYJ966SDxdp9z44ufCsMVw19a24uJZNq+UqnbxndnPzDtWksNUi7Sj5lywtWDalG1lf5Gzto21LtrmLrx/odlY397dNNHbWV39iEnl7vtMwHKxAElsHjoOQfTNZxpyl8KIjRlP4Vc6LbRtrlNO+JOk3mr2+m6hYatolzdtttl1WzMAnPopyR374rT8U+MdD8HWaT65diIyZ8qFBukkx1wo/mcD3qnQmmlbcr6tU5lDl1ZsbaNtcZ4X+LfhfxXqi6dZyXNpdyHEUd3GF80+gKkjPsSDW1f+LtP0/xnp3hmSK4kv9QjaWMxqpRFAY5YlgR909AacsPUjLlcdRyw1WMnGUdUr/I2dtG2pdtZfiXX7Pwr4fudY1JZGt7cLuWEAu2WCgAEgdT61moNuyM40nOSjFasv7aNteZ2/wAf/CM06xyW2q26scGSSBCq/Xa5P5Cuw8ReNNK8OaBZ6xOJry2vZI47f7Iqs0hcEqRkgYwPWtpYWrFpOO5tLB1oSUZRd2bm2jbUoGQDjHtRtrDlOflIttG2pdtYeo+K7DTfFumeHZYriS91JHeIxqpRFUEksSQR0PQHpVRpuTsio0pSvZbGvto21Lto21PKTykW2oLzT7a/h8q8hWVc5GRypHIZT1UggEEcggEVc20baaTTuhpNO6Oh8Dalc6p4PtZr6bz7iKSe1kmPWQwzPDuPA5Pl5OBjJOK6CuT+Gf8AyJR/7Cmpf+l09dZX2UG3BNn6BTbcE32CiiirLCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAK8u8PLm11H/sNan/AOl09eo15j4cGbPUP+w1qf8A6XT152Yq9Jev+Z5Oaq9Bev6M0ttG2pdtG2vC5T5vlItteKfE3WruH4yaENN0e41t9ItjcmztgxYsxPPyqxGMIc4r3DbXK6Z4HFj8RtV8WTX/ANokv4Fgjt/J2+QoC/xbjnOwdhW+HcYT5pdn/kdOGlCm5Skr6Wt6/wDAucEyeKvid4p0OXU/C0vh7S9KuRcySXRPmOQQQo3KpwcdhjuT0rR+F4Or+PvG3iBgCj3gtIW/2UJ/oEr1SSNmidUbYxUgNjOD64rx+w+Buu6VHJHpfxG1CyjkcyOltbvGrMerELMMngc1vCpCUXF+6rWW73d3+R0RqU6lOUJNR2S3el7v8TvPG2szaL4dYacA2p30i2lgnrM/AP0UZY/SuS8UeAoLrwro3hnQdZtbXXNKcXdsk0oDzuMlnK8nliW3YPNb3hz4f3ml6hp93rviO5119Pjl8j7SjZEkh5kyzseEAUDtye9WfFHgRdd1m01zS9Un0bWrOMxxXkMayKUOfldG4YcnuOtZx5abSi/O/wCX9eZnTlGnJKMtr620b/yt+bOO8P8AizV7jxfp/hX4neG7U6gxMlje+UrqWUE7h1GeD8ykYOBim6Db2fiL42eKdS17ZINCVIrSKfBSJecvg+m0nPbdn0rr9F8Bz2/iGHXvE+uz6/qdtG0ds726QRwA9Ssa8biOM5qh4t+FMOv68dd0TWrzw/qzoEluLXOJB05AZTnAA4PYcVpz0+bTS6tdXte/Remhrz0eaUU+W6Sur23u7dbM5n4oX+keIb7wafD80N3qd1qEcltNAQWEIPJJ6gbsdem0+hq/ooOtftFa3d4DQ6PYLbIfRmx/jJS+C/hNeeCfHkWoxz22rWktu4murhSk0EnqgyQd2ep5xnp3iuPgvrJ1/U9U0zx7eabJqNw00q2tu6ZyxIBKzDOMkCrTpRXIpaWevq/8i26Ki6anpayevV3fp2PVdteXfHieeTwnpujWS77nVNQSNIwcb8A8c/7RSuw8E+FdS8L2NzDq/iS81+WaQMst0X/dqBjaAzt3yetZfxB+HFz441DTLu2199IfTdzRGO3LtvJB3Bg64I2iualGEKybei6nLhvZ0q6k3or669tDznxjq/xFj8HPZ+L9AsdM0FvLhuLiyVJZETIAAUzEZ4Azx9RW54otrO7174beGdKZnsFK3a7/ALzRxqpUn6gNV1Pgne380aeK/HOra3Yo4c2shdQxHu0jY/AZ9660eBYx8RbbxP8Aa1EdnYfYreyWHAjHPzb93oSMY711OrTVrNaXeidttN/M6pVaSSUGtFLZNK7VupyGnyav8SfGGvw/27qOjaLo8/2WGPTJBDLLICQWZ8E4+U8e49DnN0/4i6vongHxRJeT/wBpXWi35sbO7lUfvcsVBbHUjBPvwK6+8+G15Brmoah4U8T3Ogpqj7723S2SZXbnLIWwUJyTkZ61aX4XaCngGbwqvnfZ5282S5LAzPLkHzCcYzwO2McVnzUuVJ7aaW9L/fqTz0LpPVXWltkt7+vzuYll4O8YxRaVrFr4vu59SkZJL+2vpc2jIRlkSNV+U9s/jx0rifHHjK40b40anf6dZNe3WmaZ9ng+XcsLEBmkbHYB2/HFel6R8OruHXLLVPEvia61yTTVK2URgWCOIYxlgpO88Dk+gzU2k/DqCxuvFFxf3gvn8RO3mZg2eTGdwCD5jnG7rx0FNVIRk5PXR7K2/T7gp1acJN1LS0ttbrr9y/yJPAGlTWvhuC/vNbvNZudRjW4eeeZmjG4bgsaE4QDPYf0A6nbWP4N8OTeFfCtpo1xqH9oG1DKk/leX8pJIGNx6Zx1rd21y1bSqNrY8+ouabd7kW2jbUu2jbWXKRylv4af8iWf+wpqX/pfPXWVynw2/5E1v+wrqf/pfPXV19XT+BH21P4F6BRRRVmgUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABXm9w0fhLUL221lxbWdzezXVreOu2FxNIZGRnxhXDuwwx5GCM/MF9IorGtSjVjyswr0I14csjy7/hOfCP8A0NWif+DGL/4qj/hOfCP/AENWif8Agxi/+Kr1GiuP+z4dzg/suH8zPLv+E58I/wDQ1aJ/4MYv/iqP+E58I/8AQ1aJ/wCDGL/4qvUaKP7Ph3D+y4fzM8u/4Tnwj/0NWif+DGL/AOKo/wCE58I/9DVon/gxi/8Aiq9Roo/s+HcP7Lh/Mzy7/hOfCP8A0NWif+DGL/4qj/hOfCP/AENWif8Agxi/+Kr1Gij+z4dw/suH8zPLv+E58I/9DVon/gxi/wDiqP8AhOfCP/Q1aJ/4MYv/AIqvUaKP7Ph3D+y4fzM8u/4Tnwj/ANDVon/gxi/+Ko/4Tnwj/wBDVon/AIMYv/iq9Roo/s+HcP7Lh/Mzy7/hOfCP/Q1aJ/4MYv8A4qj/AITnwj/0NWif+DGL/wCKr1Gij+z4dw/suH8zPLv+E58I/wDQ1aJ/4MYv/iqP+E58I/8AQ1aJ/wCDGL/4qvUa5Lw7/wAlK8Y/9uP/AKJNZzwUIyir7u34N/odFHJYVIVJ879xX2396Mf1uc3/AMJz4R/6GrRP/BjF/wDFUf8ACc+Ef+hq0T/wYxf/ABVeo0Vp/Z8O5z/2XD+Znl3/AAnPhH/oatE/8GMX/wAVR/wnPhH/AKGrRP8AwYxf/FV6jRR/Z8O4f2XD+Znl3/Cc+Ef+hq0T/wAGMX/xVH/Cc+Ef+hq0T/wYxf8AxVeo0Uf2fDuH9lw/mZ5d/wAJz4R/6GrRP/BjF/8AFVJH4t0K7wmlalb6tO52x2+myLcSOfQBCcdRycADkkDJr02in/Z8O4LK6d9ZMxvCWkz6L4YtrS92fameW4nEfKrJLI0rqDgZAZyM4GcZrZoor0ErKyPVSSVkFFFFMYUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFcl4d/wCSleMf+3H/ANEmutrkvDv/ACUrxj/24/8Aok1z1fjp+v8A7az0MJ/BxH+Bf+nIHW0UUV0HnhRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABXJeHf+SleMf+3H/wBEmutrkvDv/JSvGP8A24/+iTXPV+On6/8AtrPQwn8HEf4F/wCnIHW0UUV0HnhRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABXJeHf+SleMf8Atx/9EmutrzuLX/7D+JXir/iU6pqPnfY/+Qfbeb5eIf4uRjOePoa5cRJRlBvv/wC2s9TL6c6tOvCCu3Bf+lwPRKK5L/hPv+pT8Uf+C3/7Kj/hPv8AqU/FH/gt/wDsqr6zS7mf9m4r+T8V/mdbRXJf8J9/1Kfij/wW/wD2VH/Cff8AUp+KP/Bb/wDZUfWaXcP7NxX8n4r/ADOtorkv+E+/6lPxR/4Lf/sqP+E+/wCpT8Uf+C3/AOyo+s0u4f2biv5PxX+Z1tFcl/wn3/Up+KP/AAW//ZUf8J9/1Kfij/wW/wD2VH1ml3D+zcV/J+K/zOtorkv+E+/6lPxR/wCC3/7Kj/hPv+pT8Uf+C3/7Kj6zS7h/ZuK/k/Ff5nW0VyX/AAn3/Up+KP8AwW//AGVH/Cff9Sn4o/8ABb/9lR9Zpdw/s3Ffyfiv8zraK5L/AIT7/qU/FH/gt/8AsqP+E+/6lPxR/wCC3/7Kj6zS7h/ZuK/k/Ff5nW0VyX/Cff8AUp+KP/Bb/wDZVo6J4n/tq9e3/sTWdP2xmTzb+08pDggbQcnnnOPQGqjXpydkyKmAxNOLnKOi81/mblFeW+PfCWieM/i/4d0vxLZfbbMaTeSiLzXj+cSRAHKEHue9U/EvgbSvhTo48WeA/tOljTpo3vbIXcssN5AzhXVldj8wByCPSto62v1/zscTvey/rS569RXG+JPHV/Ya0dF8JeGbjxLqsUK3FzEl1HbRW8bEhd8j8bjgkKB0FOtfiNpw8I6nres2lzpUujuYtQsJgGlhlwCqDBw+7cu0g4O4dKOlw6pHYUV5/pnxJ1b+3LC08V+C73w9Z6rMILC9lu4pg8hBZUkReYmIHAOeeKoeDdZm0Lwr4lu7XSr3VrhvE97HFaWUe53ZpcDJPCKOpY8AUf1+KVvxFfb1/Ru/4Hp9FcDovxE1mTxdY+H/ABj4On8Oz6mspsJRfxXaTGNdzAlPunHPepNH+IWpa94nutN0vwtLLZafqE1lqGpNeokduYyQCFIDOSACQv3dw5NH9foPW1/6/rQ7qivN2+KeqzI+q6V4G1K+8LxFi+rpcxK7opIaSO3J3unBIPGR2r0Kzu4NQsYLyzkWW3uI1likXoysMg/kaN1cHo7E1FFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFcl4d/5KV4x/7cf/AESa62uS8O/8lK8Y/wDbj/6JNc9X46fr/wC2s9DCfwcR/gX/AKcgdbRRRXQeeFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFAHmnjrVLvw38UNB11PDuua1aR6ZdW8g0exNwyO7xFc8gDhT3qpq+raz8VLOHw9ZeEtb0PSZ7iNtTvdbgW2IhRw5jjTcSxYqBnoBmvVqKI6W8v8AO4Pe6/rSx4d8QvAmlp8R77xB4m8E6p4t0rULeFYjpDyNPZyxjaVMSOm5WGDnnBB9au2nw9ju/hTrln4U8IS+F5ry4iubS0vr55JbkwsrqZAzHyi2Cu3cfU17JRQtFZf1rcJWk7v+tLHn1h488Ta/qllp2meCdV0l1nX+0rrWYAlvFEPviJlfMrHoCOOckVy91YeKrHwNq66ZZ6xCkniy5mvY9OXy7yWyZyS0BbBJPy4K9RnBr2mij+vxT+7QPL+tmv1PnzRNFnuPi34S1vSfB3iuw0+CWdLq/wBekllmZmiYLuQuxjUH+LgEt9K9K8AaVdw2Xi6C+tp7T7Zr968RljKF43Iw656qexHBruaKN1y+TX3tP9B3eny/C/8AmfNGn/DnQNEtF0TxD8I9d1nXoi0aX9jcyiyu+flkaUSgRZGMjbx6dq+iNCsI9K8PafYQWi2UdtbpEtskxlEICgbA7csB0yetX6Kd3axNtbhRRRSGFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVyXh3/kpXjH/tx/8ARJrra5Lw7/yUrxj/ANuP/ok1z1fjp+v/ALaz0MJ/BxH+Bf8ApyB1tFFFdB54UUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVyXh3/kpXjH/ALcf/RJrra5Lw7/yUrxj/wBuP/ok1z1fjp+v/trPQwn8HEf4F/6cgdbRRRXQeeFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRUdxcQ2ltJcXUscMEKF5JZGCqigZJJPAAHesKD4geDbmdIbbxboU0rnCpHqULMx9AA3NAdLnQ0UUUAFFFQS39pBeW9pPdQxXNzu8iF5AHl2jLbVPLYBycdKAJ6KKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigArzuLUdYsPiV4q/sTQv7W3/Y/N/0xIPKxDx94c5yenTHvXolcl4d/5KV4x/7cf/RJrlxCcpQSdtf/AG19z1MvnGFOvKUVJci0d7fHDs0/xD/hIvGP/Qjf+VeH/Cj/AISLxj/0I3/lXh/wrraKr2U/+fj/APJf8jP63R/6B4ffU/8Akzkv+Ei8Y/8AQjf+VeH/AAo/4SLxj/0I3/lXh/wrraKPZT/5+P8A8l/yD63R/wCgeH31P/kzkv8AhIvGP/Qjf+VeH/Cj/hIvGP8A0I3/AJV4f8K62ij2U/8An4//ACX/ACD63R/6B4ffU/8Akzkv+Ei8Y/8AQjf+VeH/AAo/4SLxj/0I3/lXh/wrraKPZT/5+P8A8l/yD63R/wCgeH31P/kzkv8AhIvGP/Qjf+VeH/Cj/hIvGP8A0I3/AJV4f8K62ij2U/8An4//ACX/ACD63R/6B4ffU/8Akzkv+Ei8Y/8AQjf+VeH/AAo/4SLxj/0I3/lXh/wrraKPZT/5+P8A8l/yD63R/wCgeH31P/kzkv8AhIvGP/Qjf+VeH/Cj/hIvGP8A0I3/AJV4f8K62ij2U/8An4//ACX/ACD63R/6B4ffU/8Akzkv+Ei8Y/8AQjf+VeH/AArR0TVdfvr14tZ8N/2VAIyyzfbo5tzZHy7VGRwSc+1blFVGnJO7m393+RFTE0pRcVRivNOd/wAZtfgYHj3/AJJx4k/7BVz/AOimrxbT9Z8HL4EtI5/gprd3ONPRXvB4ajCSt5YzJ5wO7BPO/r3r3DxdY3GqeCtbsLGPzbm60+eGGPcF3O0bBRk8DkjrXA6F4g+KGmaDp+kL8L4la1tY7YXU3iG32ZVAu8qoJxxnArS1+Zen6nK3ZRfa/wD7aJZeMR4C+C/hb/T7DVL69SO0tbie9EVsGIJy8rdERRg9yQBwTVnwd8Tru88TRaF4m1DwtfT3cUkttdeG9Q8+MbBuZJEY7lO3JB6HBqjefCrU/wDhXeiWpj0vVNZ0rUJNSls7uPNnctKzmSEAjhfn+UkdVHTte8I+HL6fXBd6j8NPDHg+0hhkRng8me7kZhtykkSqI1wWznJOcVbbcpN+f5f5maSUUv63/wAiuPH/AI8v9BuvF2keH9JfwysMstvBLPJ9uljXOJsD5NvG7ZkEjoelaVn4qn1PVvhxPeadpz3GuafPcyzmAmS2byEciFicoCWwc5yAK42TUPGfh/4U32haRZaLqeg2tlPDbeKF1ZPKFuAQAYgCTIB8vBxnFdR4f8PapMPhZfpan7NpWkyJeOXUGIvbRqowTk5II4Bx3pxtr8vXaX/AG+l99f0Ktv41+JWu/wBuS+GtG8P/AGXR9SubUyX0sqtdLG3CxqpIDY6sxAJPQYrv/CPiBPFfg/S9djh8gX9usxiLbthI5XPfBzzWP4J0LUdI0bxBBqNv5Ml5rF7cwLvVt8cjZRuCcZHY8jvUvwv0XUPDvwx0PSdZt/s19a25SaLer7TuJxlSQevY0o/D8o/lqTJvn02vL8HodZRRRSKCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACuS8O/wDJSvGP/bj/AOiTXW1yXh3/AJKV4x/7cf8A0Sa56vx0/X/21noYT+DiP8C/9OQOtoooroPPCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAOJf4O/D+TXjrD+F7Nrwv5hyX8stnOfKzs6/7NdsBgYHAooo2Vg3dwooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACuS8O/wDJSvGP/bj/AOiTXW1yXh3/AJKV4x/7cf8A0Sa56vx0/X/21noYT+DiP8C/9OQOtoooroPPCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACvO4vFOj+GviV4q/tu8+zfaPsflfunfdth5+6Dj7w616JXJeHf8AkpXjH/tx/wDRJrlxHM5Q5d7/APtr9D1MvdNU67qJtci2dn8cOtn+Qf8AC0fB3/QY/wDJWb/4ij/haPg7/oMf+Ss3/wARXW0VXLX/AJl/4C//AJIz9pgP+fc//A1/8rOS/wCFo+Dv+gx/5Kzf/EUf8LR8Hf8AQY/8lZv/AIiutoo5a/8AMv8AwF//ACQe0wH/AD7n/wCBr/5Wcl/wtHwd/wBBj/yVm/8AiKP+Fo+Dv+gx/wCSs3/xFdbRRy1/5l/4C/8A5IPaYD/n3P8A8DX/AMrOS/4Wj4O/6DH/AJKzf/EUf8LR8Hf9Bj/yVm/+IrraKOWv/Mv/AAF//JB7TAf8+5/+Br/5Wcl/wtHwd/0GP/JWb/4ij/haPg7/AKDH/krN/wDEV1tFHLX/AJl/4C//AJIPaYD/AJ9z/wDA1/8AKzkv+Fo+Dv8AoMf+Ss3/AMRR/wALR8Hf9Bj/AMlZv/iK62ijlr/zL/wF/wDyQe0wH/Puf/ga/wDlZyX/AAtHwd/0GP8AyVm/+Io/4Wj4O/6DH/krN/8AEV1tFHLX/mX/AIC//kg9pgP+fc//AANf/Kzkv+Fo+Dv+gx/5Kzf/ABFaOieM9A8RXr2mjX/2mdIzKyeTImFBAzllA6kVuUVUY1r+9JW9H/mRUng3FqFOSfnNNfdyL8wrwDSIdE1q41m58T/F7XdDvU1i8hFjH4mS2WONZmCYjfJAx+HpXv8AXzx4c8VfCLSX1y18dW+kyauuuXxdrrRmuX2mdtvziJs8ds1qvi+X6o4nt8/0Z6R4eutH8H+CtU1fSfEuteOLKCTzJpTqMeoSxbQNyoQVAAB3Fc5xzW4fHOjyapomn6e0uoz61Ebi3+yBWEcAGfOkJI2pyB3JJwAea5zQ/iJ8N7Pwdq2o+Dfsi6dYMGnt7HT2tvMlcYRQhRdzttCjGe1c38OdMufhz4qjHiextrRfFgzaSRZK6fLuZxY5JIC4bK4wCwYc8Ve8ren5bfP+tWhbRuv68/l/WzOv1r4u6Do+rXVlHp+taolg5S/vNN09p7eyYDJEjjpgdcZxWlrnxE0PRNL0+8j+16q+qJ5lhaaXbm4nulwGLIg7AEZJwOa4bw5430f4cW2qeGvGEN7b6m2p3c1tGtjJKNSSWRnRo2RSrEhgpBIwRisTxjp3k6p4Z8Sa9Hr3gnQ20hrWVfDspRtMffvVJNkeQjLjIC8MoFZp3in3t+Tdvv0/pFvd/P8ANf8ADnqGnfEjR9R8NatqwttRtZNHhaa+068tvJu4QFLDMbEDkA4OcH1rKg+NPhqe6tc2msRaddFEj1iSxYWXmN0jMoJGcnBI+XOea4nTLHQZvBnjnWvD2s+Ldejk0KW3Op66+6GUBHYLEXVXJU5zxgbveln8aaZ4g+Dtp4C0rTr1/El5pcFoummykAhyqgTF9uzywPnDA9MVfXbtp63/AMiOi+ev3f5nWzfEC/i+PI8M/wBl61Jp32BVHl2QMQlZwftBfOfKC4Td0DAjHetPR/GGg6N4Evdbu9S1N7CDULmJpNSbzZ2kE7L5aBc5Xdwi9duKyL3UrXw38eNJXWJXiTUtATT7SYxsyzXAnzsyAcHBzzge9cxBa3X/AArnTdXtrKa/h0TxhdX91bW6b5HiW4mViq/xFdwOPY1Ktor99f8At5L8tRyvdv00/wC3b/np6+Z6F4c+J2k+IdaTSJ9N1nQ9QmVntrfWbE2zXSr1MfJDYHOM59qzr3406Ha3eo21vo3iHUZdLupLa9Fhp/nC32HBkZg2AhwcZOeDwKybnxdpfxO8Z+FYfBa3N7FpOoG+vr82skUdsixsvllnAyzFgMDPT2rZ+GSKLbxowUAt4lvtxA68r1pu9m+yb9bNf5/gPT8V+Kl/kjtNG1ey1/RbTVdKmE9neRCWGQAjcp9jyD7Vdrg/gmxb4N6BuOcRyAfTzXrvKqaSk0iY3tqFFFFSMKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAK5Lw7/yUrxj/wBuP/ok11tcl4d/5KV4x/7cf/RJrnq/HT9f/bWehhP4OI/wL/05A62iiiug88KKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAM3xHpH/CQeGNT0fz/ALP9vtZLbztm7y96ld2MjOM9MipNE03+xvD+n6Z5vnfY7aO38zbt37FC5xk4zjpmr1FG1/6/rcN7eX6/8MFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABXJeHf+SleMf+3H/0Sa62uS8O/wDJSvGP/bj/AOiTXPV+On6/+2s9DCfwcR/gX/pyB1tFFFdB54UUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFcV4i+MHgXwprk2j6/rn2S/gCmSH7JO+3coYcqhHQjvRcdmztaK5Twp8TvCHjfUJrLwvq/264gi82RPs00e1cgZy6AdSK6unZk3QUUVnwa5p1zr13osFxu1Czijmnh2MNiPnadxGDnaeh7UhmhRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFcfd6H4nsPFeqar4bm0l4tTWEypfiUNG0alQBs6jvk+uMcZPYUVnUpqpa/Q6cPiJUHLlSaas01dNXT/ADSOS/4uL/1K/wD5MUf8XF/6lf8A8mK62is/Yf3n95v9e/6dQ+7/AIJyX/Fxf+pX/wDJij/i4v8A1K//AJMV1tFHsP7z+8Pr3/TqH3f8E5L/AIuL/wBSv/5MUf8AFxf+pX/8mK62ij2H95/eH17/AKdQ+7/gnJf8XF/6lf8A8mKP+Li/9Sv/AOTFdbRR7D+8/vD69/06h93/AATkv+Li/wDUr/8AkxR/xcX/AKlf/wAmK62ij2H95/eH17/p1D7v+Ccl/wAXF/6lf/yYo/4uL/1K/wD5MV1tFHsP7z+8Pr3/AE6h93/BOS/4uL/1K/8A5MUf8XF/6lf/AMmK62ij2H95/eH17/p1D7v+Ccl/xcX/AKlf/wAmK0dE/wCEr+2v/wAJH/Y32Xyzs+webv35GM7+MYz+OK3KKqNKzvzP7yKmL54uPs4r0WoV47/bvinR/jD42Xwr4O/4SNZDYmZv7UitPJItxgYcHdnnp0xXsVeZ6j4c+I2meP8AXta8FyeF2s9X+zlk1ZrgyKYognSMYHOe57dK26/15HItn/XU6Pwvr3ivVvtv/CUeDv8AhG1hjDQP/akV35x5yMIBtxgdeua80stc+KFz8KU8eN4p0+NLO2kuTpraYjC8jRjkySAgqxAOAgA6e5r0Pw0vxIfUJV8a/wDCLfYGgYJ/ZP2jzfMyMZ8zjbjPv0qnaeBtTt/gXL4Lee0Oovps1oJQ7eTvfdg527sfMP4fwpO6Ta36fiTHdJ7f8MV9a8Q+JPEviDTfDvg3UbfRHn0pdVu9RlthctGjttRI0YhSSc5J7VS8BRa3b/GTxVD4lmt7q8j02xUXVvGY1uEBkw+zJ2k8ggEjIOOK0NS8EeI7OXRta8H6jp9vrthpiabcw36O9rdxgA4JXDKQwJBA71Y8HeEPEWk+ONa8Q+JdSs76XVLS3jItVZFiePdlVUjhBuGCWJPJOKppc6t3f62/C3/DkK7hr2X6X/G53VFFFIsKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigD//2Q==) A green box with white text

AI-generated content may be incorrect.

* Begin and End is mandatory.

**Difference between PL/SQL and SQL**.

**Key Differences:**

|  |  |  |
| --- | --- | --- |
| **Feature** | **SQL** | **PL/SQL** |
| Type | Query language for data manipulation | Procedural language with SQL capabilities |
| Functionality | Data retrieval and modification | Complex logic, error handling, and control flow |
| Execution | Single statement at a time | Multiple statements in a block |
| Procedural Support | No | Yes (Variables, loops, conditions) |
| Error Handling | Limited | Advanced (EXCEPTION blocks) |

In short, **SQL** is the language for managing data, while **PL/SQL** allows you to extend SQL with procedural logic and is useful for more complex tasks.

**1. Simple PL/SQL Block (Hello World)**

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Hello, World!');

END;

**Explanation**:

* BEGIN marks the start of the block.
* DBMS\_OUTPUT.PUT\_LINE is used to display output.
* END; ends the block.

To see output, ensure SET SERVEROUTPUT ON; is enabled in SQL\*Plus or your SQL IDE.

**2. Declaring and Using Variables**

DECLARE

v\_name VARCHAR2(20) := 'John';

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Name is: ' || v\_name);

END;

**Explanation**:

* DECLARE section is used to define variables.
* || is the string concatenation operator.

**Features of PL/SQL**

|  |  |
| --- | --- |
| **Feature** | **Description** |
| Block Structure | Organizes code into logical blocks |
| SQL Integration | Full support for DML/DDL queries |
| Procedural Logic | Supports variables, loops, conditions |
| Exception Handling | Built-in and user-defined errors |
| Cursors | For handling multi-row queries |
| Stored Procedures/Functions | Encapsulated, reusable logic |
| Triggers | Automatic execution on events |
| Packages | Groups procedures/functions logically |
| Performance Features | Bulk operations, reduced network traffic |
| Security | Access control and rule enforcement |
| Object-Oriented Support | Basic OOP features like object types |
| Modularity | Cleaner, testable code |
| Tool Integration | Works with Oracle IDEs and tools |

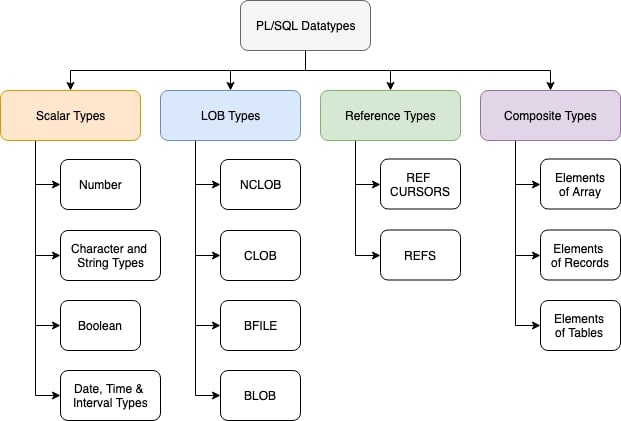
**Why do we need PL/SQL?**

|  |  |
| --- | --- |
| **Reason** | **Description** |
| Add logic to SQL | Supports IF, loops, variables |
| Group multiple statements | Execute SQL as one block |
| Improve performance | Reduces network trips, supports bulk processing |
| Handle errors | Built-in exception handling |
| Write reusable code | Use procedures and functions |
| Automate database tasks | Triggers, background jobs, logs |
| Enforce business rules | Logic enforced at database level |
| Build complete apps in DB | Procedural and OOP support |
| Integrate with Oracle tools | Works with Oracle Forms, Reports, APEX |

**Data Types:**

[**PL/SQL**](https://www.geeksforgeeks.org/plsql/plsql-introduction/)uses many data types corresponding to different types of data including scalar and composite data types and reference data types along with large objects**(LOBs).**

|  |  |
| --- | --- |
| **S.No** | **Category & Description** |
| 1 | **Scalar**  Single values with no internal components, such as a **NUMBER, DATE,** or **BOOLEAN**. |
| 2 | **Large Object (LOB)**  Pointers to large objects that are stored separately from other data items, such as text, graphic images, video clips, and sound waveforms. |
| 3 | **Composite**  Data items that have internal components that can be accessed individually. For example, collections and records. |
| 4 | **Reference**  Pointers to other data items. |



DECLARE

a1 number;

b number;

c1 number;

BEGIN

a1:=10;

b:=9;

-- finding greater number in a & b

IF a1>b THEN

c1:= a1;

ELSE

c1:=b;

END IF;

dbms\_output.put\_line('Maximum no. in a1 & b is: ' || c1);

END;

// for taking input

BEGIN

a1:= :a1;

b:= :b;

for oracle input : &

DECLARE

a1 number;

b number;

c1 number;

d1 number;

e1 number;

f1 number; (f1 number(7,4)) FOR DECIDING HOW MANY DIGITS BEFORE AND AFTER DECIMAL WE NEED IN O/P

BEGIN

a1:= 100;

b:= 0;

-- finding greater number in a & b

-- IF a1>b THEN

-- c1:= a1;

-- ELSE

-- c1:=b;

-- END IF;

c1 := a1+b;

IF a1>b THEN

d1 := a1-b;

ELSE

d1 := b-a1;

END IF;

e1 := a1\*b;

IF b!=0 THEN

f1 := a1/b;

ELSE

dbms\_output.put\_line('Division not possible ');

END IF;

dbms\_output.put\_line('Sum Result: ' || c1);

dbms\_output.put\_line('Difference Result: ' || d1);

dbms\_output.put\_line('Multiplication Result: ' || e1);

dbms\_output.put\_line('Division Result: ' || f1);

END;

-------------------------------------------

**Varchar data type.**

In Oracle databases, both VARCHAR and VARCHAR2 are used to store variable-length character strings, but there are crucial differences, particularly regarding their future behavior and handling of NULL values.

Key Differences:

* **Standard vs. Oracle-Specific:**

VARCHAR is an ANSI SQL standard data type, while VARCHAR2 is an Oracle-specific data type.

* **NULL and Empty String Handling:**
  + VARCHAR is reserved by Oracle to potentially distinguish between NULL and an empty string ('') in the future, as prescribed by the ANSI standard.
  + VARCHAR2 does not distinguish between a NULL and an empty string; both are treated as the same. This behavior is stable and will not change.

DECLARE

school\_name constant varchar2(20) := 'SPM';

BEGIN

dbms\_output.put\_line('My school is: ' || school\_name);

END;

DECLARE

a1 int;

b int;

c1 int;

BEGIN

a1 := :a1;

b := 2;

c1 := MOD (a1,b);

dbms\_output.put\_line(c1);

IF c1!= 0 THEN

dbms\_output.put\_line('The given no. is odd. ');

ELSE

dbms\_output.put\_line('The given no. is even. ');

END IF;

END;

------------------------------------------

DECLARE

a1 int;

b1 int;

-- c1 int;

BEGIN

a1 := :a1;

b1 := MOD(a1,2);

CASE b1

WHEN 0 THEN

dbms\_output.put\_line('The given no. is EVEN. ');

WHEN 1 THEN

dbms\_output.put\_line('The given no. is ODD. ');

END CASE;

END;

------------------------------------------

DECLARE

a1 int;

b1 int;

c1 number;

d1 number;

BEGIN

a1 := 100;

b1 := 58;

c1 := :c1;

CASE c1

WHEN 1 THEN

d1 := a1+b1;

dbms\_output.put\_line('Addition: ' || d1);

WHEN 2 THEN

d1 := a1-b1;

dbms\_output.put\_line('Subtraction: ' || d1);

END CASE;

END;

------------------------------------------

DECLARE

i int;

BEGIN

i := 1;

LOOP

IF i>10 THEN

EXIT;

END IF;

dbms\_output.put\_line(i);

i := i+1;

END LOOP;

END;

------------------------------------------

DECLARE

i int;

a1 number;

BEGIN

i := 1;

a1 :=1;

LOOP

IF i>10 THEN

EXIT;

END IF;

a1 := 2\*i;

dbms\_output.put\_line('2 x ' || i || ' = ' || a1 );

i := i+1;

-- i := i+1;

END LOOP;

END;

------------------------------------------

DECLARE

i NUMBER (3);

a1 number;

BEGIN

i := 1;

FOR i IN 1..10 LOOP

dbms\_output.put\_line( i );

END LOOP;

END;

DECLARE

a1 number;

b1 number;

c1 number;

FUNCTION findMax(x IN number,y IN number)

RETURN number

IS

z number;

BEGIN

IF x>y THEN

z :=x;

ELSE

z:=y;

END IF;

RETURN z;

END;

------------------------------------------

BEGIN

a1 := 10;

b1 := 10.8;

c1 := findMax(a1,b1);

dbms\_output.put\_line('Max no. is: ' || c1 );

END;

------------------------------------------

DECLARE

a1 number;

b1 number;

c1 number;

FUNCTION OddEven(x IN number)

RETURN number

IS

z number;

BEGIN

IF MOD(x,2)!=0 THEN

z :=1;

ELSE

z:=0;

END IF;

RETURN z;

END;

------------------------------------------

BEGIN

a1 := 11;

c1 := OddEven(a1);

IF c1!= 0 THEN

dbms\_output.put\_line('ODD ' || c1 );

ELSE

dbms\_output.put\_line('EVEN ' || c1 );

END IF;

END;

------------------------------------------

CREATE TABLE USER1(ID number(10) primary key, name varchar2(100));

CREATE procedure "INSERTUSER1"

(ID IN number,

name IN varchar2)

IS

BEGIN

INSERT INTO USER1 VALUES(ID, Name);

END;

BEGIN

INSERTUSER1(101,'RAHUL');

dbms\_output.put\_line('Record inserted successfully.');

END;

WRONG/ ROUGH

CREATE OR REPLACE procedure "CREATETABLE"

IS

BEGIN

BEGIN

CREATE TABLE DEPT(SR number(100),

DEPT varchar2 (100),

CODE int(5));

END;

BEGIN

INSERT INTO DEPT VALUES(SR, DEPT, CODE);

END;

BEGIN

UPDATE DEPT;

END;

END;

CREATE TABLE STUDENT(SNAME VARCHAR(100), COURSE VARCHAR(100), ROLLNO INT);

CREATE procedure "INSERTSTUDENT"

(SNAME IN varchar2,

COURSE IN varchar2,

ROLLNO IN number)

IS

BEGIN

INSERT INTO STUDENT VALUES(SNAME, COURSE, ROLLNO);

END;

BEGIN

INSERTSTUDENT('RAHUL','SCI',101);

INSERTSTUDENT('PRIYA','COMM',102);

INSERTSTUDENT('SAKSHI','SCI',103);

INSERTSTUDENT('RAJ','SCI',104);

INSERTSTUDENT('ROHAN','COMM',105);

dbms\_output.put\_line('Records inserted successfully.');

END;

------------------------------------------

DECLARE

CURSOR showRec(sno student.rollno%type) IS --DECLARING CURSOR

SELECT sname, course FROM STUDENT WHERE rollno=sno;

a1 student.sname%type;

b1 student.course%type;

c1 student.rollno%type;

d1 number;

BEGIN

d1 := :rollno;

OPEN showRec(d1);

IF showRec%Isopen = FALSE then

dbms\_output.put\_line('Cannot open cursor.');

ELSE

LOOP

FETCH showRec into a1,b1;

EXIT WHEN showRec%NOTFOUND;

dbms\_output.put\_line( a1 || ' ' || b1);

END LOOP;

END IF;

CLOSE showRec;

END;

------------------------------------------

INSERT INTO CUSTOMER (ID, NAME,AGE,ADDRESS, SALARY)

VALUES (3, 'CUST3', 28,'KOLKATA',60000) ;

(2,CUST2, 39,MUMBAI,9800),

(3, ‘CUST3’, 28,’ KOLKATA’,60000),

(4, CUST4, 40, PUNE,65000),

(5, CUST5, 34, MUMBAI,80000)

CREATE OR REPLACE TRIGGER DISPLAY\_SALARY\_CHANGES2

BEFORE DELETE OR INSERT OR UPDATE ON CUSTOMER

FOR EACH ROW

WHEN (NEW.ID>0)

DECLARE

SAL\_DIFF NUMBER;

BEGIN

SAL\_DIFF := :NEW.SALARY - :OLD.SALARY;

dbms\_output.put\_line( 'OLD SALARY: ' || :OLD.SALARY);

dbms\_output.put\_line( 'NEW SALARY: ' || :NEW.SALARY);

dbms\_output.put\_line( ' SALARY DIFF: ' || SAL\_DIFF);

END;

------------------------------------------

INSERT INTO CUSTOMER (ID, NAME,AGE,ADDRESS, SALARY)

VALUES (77, 'CUST3', 28,'KOLKATA',60000) ;

CREATE OR REPLACE PACKAGE C\_PACKAGE AS

--ADD CUSTOMER

PROCEDURE addCustomer(C\_ID CUSTOMER.ID%TYPE,

C\_NAME CUSTOMER.NAME%TYPE,

C\_AGE CUSTOMER.AGE%TYPE,

C\_ADDR CUSTOMER.ADDRESS%TYPE,

C\_SAL CUSTOMER.SALARY%TYPE);

--REMOVE CUSTOMER

PROCEDURE delCustomer(C\_ID CUSTOMER.ID%TYPE);

--LIST ALL CUSTOMER

PROCEDURE listCustomer;

END C\_PACKAGE;

------------------------------------------

CREATE OR REPLACE PACKAGE BODY C\_PACKAGE AS

PROCEDURE addCustomer(C\_ID CUSTOMER.ID%TYPE,

C\_NAME CUSTOMER.NAME%TYPE,

C\_AGE CUSTOMER.AGE%TYPE,

C\_ADDR CUSTOMER.ADDRESS%TYPE,

C\_SAL CUSTOMER.SALARY%TYPE)

IS

BEGIN

INSERT INTO CUSTOMER(ID,NAME,AGE,ADDRESS,SALARY)

VALUES(C\_ID,C\_NAME,C\_AGE,C\_ADDR,C\_SAL);

END addCustomer;

PROCEDURE delCustomer(C\_ID CUSTOMER.ID%TYPE)

IS

BEGIN

DELETE FROM CUSTOMER

WHERE ID=C\_ID;

END delCustomer;

END C\_PACKAGE;

PROCEDURE listCustomer IS

CURSOR C\_CUSTOMER IS

SELECT NAME FROM CUSTOMER;

TYPE C\_LIST IS TABLE OF CUSTOMER.NAME%TYPE;

NAME\_LIST C\_LIST := C\_LIST();

# Day 3 -Agile

**Scrum master**

A Scrum Master is a "servant-leader" and coach for an Agile team using the Scrum framework, focusing on facilitating communication, removing obstacles, coaching the team on Scrum principles, and protecting them from distractions to ensure project success. They are not a manager but a facilitator who helps the team become more self-sufficient and productive by fostering an effective environment and promoting the values of transparency, inspection, and adaptation.

Key Characteristics

* **Servant-Leader:** Focuses on serving the needs of the team and helping them succeed, rather than managing them.
* **People-Oriented:** Possesses high emotional intelligence and enjoys helping people grow and learn.
* **Flexible:** Stays open to new opportunities for the team to improve their workflows.
* **Knowledgeable:** Has a deep understanding of the Scrum framework and Agile principles.

**Product Owner**

In an Agile framework, a Product Owner (PO) is an individual responsible for maximizing the value of the product by defining its vision, managing the product backlog, and representing the needs of customers and stakeholders to the development team. Key responsibilities include creating and ordering product backlog items (like [user stories](https://www.google.com/search?sca_esv=61ade5bcf5ab3323&rlz=1C1GCEA_enIN1177IN1177&q=user+stories&sa=X&ved=2ahUKEwibubm3qbmPAxUPe2wGHbUgKjUQxccNegQILhAB&mstk=AUtExfAQxkXPM9V39DB6aOXJZ11-gGytoeyd1JmIhmFqnE21IrsStPiOSMu8Yfu6_Zm63EWEBEZj9BN0qnvvmz-bLLQ3FXAPyEaRo2343Dl-R0zw1yzTiLZU3aub6khMjDr1ROU&csui=3)), ensuring the backlog is transparent, and making decisions about product direction to meet business goals and user needs. The Product Owner acts as a crucial link between stakeholders and the development team, providing clear direction to ensure the team builds the right product.

**12 steps of agile**

**1**: Our highest priority is to satisfy the customer through the early and continuous delivery of valuable software.

**2**: Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.

**3**: Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

**4**: Business people and developers must work together daily throughout the project.

**5**: Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

**6**: The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

**7**: Working software is the primary measure of progress.

**8:** Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

**9:** Continuous attention to technical excellence and good design enhances agility.

**10:** Simplicity–the art of maximizing the amount of work not done–is essential.

**11:** The best architectures, requirements, and designs emerge from self-organizing teams.

**12:** At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

**Advantages of Agile methodology**

Agile Software Development Methodology is a process of software development similar to other software development methodologies like waterfall models, V-models, iterative models, etc. Agile methodology follows the iterative as well as incremental approach that emphasizes the importance of delivering of working product very quickly.

**Advantages of Agile Methodology**

Following are the advantages of agile methodology:

1. **Focus on Customer Value:** Agile places a high priority on providing customers with value by attending to their requirements and preferences. Agile guarantees that the most important features are produced first and that iterative changes are driven by customer feedback by dividing work down into small, manageable tasks.
2. **Enhanced Team Morale and Motivation:** Agile gives teams the freedom to own their work and decide together. Team members feel motivated, proud, and owned when they have this autonomy together with a focus on providing value and ongoing growth.
3. **Stakeholder Collaboration:**Throughout the development process, agile promotes strong coordination between product owners, developers, and other stakeholders. Better communication, a common understanding of the objectives, and ongoing feedback are all fostered by this partnership, which produces results that are higher quality and boost stakeholder satisfaction.
4. **Early and Continuous Delivery:**Agile encourages the tiny, incremental releases of functional software. This gives early access to observable progress and facilitates early input and validation for stakeholders. Continuous delivery reduces risks by spotting problems early on and taking action to fix them.
5. **Delivering high-quality software:** It is a key component of agile development, and this is emphasized by techniques like continuous integration, automated testing, and frequent inspection and modification. Agile guarantees that the software satisfies the required standards and lowers the likelihood of faults by integrating quality assurance throughout the development process.

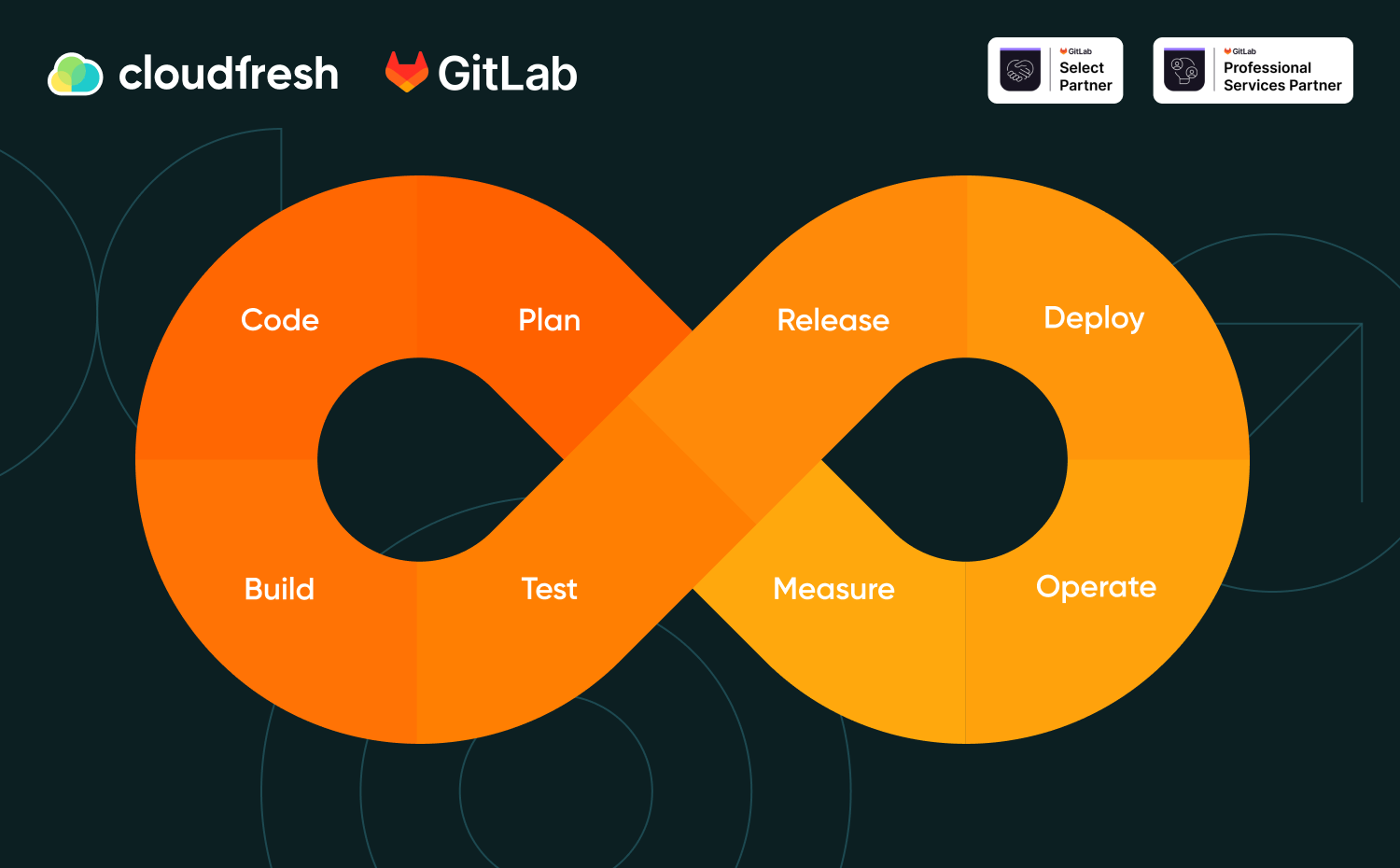
**Disadvantages of the Agile Methodology**

Following are the disadvantages of the agile methodology:

1. **Lack of Predictability:** Project timeframes and outcomes might be difficult to predict with accuracy due to Agile iterative and incremental methodology.
2. **Dependency on Customer Availability:** Agile highly depends on ongoing customer and stakeholder feedback and participation. Customers who are unavailable or who don't know enough about the domain can impede development and slow it down.
3. **Scaling Agile**: While Agile works effectively for small to medium-sized teams working on relatively basic projects, scaling Agile methods to bigger teams or more complicated projects can be more difficult. As the project grows, it gets harder to maintain coordination, alignment, and communication.
4. **Dependency on Team Dynamics:** Agile's focus on self-organizing, cross-functional teams with the authority to reach decisions together is paramount. Inadequate communication within the team or a lack of experience or expertise among team members can negatively affect output quality and productivity.
5. **Increased Overhead:** Planning, coordinating, and communicating take more time and effort when using agile frameworks like Scrum. This overhead can take a lot of time, especially for projects with short deadlines or small teams.

**CI/CD**

CI/CD (Continuous Integration/Continuous Delivery or Deployment) is an automated software development process that accelerates and streamlines the software lifecycle by continuously integrating, testing, and deploying code changes. Continuous Integration (CI) involves developers frequently merging their code changes into a central repository, where automated tests verify the changes and ensure the code remains releasable. Continuous Delivery/Deployment (CD) then automatically delivers or deploys these validated changes to a production environment, making updates available to users quickly and efficiently.



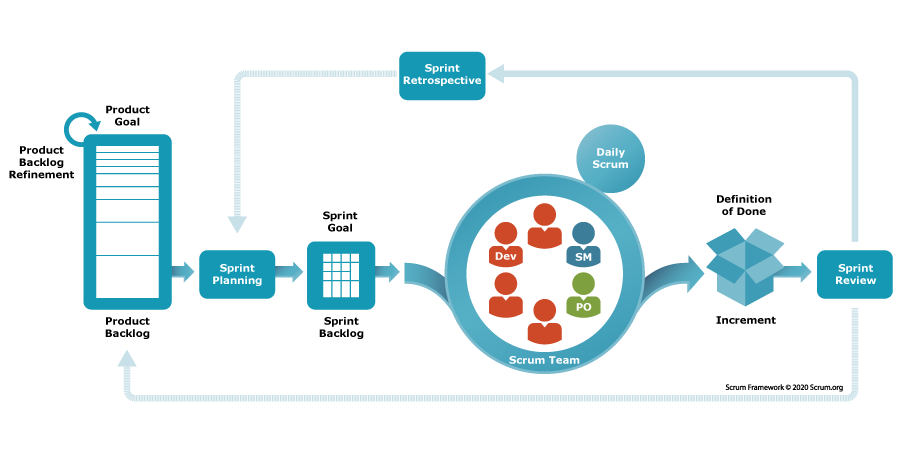
CI/CD tools automate and connect the entire software delivery process. They build code, run tests, and deploy to environments in a consistent, repeatable way. This automation provides fast feedback, reduces errors, and allows rapid delivery of high-quality software.

**Key benefits of Continuous Integration and Development:**

* Build automation – Eliminates manual work and speeds up release cycles.
* Test automation – Catches defects early.
* Automated deployments – Allows push-button releases.
* End-to-end pipelines – Connects build, test, deploy stages.
* Analytics/reports – Provides insight into delivery process.

By adopting CI/CD practices, teams can ship faster, more frequently, and with fewer headaches. The automation and rapid feedback cycles enabled by CI/CD are critical for agile workflows.

**What is scrum?**

If you are just getting started, think of Scrum as a way to get work done as a team in small pieces at a time, with continuous experimentation and feedback loops along the way to learn and improve as you go. Scrum helps people and teams **deliver value** incrementally in a collaborative way. As an **agile framework**, Scrum provides just enough structure for people and teams to integrate into how they work, while adding the right practices to optimize for their specific needs. 

**Uses of scrum**  
Scrum is used in project management as an Agile framework for teams to self-organize, manage complex projects, and adapt to changing requirements by working in short, iterative cycles called Sprints. Its primary uses are to deliver value quickly, promote team collaboration and accountability, and provide a transparent, empirical process for continuous improvement and innovation, especially in environments where the final product isn't entirely clear at the outset.

**Key Uses of Scrum**

* **Managing Complex Projects:**

Scrum is ideal for projects that are innovative, creative, or require extensive customer and stakeholder feedback, as its iterative nature allows for flexibility and adaptation.

* **Rapid Delivery of Value:**

By dividing work into short Sprints (typically 1-4 weeks), Scrum teams can deliver working increments of a product quickly and frequently.

* **Adapting to Change:**

Unlike traditional linear methods, Scrum allows teams to incorporate feedback and changes throughout the project lifecycle, ensuring the final product meets evolving needs.

* **Enhancing Teamwork and Self-Organization:**

The framework encourages teams to self-manage their work and collaborate effectively to achieve a common goal.

* **Promoting Transparency and Continuous Improvement:**

Scrum's processes, including regular meetings (events) and observable artifacts, increase transparency and provide opportunities for teams to inspect and adapt based on empirical observations.

* **Fostering Customer Satisfaction:**

Delivering usable portions of the product early allows for user feedback, ensuring that the project delivers what the customer truly needs and avoids unnecessary work.

**Core Applications of Scrum in Project Management**

**1. Iterative Development**

* Projects are broken into short cycles called **Sprints** (usually 1–4 weeks).
* Each Sprint delivers a potentially shippable product increment.
* This allows for frequent reassessment and adaptation.

**2. Enhanced Collaboration**

* Scrum encourages **daily stand-up meetings** (Daily Scrum) to keep everyone aligned.
* Roles like **Product Owner**, **Scrum Master**, and **Development Team** ensure clear responsibilities and communication.

**3. Transparency & Visibility**

* Artifacts like the **Product Backlog** and **Sprint Backlog** make work visible to all stakeholders.
* Regular reviews and retrospectives promote accountability and continuous improvement.

**4. Flexibility to Change**

* Scrum thrives in environments where requirements evolve.
* Teams can pivot quickly based on feedback, market changes, or new insights.

**5. Customer-Centric Delivery**

* The Product Owner represents the customer’s voice.
* Frequent demos and feedback loops ensure the product aligns with user needs.

**DevOps Model**

* 1. **Docker**
* Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called [containers](https://aws.amazon.com/containers/) that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.
* Running Docker on AWS provides developers and admins a highly reliable, low-cost way to build, ship, and run distributed applications at any scale.
* **Recent announcements**: Docker collaborates with AWS to help developers speed delivery of modern apps to the cloud. This collaboration helps developers use Docker Compose and Docker Desktop to leverage the same local workflow they use today to seamlessly deploy apps on Amazon ECS and AWS Fargate.

Docker works by providing a standard way to run your code. Docker is an operating system for containers. Similar to how a [virtual machine](https://aws.amazon.com/ec2/) virtualizes (removes the need to directly manage) server hardware, containers virtualize the operating system of a server. Docker is installed on each server and provides simple commands you can use to build, start, or stop containers.

2) **Jenkins**

Jenkins is a free, open-source tool that helps developers automate the process of building, testing, and deploying code. It is written in Java and runs on the Java platform. By using Jenkins, we can make a continuous integration of projects(jobs) or end-to-end automation.

Jenkins features:

* **Easy setup:**Jenkins runs anywhere Java does (Windows, Linux, macOS), with a simple web UI to configure it.
* **Availability:**Jenkins is free and open-source CI/CD server that can be easily downloaded, use and customize by anyone.
* **Massive plugin ecosystem:** Jenkins offers more than 2000+ plugins with compatibility with almost any tool.
* **Built for CI/CD: A**utomate builds, tests, and deployments instantly after each change.
* **Secure by design:**Jenkin authenticate from MFA UI logins to role-based access, it helps to secure both server and user access.

**3) Git**

Git is a popular version control system.

It was created by Linus Torvalds in 2005, and has been maintained by Junio Hamano since then.

It is used for:

* Tracking code changes
* Tracking who made changes
* Coding collaboration

**4) Selenium (IMP)**

**Kanban**

Kanban is a visual work management method, originating from Toyota's lean manufacturing, that uses a visual board to track tasks through a process, like "To Do," "Doing," and "Done". It emphasizes visualizing work, limiting work in progress (WIP) to prevent bottlenecks, and fostering continuous improvement by managing the flow of tasks and proactively addressing obstacles. The term "kanban" means "visual signal" or "card" in Japanese, referring to the cards that represent tasks on the board.

Kanban is a popular framework used to implement [Agile](https://www.atlassian.com/agile) and [DevOps](https://www.atlassian.com/devops/what-is-devops) software development. It requires real-time communication of capacity and full transparency of work. Work items are represented visually on a [kanban board](https://www.atlassian.com/agile/kanban/boards), allowing team members to see the state of every piece of work at any time.

**Visualize workflow**: Begin by visualizing your team's workflow on a Kanban board. Whether physical or virtual, the board should depict each stage of the development process, from task inception to completion.

**Standardize workflow**: Define and standardize the workflow stages according to your team's processes and requirements. Common stages include "To Do," "In Progress," and "Done," but customize as needed to reflect your unique workflow.

**Identify blockers and dependencies**: Ensure that your kanban board enables immediate identification of blockers and dependencies. This transparency allows for prompt resolution and prevents workflow disruptions.

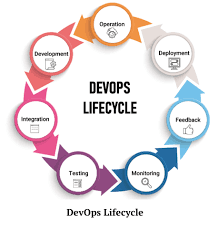
**Set work-in-progress (WIP) limits**: Implement WIP limits for each workflow stage to avoid overburdening and to maintain a steady workflow. WIP limits help optimize resource allocation and reduce multitasking, fostering higher productivity.

**Encourage collaboration**: Foster a culture of collaboration within your team, where members collectively address bottlenecks and work together to ensure smooth workflow progression. This collaborative approach promotes efficiency and accelerates task completion.

**Utilize kanban cards**: Represent each task as a kanban card on the board, containing essential details such as task description, assignee, and estimated time for completion. Kanban cards facilitate visual tracking of task progress and promote transparency within the team.

* **Visual Workflow Management**: Tasks are clearly displayed on a Kanban board, making progress and bottlenecks easy to spot.
* **Flexibility & Adaptability**: Teams can adjust priorities and workflows without rigid planning cycles.
* **Faster Delivery**: Limiting work-in-progress (WIP) helps reduce cycle times and speeds up task completion.
* **Improved Transparency**: Everyone can see what’s being worked on, what’s pending, and what’s done.
* **Enhanced Collaboration**: Shared boards foster team communication and alignment.
* **Reduced Waste**: Focused workflows minimize unnecessary tasks and distractions.
* **Continuous Improvement**: Regular reviews and data-driven tweaks help refine processes over time.
* **Better Prioritization**: Teams can focus on high-value tasks based on real-time needs.
* **Empowered Decision-Making**: Visibility and data support smarter, faster decisions.
* **Less Overload**: WIP limits prevent burnout and keep workloads manageable.

**DevOps Lifecycle**



**Phases of DevOps Lifecycle**

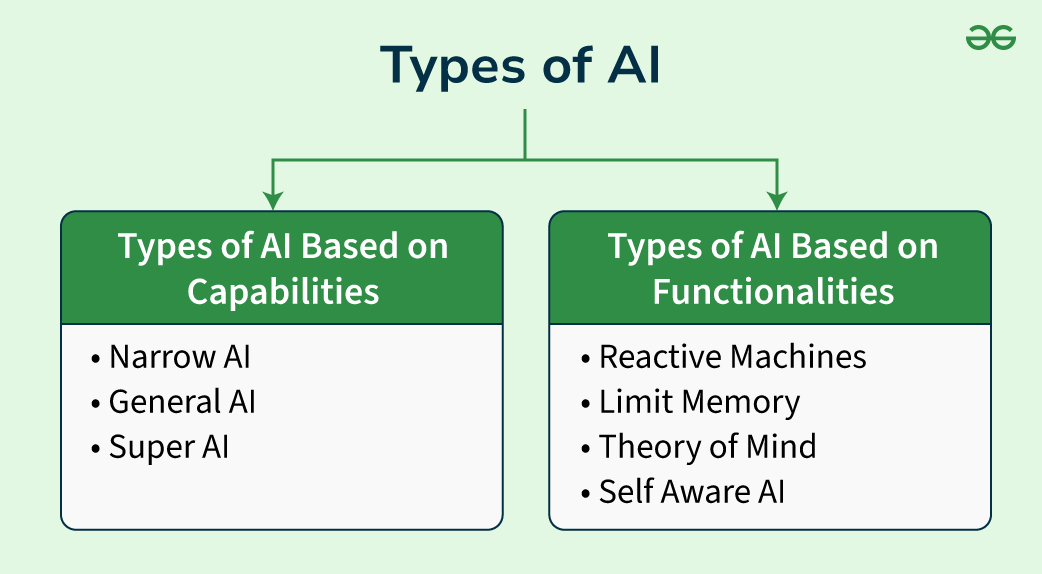
1. **Plan**: This phase focuses on understanding the business needs and gathering feedback from end-users. Teams create a plan that aligns the project with business goals and ensures the right results are delivered.
2. **Code**: In this phase, developers write the actual code for the software. Tools like Git help manage the code, making sure that the code is well-organized and free from security issues or bad coding practices.
3. **Build**: Once the code is written, it is submitted to a central system using tools like Jenkins. This step ensures the code is compiled, and all components are integrated together smoothly.
4. **Test**: The software is then tested to ensure it works properly. This includes different types of tests like security, performance, and user acceptance. Tools like JUnit and Selenium are used to automate these tests and verify the software’s integrity.
5. **Release**: After testing, the software is ready to be released to production. The DevOps team ensures that all checks are passed and then sends the latest version to the production environment.
6. **Deploy**: Using Infrastructure-as-Code (IaC) tools like Terraform, the necessary infrastructure (servers, networks, etc.) is automatically created. Once the infrastructure is set up, the code is deployed to various environments in an automated and repeatable way.
7. **Operate**: Once deployed, the software is available for users. Tools like Chef help manage the configuration and ongoing deployment of the system to ensure it operates smoothly.
8. **Monitor**: This phase involves observing how the software is performing in the real world. Data about user behaviour and application performance is collected to identify any issues or bottlenecks. By monitoring the system, the team can quickly spot and fix problems that may affect performance.

# Day 4 – Gen AI

**What is AI?**

Artificial intelligence (AI) is technology that enables computers and machines to simulate human learning, comprehension, problem solving, decision making, creativity and autonomy.

Artificial Intelligence refers to something which is made by humans or non-natural things and Intelligence means the ability to understand or think. AI is not a system but it is implemented in the system. There are many different**types of AI**, each with its own strengths and weaknesses.



**How Does AI Affect Our Daily Lives?**

Artificial intelligence (AI), once the stuff of science fiction, has quietly woven itself into the fabric of our everyday lives. It’s present in the technology we use daily, from the phones in our pockets to the smart systems in our homes and workplaces. For example, AI can be used to help shape our experience by personalizing news feeds on our screens or self-adjusting thermostats in our living spaces. These AI algorithms are not just learning our habits; they can potentially anticipate our needs and reshape our experiences in ways both subtle and profound. As AI blurs the lines between human and machine, it opens up possibilities and presents new challenges, marking an era of ongoing technological evolution. One thing is clear: AI is here to stay. In this article, we analyze five key ways in which Artificial intelligence’s ever-expanding technologies have impacted our daily lives.

**1. Artificial Intelligence in Personal Finance**

AI is transforming the personal finance sector, offering new ways to manage money. This change is evident in several areas:

* **Robo-Advisors:** AI-powered robo-advisors, digital platforms that automatically manage your investments based on pre-defined financial goals and risk tolerance, are redefining accessibility in financial advice. By analyzing personal financial objectives and risk preferences, these platforms offer tailored portfolio recommendations. These tools aim to provide guidance and portfolio management, often with a focus on cost-efficiency compared to traditional advisory services.[**1**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#e9c30bb2-d375-4ad2-a381-30de29335726)
* **AI Investment Strategies:**AI-powered trading tools use algorithms to analyze market trends, offering insights for making informed trading decisions. These tools operate based on an investor’s [AI investment](https://www.linqto.com/blog/how-to-invest-in-artificial-intelligence/) goals and risk tolerance, aiding in navigating the investment landscape.
* **Security and Fraud Detection:** AI’s role in security and fraud detection involves deep learning algorithms that analyze financial data to identify patterns indicative of fraud. This proactive approach allows AI systems to adapt to new threats, enhancing the protection of your finances.[**2**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#cdebc4a6-c923-429e-8d0e-e2563af3ef1c)

**2. Artificial Intelligence in Healthcare**

AI is impacting the healthcare sector, enhancing diagnostics, treatment accuracy, and patient care, along with streamlining administrative processes. Key areas of impact include:

* **Enhanced Diagnostics and Precision Medicine:** AI algorithms are being utilized to analyze medical images like X-rays and MRIs, with high accuracy. Their ability to detect subtle abnormalities may aid in early diagnosis and potentially improve treatment planning, especially for conditions such as cancer and neurological diseases.[**3**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#cc2321d8-69f1-4d0f-ad14-cc6da1897b8c)
* **Robo-Assisted Surgery:** AI-driven surgical robots are performing complex procedures with a level of precision and dexterity challenging for human hands, leading to benefits such as minimal incisions, quicker recovery, and lower complication risk. Johns Hopkins found that one of these robots was able to outperform human surgeons for certain procedures.[**4**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#4ecd82f6-8fc7-4317-b0f5-9ff9bae21812)
* **Drug Discovery and Development:** In the future, AI may help play a role in analyzing datasets of genetic and molecular information to help identify potential drug targets and the development of effective medications. Additionally, AI’s ability to consider a patient’s genetic makeup and medical history could enhance personalized treatment plans, potentially improving clinical outcomes.
* **Disease and Virus Studies:** AI is being used in pathology and microbiology. This development holds the potential to enhance our understanding of the cellular foundations of diseases and the clustering of conditions within patient populations. This advancement could enable more precise preventive strategies, such as leveraging immunomics to improve diagnosis and predict care and treatment options more effectively. Such developments are poised to revolutionize multiple standards of care, particularly in cancer, neurological disorders, and rare diseases, paving the way for a more personalized healthcare experience tailored to the individual.[**5**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#93db8421-6a7f-414d-8d03-ef4f2a7e32d2)

**3. How AI is Redefining Education**

Imagine an educational environment where each student’s learning journey is uniquely catered to their pace and style. In classrooms powered by artificial intelligence, this is becoming a reality. Virtual tutors provide tailored explanations of complex concepts, and real-time assessments offer personalized feedback. This isn’t a futuristic utopia – this is the emerging landscape of AI education. AI’s influence in the sector is multifaceted, including:

* **Engaged Learning Platforms:** These AI-powered platforms function like interactive tutors disguised as digital textbooks. They assess a student’s strengths, weaknesses, and learning preferences, customizing the curriculum and difficulty level to their individual needs. Through engaging tools such as interactive games or visual aids, these platforms not only maintain student engagement but also encourage a deeper understanding of varied subject materials.
* **AI in Classroom Management**: AI is streamlining administrative tasks like attendance tracking and grading. This allows teachers more time for direct student interaction and tailored guidance. AI tools can also identify students who may need help in specific areas, enabling teachers to intervene effectively and early. This approach may lead to a more efficient and supportive learning environment.
* **Accessibility and Inclusivity:**AI is also playing a role in helping to create a more inclusive learning environment for students with disabilities. For instance, text-to-speech and speech-to-text technologies support students with reading or writing difficulties, while AI-driven translation tools help non-native speakers overcome language barriers.[**6**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#b5a3b2cf-6777-4880-852b-95b9efa09323)

**4. AI in Transportation**

The transportation sector–an industry once powered by pistons and petrol–is undergoing a seismic shift with the integration of AI. From the development of self-driving cars gliding silently through city streets to advanced traffic management systems, AI is helping to redefine the way we travel. Key technological advancements include:

* **Self-Driving Vehicles:** Often seen as the hallmark of AI-powered transportation, self-driving cars are being developed with the aim to enhance safety, efficiency, and accessibility. These vehicles use AI algorithms to interpret data from sensors and cameras, allowing them to navigate roads with quick reaction times and heightened precision.[**7**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#a96a5ea1-b5c3-46c6-8ba5-43c6261e876b)  
  **AI in Traffic Management:** Gridlock and urban congestion could soon become relics of the past, as AI-powered traffic management systems begin to take the helm. These systems utilize real-time traffic data, dynamically adjusting traffic lights, suggesting alternate routes, and even optimizing public transportation schedules. By doing so, AI aims to enhance the efficiency of city streets, rural roads, and suburban routes, helping everyone reach their destinations more quickly and safely.[**8**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#e333403e-ee3d-460b-b9ad-1bfe84a1ba9f)  
  **Enhanced Public Transportation:** AI contributes to making buses, trains, and subways more efficient. Predictive maintenance helps in reducing breakdowns, while schedule optimization minimizes delays. AI-enhanced passenger information systems offer real-time updates and personalized travel suggestions, while accessibility features like automated doors and voice announcements are making public transportation more accessible, particularly for passengers with disabilities.[**9**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#ddca9461-15a1-4ee7-a869-848e364dc6b3)

**5. Evolving E-Commerce**

The e-commerce landscape has been influenced by AI, leading to a more personalized and intuitive shopping experience. AI plays a key role from the moment a consumer begins their online shopping journey to the delivery of their purchases at their doorstep. Key AI capabilities include:

* **Smart Product Recommendations:** AI algorithms can be adept at suggesting products tailored to consumer preferences. These algorithms can often analyze not just past purchases, a consumer’s browsing history, and other interactions. They can use these insights to match customers with products that align with their interests.[**10**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#29361c83-2d44-46a1-ae84-06dc5ee94eed)
* **Virtual Fitting Rooms:** By analyzing body measurements or scanned photos, AI algorithms can suggest accurately sized clothing and accessories. This technology extends to augmented reality (AR) experiences, allowing customers to virtually try on items from the comfort of their homes, adding a new dimension to online shopping.[**11**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#b2b705f8-f07d-4add-ac13-84b84f4011ba) [**12**](https://www.linqto.com/blog/ways-artificial-intelligence-ai-is-affecting-our-daily-lives/#ae40ce44-c26e-4274-831d-66e25f227ba6)
* **Fraud Detection and Secure Transactions:** AI algorithms can monitor and analyze transactions in real-time, detecting unusual patterns and activities that may indicate fraud. These algorithms can help identify irregular spending behaviors, unfamiliar login attempts, and even unexpected location changes, helping to safeguard consumers’ online-shopping activities.

**Goals of AI**

AI’s primary goals revolve around solving real-world problems, improving human decision-making capabilities, automating tasks, and driving innovation. These goals make AI integral to enhancing efficiency across industries, fostering creativity, and enabling groundbreaking advancements. Below, we delve into the major goals of AI, their applications, and how they bring value to modern society.

**1. Problem-Solving and Decision-Making**

One of the fundamental goals of Artificial Intelligence is to enable systems to solve complex problems and automate decision-making processes. AI algorithms use data-driven insights to analyze vast amounts of information, identify patterns, and make accurate, informed decisions in real time. By automating these processes, AI enhances operational efficiency and ensures faster outcomes.

**Applications:**

* **Chess Engines:** AI-powered engines like *Deep Blue* demonstrate decision-making capabilities by evaluating countless possible moves to find the optimal solution.
* **Predictive Analytics:** In finance, AI forecasts stock market trends, optimizes investments, and identifies potential risks.
* **Logistics Optimization:** AI tools streamline supply chains by minimizing delivery times, optimizing routes, and predicting maintenance needs.

**Benefits:**AI’s ability to solve problems efficiently allows businesses to reduce human errors, make data-backed decisions, and boost productivity. By analyzing vast datasets, AI uncovers insights that human decision-makers might overlook, ensuring real-time solutions in critical situations.

**2. Natural Language Processing (NLP)**

Natural Language Processing is another critical goal of AI. It focuses on enabling machines to understand, interpret, and generate human language. NLP bridges the communication gap between humans and machines, making interactions more seamless and technology more accessible.

**Applications:**

* **Chatbots:** AI-powered chatbots, such as those used in customer service, provide instant and accurate responses to user queries, reducing response time and improving user satisfaction.
* **Translation Tools:** Applications like *Google Translate* overcome language barriers by accurately translating text and speech into multiple languages.
* **Voice Assistants:** AI assistants like *Siri*, *Alexa*, and *Google Assistant* use NLP to understand voice commands and perform tasks such as setting reminders, providing information, or controlling smart devices.

**Benefits:**NLP enhances communication between humans and machines, simplifying complex interactions. It plays a vital role in automating customer service, enabling cross-language communication, and offering personalized user experiences.

**3. Machine Learning and Deep Learning**

Machine Learning (ML) and Deep Learning (DL) are at the heart of AI systems, allowing machines to learn from data and improve their performance over time without human intervention. ML focuses on building models that identify patterns and make predictions, while DL uses neural networks to solve more complex problems.

**Applications:**

* **Image Recognition:** AI tools identify objects, people, and activities in images, which is widely used in healthcare for diagnosing diseases and in security systems for facial recognition.
* **Recommendation Systems:** Platforms like *Netflix*, *YouTube*, and *Amazon* analyze user behavior to recommend movies, videos, or products, improving customer engagement and satisfaction.
* **Fraud Detection:** In the finance industry, ML algorithms analyze transactions to detect anomalies and flag fraudulent activities.

**Benefits:**ML and DL automate insights, predictions, and decision-making processes across various domains. They power applications ranging from autonomous vehicles to predictive healthcare, transforming industries by providing innovative, data-driven solutions.

**4. Robotics and Automation**

Robotics and automation aim to combine AI with machines to perform tasks with precision, consistency, and autonomy. AI-powered robots are designed to handle labor-intensive, repetitive, or hazardous tasks, enabling improved operational efficiency and workplace safety.

**Applications:**

* **Manufacturing Automation:** AI-driven robots assemble products, conduct quality inspections, and manage production lines, increasing accuracy and reducing production costs.
* **Autonomous Vehicles:** Companies like *Tesla* and *Waymo* develop self-driving cars that navigate roads using AI, minimizing human intervention and reducing accidents.
* **Delivery Drones:** Logistics companies use drones to ensure faster and more efficient package deliveries, particularly in remote or inaccessible areas.

**Benefits:**Robotics and automation streamline workflows, enhance productivity, and reduce human involvement in dangerous tasks. Industries such as manufacturing, logistics, and healthcare rely on AI-powered robots to drive innovation and ensure operational excellence.

**5. Enhancing Healthcare and Medicine**

AI is revolutionizing the healthcare sector by enabling better diagnosis, treatment, and research capabilities. AI systems analyze vast medical datasets to uncover insights, improving patient outcomes while reducing healthcare costs.

**Applications:**

* **Medical Imaging:** AI-powered tools like *DeepMind* analyze radiology scans to detect diseases such as cancer at early stages, improving diagnosis accuracy.
* **Personalized Treatments:** AI tailors treatment plans based on patient data, ensuring customized and effective care.
* **Drug Discovery:** AI accelerates drug research and development by predicting the efficacy of compounds, reducing the time and cost of clinical trials.

**Benefits:**By leveraging AI, healthcare providers can diagnose diseases faster, reduce costs, and deliver personalized treatments. AI’s integration into healthcare ensures improved patient care and better resource management.

**6. Fostering Creativity and Innovation**

Contrary to the perception that AI focuses solely on analytical tasks, one of its emerging goals is fostering creativity and innovation. AI tools assist humans in generating creative outputs across industries, enabling professionals to push the boundaries of art, design, and technology.

**Applications:**

* **Generative AI:** Tools like *DALL·E* and *MidJourney* create unique artworks, images, and designs based on textual descriptions, enabling artists and designers to experiment with new ideas.
* **Music Composition:** AI-powered platforms generate original music compositions, helping artists streamline their creative processes.
* **Content Creation:** AI tools assist in writing articles, brainstorming ideas, and automating storytelling, boosting productivity for writers and marketers.

**Benefits:**AI enhances human creativity by automating repetitive tasks, offering fresh perspectives, and accelerating innovation. It empowers professionals to focus on high-value, imaginative work, transforming industries like art, media, and entertainment.

AI’s primary goals encompass solving complex challenges, improving decision-making, and driving innovation across diverse sectors. From automating workflows and enhancing healthcare to fostering creativity, AI continues to reshape industries and improve lives worldwide. Each goal plays a critical role in harnessing the true potential of Artificial Intelligence, making it a transformative force in modern

What is Gen AI?

Generative AI is a type of artificial intelligence that can create new content, such as text, images, music, audio, and code, based on the data it has been trained on. Unlike traditional AI, which typically analyzes existing data or performs predefined tasks, generative AI learns patterns and structures from vast datasets and then uses that knowledge to produce novel, human-like outputs in response to prompts

**Strengths of Generative AI**

* Diverse outputs

GenAI can produce what seems like diverse and original outputs. It can create content through capturing nuances in language, based on patterns which we may not have seen before in the data they were trained on. This helps to open up different perspectives and can give us ideas on how to explore topics from a variety of viewpoints.

* Levelling the playing field

Gen AI can process and interpret human language in a conversational style, which allows it to generate contextually relevant responses to user prompts. It can also reformulate text to simplify or summarise it, which can help people start to understand more complex ideas. Gen AI can also process and generate text in multiple languages.

If used appropriately, Gen AI can be a great leveller for those who do not speak English as their first language or may not have the same literacy or language skills as others.

* Organizational productivity

Gen AI can be fine-tuned for different domains, so it can be made widely available for a variety of tasks. Some examples include chatbots, content generation and language translation. This can help boost organizational productivity. Gen AI can answer questions in a human-like style, reduce effort on tedious and monotonous tasks, provide accessible summaries of complex topics, produce automatic translations and transcriptions, etc.

* Personalization

Gen AI models can remember previous interactions, which results in more coherent and relevant conversation experiences for users. You can ask some models to remember your writing style or how you want to present your data. You can even ask a chat tool to test your knowledge against any piece of content. Gen AI can generate quick responses, which produces rapid interactions and real-time applications.

* Industry applications

It is anticipated that most industry and workplaces will be using a form of Generative AI in the future to enhance and optimise their work. Gen AI is already integrating into our daily learning and work tools, such as Copilot within Microsoft Office, or the AI content generator in Grammarly. So it is important to develop your skills in using Gen AI effectively and ethically.

**Weaknesses of Generative AI**

* Lack of trust and authenticity

Gen AI can generate information that appears factual but is often inaccurate. This is often called AI hallucinations. We must remember that:

* although Gen AI models appear to understand the content that they use and generate, they do not understand it
* the data that Gen AI models use for training have lots of inaccuracies and biases in them already
* Gen AI can also easily create fake news, misinformation and ‘deep fakes’.

Most AI models are created to provide a likely output based on prompts and training. Their outputs are designed to appear convincing even when there is no factual basis for the output. Consequently, ‘facts’ provided by these tools may appear to be trustworthy, but that appearance is false. Both input data and prompts can lead to bias in the output. As a result, ALL outputs from all AI tools must be independently verified for truthfulness.

* Copyright and ownership

Gen AI output imitates or summarises existing content, mostly without the permission of the original content owners. The output's appearance of creativity and originality generates challenges for us. There are issues of copyright, ownership, intellectual property and lack of authoritative legislation in this rapidly evolving area. It is important to keep this in mind when using Gen AI tools.

You should not copy and paste any copyrighted text, or other sensitive or personal data, into an AI tool for it to use. The AI tool could incorporate this data into its training dataset, which could then be used illegally or unethically.

* Carbon footprint

Training Gen AI requires huge amounts of power, which indirectly generates huge amounts of carbon. This has important consequences for climate change.

For example, an estimate of the electricity needed to train ChatGPT-4 is between 51,772–62,318 megawatt hours of electricity. This generated between 1,035–14,994 metric tons of carbon dioxide emissions. The variations depend on the global location of the training. As a comparison, a 3000-mile round trip flight from London to Boston emits 1 metric ton of carbon dioxide.

* Feedback loop

The output of Gen AI is flooding the internet through tools such as ChatGPT. This poses an interesting risk for future GPT (Generative Pre-trained Transformer) models and leads to the concept of model collapse.

Future models trained with online content that earlier GPT models have created will include all its biases and errors. This self-referential loop compounds the mistakes in data. This might contaminate the training data and lead to model collapse. A model collapse occurs where the models forget most of the original data that they learnt from.

* Ethical, Social and Human costs

After training, the Gen AI model is often checked and refined in a process known as Reinforcement Learning from Human Feedback (RLHF). In RLHF, human beings review the Gen AI responses and validate them. This ensures that the Gen AI responses are appropriate, accurate and align with the intended purpose.

There are issues of exploitation around this work. For ChatGPT, the RLHF reviewers were mostly workers in global south countries such as Kenya. Workers were paid less than $3 per hour to review the outputs of ChatGPT and identify any objectionable or toxic materials. This work has had a massive negative impact on many of those who were involved, including experiencing trauma.

Gen AI also tends to output standard answers that replicate the values and biases of the creators of the data used to train the models. This may constrain the development of plural opinions and further marginalise already marginalised voices.

**Advantages of Generative AI**

**1. Boosts Productivity**

* Automates content creation (text, images, code, etc.)
* Assists in writing, editing, and brainstorming

**2. Enhances Creativity**

* Generates novel ideas for art, music, design, and storytelling
* Helps non-experts produce high-quality creative work

**3. Cost and Time Efficiency**

* Reduces time and cost in areas like marketing, software development, customer service, and design

**4. Personalization at Scale**

* Customizes content (e.g., ads, emails, product suggestions) for individual users efficiently

**5. Accessibility**

* Assists people with disabilities (e.g., generating image descriptions for the visually impaired)
* Offers real-time translation and language support

**6. Rapid Prototyping**

* Designers and engineers can test ideas faster using AI-generated mockups or simulations

**Disadvantages of Generative AI**

**1. Misinformation & Deepfakes**

* Can easily generate convincing fake news, videos, or images
* Risk of spreading false or harmful content

**2. Job Displacement**

* May replace human roles in writing, art, customer service, coding, etc.
* Raises concerns about long-term employment impacts

**3. Bias & Inaccuracy**

* Models may reproduce or amplify biases from training data
* Can generate incorrect or misleading information

**4. Ethical and Legal Issues**

* Intellectual property violations (e.g., generating content in the style of real artists)
* Lack of clear legal frameworks around ownership and accountability

**5. Over-reliance**

* People may depend too much on AI, weakening their own critical thinking or creative skills

**6. Privacy Risks**

* AI trained on public or scraped data may inadvertently reveal sensitive or personal information

**Tic tac toe Game**

**HTML**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Tic Tac Toe</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<h1>Tic Tac Toe</h1>

<div id="game">

<div class="cell" data-index="0"></div>

<div class="cell" data-index="1"></div>

<div class="cell" data-index="2"></div>

<div class="cell" data-index="3"></div>

<div class="cell" data-index="4"></div>

<div class="cell" data-index="5"></div>

<div class="cell" data-index="6"></div>

<div class="cell" data-index="7"></div>

<div class="cell" data-index="8"></div>

</div>

<p id="status">Player X's turn</p>

<button id="reset">Restart Game</button>

<script src="script.js"></script>

</body>

</html>

**CSS**

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

background-color: #f4f4f4;

}

h1 {

margin-bottom: 20px;

}

#game {

display: grid;

grid-template-columns: repeat(3, 100px);

gap: 5px;

justify-content: center;

margin-bottom: 20px;

}

.cell {

width: 100px;

height: 100px;

background-color: white;

font-size: 2.5rem;

display: flex;

align-items: center;

justify-content: center;

border: 2px solid #333;

cursor: pointer;

}

.cell:hover {

background-color: #eee;

}

#status {

font-size: 1.2rem;

margin-bottom: 10px;

}

#reset {

padding: 10px 20px;

font-size: 1rem;

cursor: pointer;

}

**JS**

const cells = document.querySelectorAll('.cell');

const statusText = document.getElementById('status');

const resetBtn = document.getElementById('reset');

let board = ["", "", "", "", "", "", "", "", ""];

let currentPlayer = "X";

let gameActive = true;

const winConditions = [

[0, 1, 2], [3, 4, 5], [6, 7, 8], // Rows

[0, 3, 6], [1, 4, 7], [2, 5, 8], // Columns

[0, 4, 8], [2, 4, 6] // Diagonals

];

function handleCellClick(e) {

const index = e.target.dataset.index;

if (board[index] !== "" || !gameActive) return;

board[index] = currentPlayer;

e.target.textContent = currentPlayer;

if (checkWinner()) {

statusText.textContent = `Player ${currentPlayer} wins!`;

gameActive = false;

} else if (!board.includes("")) {

statusText.textContent = "It's a draw!";

gameActive = false;

} else {

currentPlayer = currentPlayer === "X" ? "O" : "X";

statusText.textContent = `Player ${currentPlayer}'s turn`;

}

}

function checkWinner() {

return winConditions.some(combination => {

return combination.every(i => board[i] === currentPlayer);

});

}

function resetGame() {

board = ["", "", "", "", "", "", "", "", ""];

currentPlayer = "X";

gameActive = true;

statusText.textContent = "Player X's turn";

cells.forEach(cell => cell.textContent = "");

}

cells.forEach(cell => cell.addEventListener('click', handleCellClick));

resetBtn.addEventListener('click', resetGame);

 A qr code on a white background

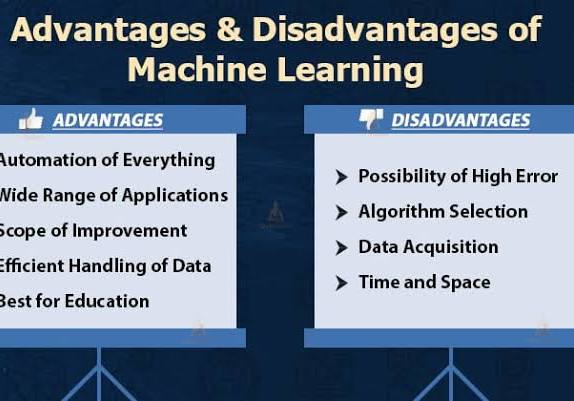
AI-generated content may be incorrect.



**What is ML?**

Machine learning is a branch of artificial intelligence that enables algorithms to uncover hidden patterns within datasets. It allows them to predict new, similar data without explicit programming for each task. Machine learning finds applications in diverse fields such as image and speech recognition, natural language processing, recommendation systems, fraud detection, portfolio optimization, and automating tasks.

**Adv and dis adv of ML**



**LLM Model**

An LLM, or Large Language Model, is an advanced artificial intelligence (AI) system designed to understand and generate human language. Trained on enormous amounts of text data, LLMs learn patterns and context to produce relevant and coherent text in response to prompts. They power applications like chatbots, content creation tools, and translation services by leveraging complex neural networks, such as the [transformer architecture](https://www.google.com/search?sca_esv=fc1f47d9dc5800d6&rlz=1C1GCEA_enIN1177IN1177&q=transformer+architecture&sa=X&ved=2ahUKEwix2Ja349-PAxW72TgGHdjGI2wQxccNegQIKxAB&mstk=AUtExfA6Aux5o5yBs51o2R_ozdz9SH1a8w5fl0HcZPZc66OP7HLzpySBkaPdrACvDbx6uVDAFNQc953R_XuuT6r9VTWc9BpCkuARZw42m-Rhym2a7AaTEtX-NBe3F2i7AIlzaYE&csui=3), to process entire sequences of text in parallel and predict likely next words.

**Reaction time game**

HTML

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<title>Reaction Time Tester</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<div class="container" id="screen">

<h1 id="message">Click to Start</h1>

</div>

<script src="script.js"></script>

</body>

</html>

**CSS**

body, html {

height: 100%;

margin: 0;

font-family: Arial, sans-serif;

}

.container {

display: flex;

justify-content: center;

align-items: center;

height: 100%;

background-color: #3498db;

color: white;

font-size: 2rem;

cursor: pointer;

user-select: none;

transition: background-color 0.3s ease;

text-align: center;

}

**JS**

const screen = document.getElementById("screen");

const message = document.getElementById("message");

let startTime;

let timeout;

let isWaiting = false;

let isGreen = false;

function startGame() {

message.textContent = "Wait for green...";

screen.style.backgroundColor = "#e74c3c"; // red

isWaiting = true;

isGreen = false;

// Random time between 2 and 5 seconds

const delay = Math.random() \* 3000 + 2000;

timeout = setTimeout(() => {

screen.style.backgroundColor = "#2ecc71"; // green

message.textContent = "CLICK!";

startTime = Date.now();

isGreen = true;

isWaiting = false;

}, delay);

}

screen.addEventListener("click", () => {

if (isGreen) {

const reactionTime = Date.now() - startTime;

message.textContent = `Your reaction time is ${reactionTime} ms. Click to try again.`;

screen.style.backgroundColor = "#3498db"; // blue (neutral)

isGreen = false;

} else if (isWaiting) {

// Clicked too early

clearTimeout(timeout);

message.textContent = "Too soon! Click to try again.";

screen.style.backgroundColor = "#f39c12"; // orange

isWaiting = false;

} else {

startGame();

} });

A screen shot of a cell phone

AI-generated content may be incorrect.

**Difference between LLM and NLP**

| **Aspect** | **NLP (Natural Language Processing)** | **LLMs (Large Language Models)** |
| --- | --- | --- |
| **Definition** | Field of AI for language understanding and processing | Subset of NLP focused on massive pre-trained models |
| **Key Techniques** | Rule-based systems, statistical models, ML, deep learning | Deep learning with transformers (GPT, BERT, T5) |
| **Components** | Tokenization, POS tagging, NER, MT, sentiment analysis | Transformers, attention mechanisms, fine-tuning |
| **Performance** | Task-specific and varies with data and method | High performance in open-domain tasks; supports zero-shot learning |
| **Complexity** | Ranges from simple to moderate | Highly complex; resource-intensive |
| **Resource Needs** | Moderate; runs on standard hardware | High; requires GPUs or TPUs for training and inference |
| **Examples** | spaCy, NLTK, OpenNLP, Stanford NLP | GPT-3/4 (OpenAI), BERT, T5, Claude |



# Day 5 -Java and OOPs

**JRE**

JRE stands for the Java Runtime Environment. It is a software package that provides the necessary components to run Java programs, including the [**Java Virtual Machine (JVM)**](https://www.google.com/search?sca_esv=fc1f47d9dc5800d6&rlz=1C1GCEA_enIN1177IN1177&q=Java+Virtual+Machine+%28JVM%29&sa=X&ved=2ahUKEwjypav349-PAxXlxjgGHaANAz8QxccNegQIJxAB&mstk=AUtExfCux52mrSvY1VClbdNq69vQ3Rmvi0SS2h7YaQiEPtr59Mm4lW0cnf30b6cPixUB_JfwAE6TLNwL8zqIbLp3doubPn-ixdeycsEAK0_BE1r0BhNXBmT1egNpO1dkfv4W8H4&csui=3) and core Java class libraries and supporting files. Users need a JRE installed on their computer to execute Java applications, even if they are not programmers themselves.

**JVM**

A Java Virtual Machine (JVM) is a virtual engine that allows Java programs to run on any device, providing a platform-independent environment. It translates compiled Java [**bytecode**](https://www.google.com/search?sca_esv=fc1f47d9dc5800d6&rlz=1C1GCEA_enIN1177IN1177&q=bytecode&sa=X&ved=2ahUKEwjEueCH5N-PAxUB8DgGHSJpAMIQxccNegQIKxAB&mstk=AUtExfD9oXd9huHqyUhJHXRMD8FamODkJS3ABpBxhm_ZrUPa1yLKoevTTMsfNBiUft6lsibEzZ5JkoE3j14ZZjWsQpySdcW_YU6l75wo6ZGpNWfNXie-BDiS3fG0lcqB7UIWk5Y&csui=3) into the machine code of the specific operating system or hardware it's running on. This makes Java code portable, enabling the "Write Once, Run Anywhere" (WORA) principle, where a single Java program can execute on different platforms without modification.

**Java code to do arithmetic operations**

import java.util.Scanner;

public class Calculator {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter first number: ");

double num1 = input.nextDouble();

System.out.print("Enter second number: ");

double num2 = input.nextDouble();

double sum = num1 + num2;

double difference = num1 - num2;

double product = num1 \* num2;

double quotient = num2 != 0 ? num1 / num2 : Double.NaN;

System.out.println("Addition: " + sum);

System.out.println("Subtraction: " + difference);

System.out.println("Multiplication: " + product);

if (num2 != 0) {

System.out.println("Division: " + quotient);

} else {

System.out.println("Division: Cannot divide by zero");

}

input.close();

} }

--------------------------------------------

import java.util.Scanner;

public class StudentResult {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter student's name: ");

String name = input.nextLine();

// Array to store marks

double[] marks = new double[5];

double total = 0;

// Taking marks for 5 subjects

for (int i = 0; i < 5; i++) {

System.out.print("Enter marks for Subject " + (i + 1) + ": ");

marks[i] = input.nextDouble();

total += marks[i];

}

// Calculating percentage

double percentage = total / 5;

// Displaying result

System.out.println("\n--- Result ---");

System.out.println("Student Name: " + name);

System.out.println("Total Marks: " + total + " out of 500");

System.out.println("Percentage: " + percentage + "%");

input.close();

}

}

O/P:

Enter student's name: Abc xyz

Enter marks for Subject 1: 49

Enter marks for Subject 2: 80

Enter marks for Subject 3: 74

Enter marks for Subject 4: 38

Enter marks for Subject 5: 85

--- Result ---

Student Name: Abc xyz

Total Marks: 326.0 out of 500

Percentage: 65.2%

**Data types in java**

**Java Data Type Categories**

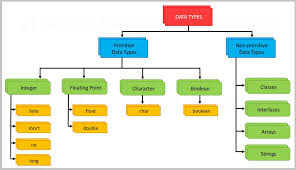
1. Primitive Data Type: These are the basic building blocks that store simple values directly in memory. Examples of primitive data types are

* boolean
* char
* byte
* short
* int
* long
* float
* double

*Note: The Boolean with uppercase B is a wrapper class for the primitive boolean type.*

2. Non-Primitive Data Types (Object Types): These are reference types that store memory addresses of objects. Examples of Non-primitive data types are

* String
* Array
* Class
* Interface
* Object



import java.util.Scanner;

public class StudentResult {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter number: ");

int num = input.nextInt();

if(num%2==0)

{

System.out.println("Even number"); }

else {

System.out.println("Odd number"); }

input.close();

} }

----------------------------------

import java.util.Scanner;

public class StudentResult {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter year: ");

int year = input.nextInt();

if(year%4==0) {

System.out.println("Leap year"); }

else {

System.out.println("Not Leap year"); }

input.close();

} }

---------------------------

import java.util.Scanner;

public class StudentResult {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter number: ");

int num = input.nextInt();

if(num%2==0) {

System.out.println("Multiple of 2."); }

else if(num%3==0) {

System.out.println("Multiple of 3"); }

else {

System.out.println("Not a multiple of 2 and 3"); }

input.close(); } }

---------------------------------

import java.util.Scanner;

public class StudentResult {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter number: ");

int num = input.nextInt();

int i=1;

while(i<=10) {

System.out.println(num + " x " + i + " = " + num\*i);

i++; }

input.close(); } }

Enter number: 8

8 x 1 = 8

8 x 2 = 16

8 x 3 = 24

8 x 4 = 32

8 x 5 = 40

8 x 6 = 48

8 x 7 = 56

8 x 8 = 64

8 x 9 = 72

8 x 10 = 80

--------------------------------------

public class DoWhileExample {

public static void main(String[] args) {

int i = 1;

do {

System.out.println("Number: " + i);

i++;

} while (i <= 5); } }

------------------------------

public class DoWhileExample {

public static void main(String[] args) {

// int i = 1;

for(int i=0;i<10;i++)

{

System.out.println(i); } } }

-----------------------

import java.util.Scanner;

public class ArithmeticSwitch {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter first number: ");

double num1 = scanner.nextDouble();

System.out.print("Enter second number: ");

double num2 = scanner.nextDouble();

System.out.println("Choose operation\n 1) Add.\n 2) Sub.\n 3) Mul. \n 4) Div.");

char operator = scanner.next().charAt(0);

double result;

switch (operator) {

case '1':

result = num1 + num2;

System.out.println("Result: " + result);

break;

case '2':

result = num1 - num2;

System.out.println("Result: " + result);

break;

case '3':

result = num1 \* num2;

System.out.println("Result: " + result);

break;

case '4':

if (num2 != 0) {

result = num1 / num2;

System.out.println("Result: " + result);

} else {

System.out.println("Error: Division by zero!"); }

break;

default:

System.out.println("Invalid operator."); }

scanner.close(); } }

Enter first number: 2

Enter second number: 6

Choose operation

1) Add.

2) Sub.

3) Mul.

4) Div.

3

Result: 12.0

--------------------------

import java.util.Scanner;

class abc

{ public void add(int a, int b)

{

int res=a+b;

System.out.println("Addition:" + res); }

}

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter first number: ");

int num1 = scanner.nextInt();

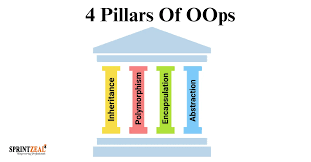
System.out.print("Enter second number: ");

int num2 = scanner.nextInt();

abc ab=new abc();

ab.add(num1,num2);

scanner.close(); } }



// Class with encapsulated fields

class Student {

// Private fields (encapsulated)

private String name;

private int age;

// Public getter method for name

public String getName() {

return name;

}

// Public setter method for name

public void setName(String name) {

this.name = name; }

// Public getter method for age

public int getAge() {

return age; }

// Public setter method for age

public void setAge(int age) {

if (age > 0) {

this.age = age;

} else {

System.out.println("Age must be positive."); } } }

// Main class to test encapsulation

public class Main {

public static void main(String[] args) {

Student student = new Student();

// Setting values using setters

student.setName("Aarav");

student.setAge(20);

// Getting values using getters

System.out.println("Student Name: " + student.getName());

System.out.println("Student Age: " + student.getAge()); } }

// Parent class (Superclass)

class Animal {

void eat() {

System.out.println("This animal eats food."); }

void sleep() {

System.out.println("This animal sleeps."); } }

// Child class (Subclass)

class Dog extends Animal {

void bark() {

System.out.println("The dog barks."); } }

// Main class to test inheritance

public class Main {

public static void main(String[] args) {

Dog myDog = new Dog();

// Inherited methods from Animal

myDog.eat();

myDog.sleep();

// Method from Dog class

myDog.bark(); }}

Override ex

// Superclass

class Animal {

void makeSound() {

System.out.println("Some generic animal sound"); } }

// Subclass

class Dog extends Animal {

// Overriding the makeSound method

@Override

void makeSound() {

System.out.println("Dog barks"); } }

// Main class to test overriding

public class Main {

public static void main(String[] args) {

Animal myAnimal = new Animal();

Animal myDog = new Dog(); // Polymorphism in action

myAnimal.makeSound(); // Calls Animal's method

myDog.makeSound(); // Calls Dog's overridden method

} }

// Abstract class

abstract class Shape {

// Abstract method (no body)

abstract void draw();

// Regular method

void display() {

System.out.println("Displaying shape..."); }}

// Subclass that extends abstract class

class Circle extends Shape {

// Providing implementation for abstract method

@Override

void draw() {

System.out.println("Drawing a circle"); } }

// Main class to test abstract class

public class Main {

public static void main(String[] args) {

Shape myShape = new Circle(); // Polymorphism

myShape.display(); // Calls regular method

myShape.draw(); // Calls overridden method } }

**Exception handling in Java**

In Java, an exception is an event that disrupts the normal flow of a program's instructions during execution. It is an object that describes an exceptional or error condition that has occurred in the code. Exceptions are a core part of Java's robust error handling mechanism, designed to prevent programs from crashing and to allow for graceful recovery from unexpected situations.

Exception handling in Java is a mechanism for managing runtime errors to ensure the normal flow of an application. An exception is an event that disrupts the typical execution sequence of a program. When an exception occurs, an exception object is created, containing information about the error.

* try block: This block encloses the code that might throw an exception.
* catch block: This block follows a try block and contains the code to handle a specific type of exception thrown within the try block. Multiple catch blocks can be used to handle different exception types.
* finally block: This block executes regardless of whether an exception was thrown or caught. It is typically used for resource cleanup, such as closing files or database connections.
* throw keyword: This keyword is used to explicitly throw an exception object.
* throws keyword: This keyword is used in a method signature to declare that the method might throw one or more specified checked exceptions.

public class Main {

public static void main(String[] args) {

try {

int a = 10;

int b = 0;

int result = a / b; // This will throw ArithmeticException

System.out.println("Result: " + result);

} catch (ArithmeticException e) {

System.out.println("Error: Cannot divide by zero!");

} finally {

System.out.println("This block always executes."); } } }

// Custom exception class

class InvalidAgeException extends Exception {

public InvalidAgeException(String message) {

super(message); } }

// Class that uses the custom exception

public class Main {

public static void main(String[] args) {

int age = 15;

try {

validateAge(age);

System.out.println("Age is valid.");

} catch (InvalidAgeException e) {

System.out.println("Exception caught: " + e.getMessage()); } }

// Method that throws custom exception

static void validateAge(int age) throws InvalidAgeException {

if (age < 18) {

throw new InvalidAgeException("Age must be 18 or above."); } } }

**Types of Exceptions:**

* **Checked Exceptions**:

These are exceptions that the compiler forces you to handle (e.g., IOException, SQLException). They must be caught or declared in the method signature using throws.

* **Unchecked Exceptions (Runtime Exceptions)**:

These are exceptions that the compiler does not require you to handle (e.g., NullPointerException, ArrayIndexOutOfBoundsException). While they can be handled, it's often more appropriate to fix the underlying programming error.

* **Errors**:

These represent serious problems that are typically irrecoverable and occur in the Java Virtual Machine (JVM), such as OutOfMemoryError or StackOverflowError. Errors are generally not handled by application code.

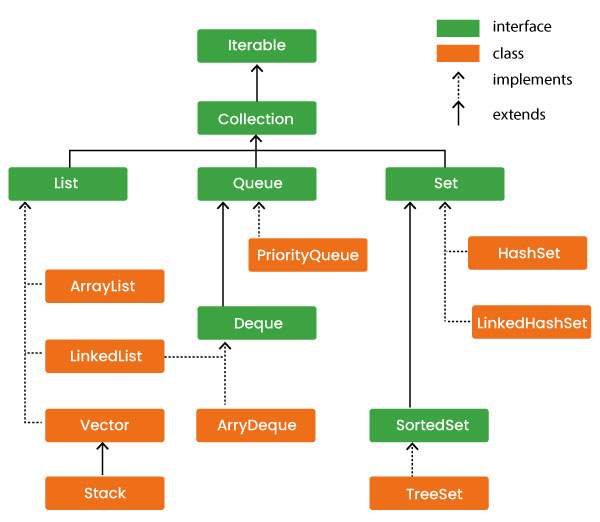
**Exception handling ways**

Try catch

Try multiple catch

Custom exception

Collection framework



**Applications of Array Data Structure:**

Arrays mainly have advantages like random access and cache friendliness over other data structures that make them useful.

Below are some applications of arrays.

* **Storing and accessing data**: Arrays store elements in a specific order and allow constant-time O(1) access to any element.
* **Searching**: If data in array is sorted, we can search an item in O(log n) time. We can also find floor(), ceiling(), kth smallest, kth largest, etc efficiently.
* **Matrices**: Two-dimensional arrays are used for matrices in computations like graph algorithms and image processing.
* **Implementing other data structures:** Arrays are used as the underlying data structure for implementing stacks and queues.
* **Dynamic programming**: Dynamic programming algorithms often use arrays to store intermediate results of subproblems in order to solve a larger problem.
* **Data Buffers:** Arrays serve as data buffers and queues, temporarily storing incoming data like network packets, file streams, and database results before processing.

**Advantages of Array Data Structure:**

* **Efficient and Fast Access:** Arrays allow direct and efficient access to any element in the collection with constant access time, as the data is stored in contiguous memory locations.
* **Memory Efficiency:** Arrays store elements in contiguous memory, allowing efficient allocation in a single block and does not require extra storage for linking different blocks.
* **Versatility:** Arrays can be used to store a wide range of data types, including integers, floating-point numbers, characters, and even complex data structures such as objects and pointers.
* **Compatibility with hardware:**The array data structure is compatible with most hardware architectures, making it a versatile tool for programming in a wide range of environments.

**Disadvantages of Array Data Structure:**

* **Fixed Size:** Arrays have a fixed size set at creation. Expanding an array requires creating a new one and copying elements, which is time-consuming and memory-intensive. Even dynamic sized arrays internally use fixed sized memory allocation and de-allocation.
* **Memory Allocation Issues:** Allocating large arrays can cause memory exhaustion, leading to crashes, especially on systems with limited resources.
* **Insertion and Deletion Challenges:** Adding or removing elements requires shifting subsequent elements, making these operations inefficient.

import java.util.\*;

public class Main {

public static void main(String[] args) {

List<String> cart = new ArrayList<>();

cart.add("Laptop");

cart.add("Phone");

cart.add("Phone");

cart.add("Headphone");

System.out.println("Shopping cart:");

for(String item: cart)

{

System.out.println(item); } } }

**ArrayList Vs vector**

| **S. No.** | **ArrayList** | **Vector** |
| --- | --- | --- |
| 1. | ArrayList is not synchronized. | Vector is synchronized. |
| 2. | ArrayList increments 50% of the current array size if the number of elements exceeds ts capacity. | Vector increments 100% means doubles the array size if the total number of elements exceeds its capacity. |
| 3. | ArrayList is not a legacy class. It is introduced in JDK 1.2. | Vector is a legacy class. |
| 4. | ArrayList is fast because it is non-synchronized. | Vector is slow because it is synchronized, i.e., in a multithreading environment, it holds the other threads in a runnable or non-runnable state until the current thread releases the lock of the object. |
| 5. | ArrayList uses the Iterator interface to traverse the elements. | A Vector can use the Iterator interface or Enumeration interface to traverse the elements. |
| 6 | ArrayList performance is high | Vector performance is low |
| 7 | Multiple threads is allowed | only one threads are allowed . |

import java.util.Vector;

public class Main {

public static void main(String[] args) {

// Creating a Vector of Strings

Vector<String> names = new Vector<>();

// Adding elements to the Vector

names.add("Aarav");

names.add("Diya");

names.add("Kabir");

names.add("Meera");

// Displaying the Vector elements

System.out.println("Student Names:");

for (String name : names) {

System.out.println(name); }

// Showing size of the Vector

System.out.println("Total students: " + names.size()); } }

List, Set, Queue,Vector

Inplement linked list add and remove element

import java.util.LinkedList;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

LinkedList<String> list = new LinkedList<>();

Scanner scanner = new Scanner(System.in);

int choice;

do {

System.out.println("\n--- Menu ---");

System.out.println("1. Add Element");

System.out.println("2. Remove Element");

System.out.println("3. Display List");

System.out.println("4. Exit");

System.out.print("Enter your choice: ");

choice = scanner.nextInt();

scanner.nextLine(); // Consume newline

switch (choice) {

case 1:

System.out.print("Enter element to add: ");

String elementToAdd = scanner.nextLine();

list.add(elementToAdd);

System.out.println("Added: " + elementToAdd);

break;

case 2:

System.out.print("Enter element to remove: ");

String elementToRemove = scanner.nextLine();

if (list.remove(elementToRemove)) {

System.out.println("Removed: " + elementToRemove);

} else {

System.out.println("Element not found."); }

break;

case 3:

System.out.println("Current List: " + list);

break;

case 4:

System.out.println("Exiting program...");

break;

default:

System.out.println("Invalid choice. Try again."); }

} while (choice != 4);

scanner.close(); } }

--- Menu ---

1. Add Element

2. Remove Element

3. Display List

4. Exit

Enter your choice: 1

Enter element to add: 5

Added: 5

--- Menu ---

1. Add Element

2. Remove Element

3. Display List

4. Exit

Enter your choice: 1

Enter element to add: 7

Added: 7

--- Menu ---

1. Add Element

2. Remove Element

3. Display List

4. Exit

Enter your choice: 1

Enter element to add: 5

Added: 5

--- Menu ---

1. Add Element

2. Remove Element

3. Display List

4. Exit

Enter your choice: 3

Current List: [5, 7, 5]

--- Menu ---

1. Add Element

2. Remove Element

3. Display List

4. Exit

Enter your choice: 2

Enter element to remove: 5

Removed: 5

--- Menu ---

1. Add Element

2. Remove Element

3. Display List

4. Exit

Enter your choice: 3

Current List: [7, 5]

--- Menu ---

1. Add Element

2. Remove Element

3. Display List

4. Exit

Enter your choice: 4

# Day 6 - Python

Python is a high-level, interpreted, object-oriented programming language created by Guido van Rossum and first released in 1991. It is known for its clear, readable syntax, which emphasizes code readability and allows developers to express concepts in fewer lines of code compared to other languages.

Python includes several built-in data types to store various kinds of data. These data types are categorized as follows:

1. Numeric Types:

* int (Integers): Whole numbers, positive or negative, without decimal points. Examples: 5, -10, 1000.
* float (Floating-point numbers): Numbers with decimal points. Examples: 3.14, -0.5, 2.0.
* complex (Complex numbers): Numbers with a real and an imaginary part, represented as a + bj. Example: 2 + 3j.

2. Sequence Types:

* str (Strings):

Sequences of characters enclosed in single quotes, double quotes, or triple quotes. Examples: "hello", 'Python', """multi-line string""".

* list (Lists):

Ordered, mutable collections of items enclosed in square brackets. Items can be of different data types. Example: [1, "apple", 3.14].

* tuple (Tuples):

Ordered, immutable collections of items enclosed in parentheses. Example: (1, "banana", 2.71).

* range (Ranges):

Immutable sequences of numbers, often used in loops. Example: range(5) generates numbers from 0 to 4.

3. Set Types:

* set (Sets): Unordered collections of unique items enclosed in curly braces. Duplicate elements are automatically removed. Example: {1, 2, 3, 2} results in {1, 2, 3}.
* frozenset (Frozen Sets): Immutable versions of sets.

4. Mapping Type:

* dict (Dictionaries): Unordered collections of key-value pairs enclosed in curly braces. Keys must be unique and immutable. Example: {"name": "Alice", "age": 30}.

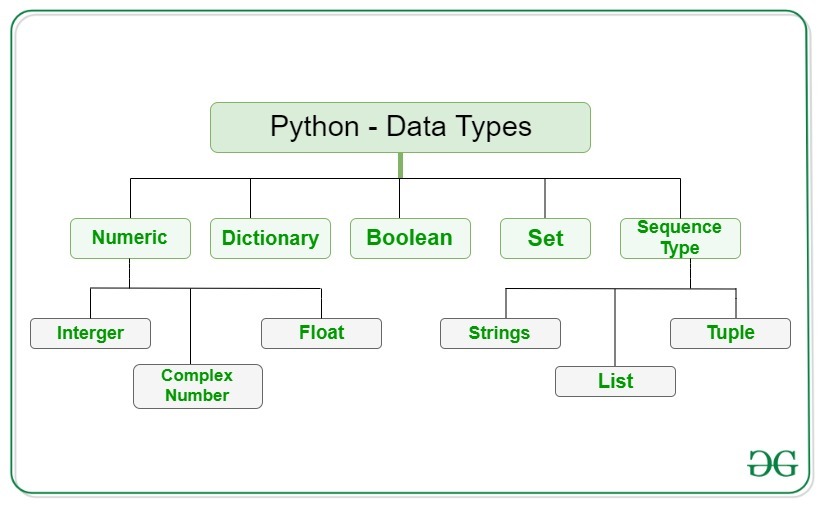
5. Boolean Type:

* bool (Booleans): Represents truth values, either True or False. Used for logical operations.

6. Binary Types:

* bytes: Immutable sequences of bytes.
* bytearray: Mutable sequences of bytes.
* memoryview: Provides a memory-efficient way to access the internal data of an object without copying it.

7. None Type:

* NoneType: Represents the absence of a value. The only value of this type is None.
* 

# Define variables

a = 10

b = 3.14

c = "Hello"

d = True

e = [1, 2, 3]

f = {'name': 'Alice', 'age': 25}

# Display their data types

print("Type of a:", type(a))

print("Type of b:", type(b))

print("Type of c:", type(c))

print("Type of d:", type(d))

print("Type of e:", type(e))

print("Type of f:", type(f))

Type of a: <class 'int'>

Type of b: <class 'float'>

Type of c: <class 'str'>

Type of d: <class 'bool'>

Type of e: <class 'list'>

Type of f: <class 'dict'>

| **Parameters** | **High-Level Language** | **Low-Level Language** |
| --- | --- | --- |
| **Abstraction Level** | High abstraction, closer to human language | Low abstraction, closer to machine code |
| **Ease of Use** | Easier to learn and use | More complex and harder to learn |
| **Portability** | Highly portable across different systems | Less portable, often system-specific |
| **Development Speed** | Faster development time | Slower development time |
| **Examples** | Python, Java, C++, JavaScript | Assembly language, Machine code |
| **Memory Management** | Automatic memory management | Manual memory management |
| **Error Handling** | Built-in error handling features | Limited error handling, requires manual checks |
| **Performance** | Generally slower execution | Generally faster execution |
| **Use Cases** | Application development, scripting, web development | System programming, embedded systems, device drivers |

Python has a set of keywords that are reserved words that cannot be used as variable names, function names, or any other identifiers:

|  |  |
| --- | --- |
| **Keyword** | **Description** |
| [and](https://www.w3schools.com/python/ref_keyword_and.asp) | A logical operator |
| [as](https://www.w3schools.com/python/ref_keyword_as.asp) | To create an alias |
| [assert](https://www.w3schools.com/python/ref_keyword_assert.asp) | For debugging |
| [break](https://www.w3schools.com/python/ref_keyword_break.asp) | To break out of a loop |
| [class](https://www.w3schools.com/python/ref_keyword_class.asp) | To define a class |
| [continue](https://www.w3schools.com/python/ref_keyword_continue.asp) | To continue to the next iteration of a loop |
| [def](https://www.w3schools.com/python/ref_keyword_def.asp) | To define a function |
| [del](https://www.w3schools.com/python/ref_keyword_del.asp) | To delete an object |
| [elif](https://www.w3schools.com/python/ref_keyword_elif.asp) | Used in conditional statements, same as else if |
| [else](https://www.w3schools.com/python/ref_keyword_else.asp) | Used in conditional statements |
| [except](https://www.w3schools.com/python/ref_keyword_except.asp) | Used with exceptions, what to do when an exception occurs |
| [False](https://www.w3schools.com/python/ref_keyword_false.asp) | Boolean value, result of comparison operations |
| [finally](https://www.w3schools.com/python/ref_keyword_finally.asp) | Used with exceptions, a block of code that will be executed no matter if there is an exception or not |
| [for](https://www.w3schools.com/python/ref_keyword_for.asp) | To create a for loop |
| [from](https://www.w3schools.com/python/ref_keyword_from.asp) | To import specific parts of a module |
| [global](https://www.w3schools.com/python/ref_keyword_global.asp) | To declare a global variable |
| [if](https://www.w3schools.com/python/ref_keyword_if.asp) | To make a conditional statement |
| [import](https://www.w3schools.com/python/ref_keyword_import.asp) | To import a module |
| [in](https://www.w3schools.com/python/ref_keyword_in.asp) | To check if a value is present in a list, tuple, etc. |
| [is](https://www.w3schools.com/python/ref_keyword_is.asp) | To test if two variables are equal |
| [lambda](https://www.w3schools.com/python/ref_keyword_lambda.asp) | To create an anonymous function |
| [None](https://www.w3schools.com/python/ref_keyword_none.asp) | Represents a null value |
| [nonlocal](https://www.w3schools.com/python/ref_keyword_nonlocal.asp) | To declare a non-local variable |
| [not](https://www.w3schools.com/python/ref_keyword_not.asp) | A logical operator |
| [or](https://www.w3schools.com/python/ref_keyword_or.asp) | A logical operator |
| [pass](https://www.w3schools.com/python/ref_keyword_pass.asp) | A null statement, a statement that will do nothing |
| [raise](https://www.w3schools.com/python/ref_keyword_raise.asp) | To raise an exception |
| [return](https://www.w3schools.com/python/ref_keyword_return.asp) | To exit a function and return a value |
| [True](https://www.w3schools.com/python/ref_keyword_true.asp) | Boolean value, result of comparison operations |
| [try](https://www.w3schools.com/python/ref_keyword_try.asp) | To make a try...except statement |

**List data type**

**Dictionary**

**Set**

Control structure

# Take input from user

num1 = float(input("Enter first number: "))

num2 = float(input("Enter second number: "))

# Perform arithmetic operations

print("Addition:", num1 + num2)

print("Subtraction:", num1 - num2)

print("Multiplication:", num1 \* num2)

# Handle division carefully

if num2 != 0:

print("Division:", num1 / num2)

else:

print("Division: Cannot divide by zero")

Enter first number: 4

Enter second number: 7

Addition: 11.0

Subtraction: -3.0

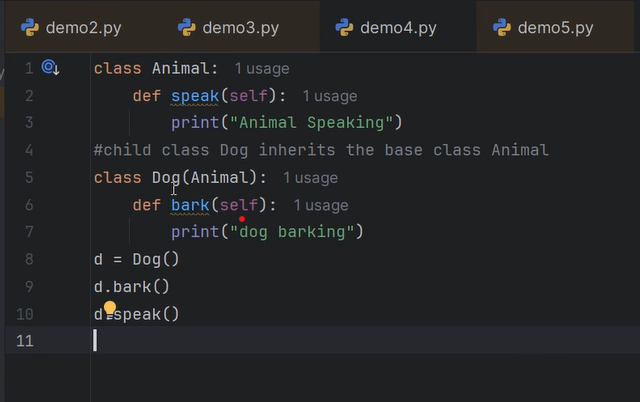
Multiplication: 28.0

Division: 0.5714285714285714

Create obj, constructer , \_\_init\_\_

Derived class

Inheritnce



Create multi level inheritance example

class Person:

def show\_identity(self):

print("I am a Abc.")

class Age(Person):

def show\_age(self):

print("I am 20yrs.")

class Hobby(Age):

def show\_hobby(self):

print("I like to watch movies. ")

mov = Hobby()

mov.show\_identity()

mov.show\_age()

mov.show\_hobby()

I am a Abc.

I am 20yrs.

I like to watch movies.

Polymorphism example

# Day 7 – Python

**Exception handling**

Python Exception Handling allows a program to gracefully deal with unexpected events (like invalid input or missing files) without crashing. Instead of terminating abruptly, Python lets you detect the problem, respond to it, and continue execution when possible.

n = 10

try:

res = n / 0

except ZeroDivisionError:

print("Can't be divided by zero!")

Errors and exceptions are both issues in a program, but they differ in severity and handling. Let's see how:

* **Error:**Serious problems in the program logic that cannot be handled. Examples include syntax errors or memory errors.
* **Exception:** Less severe problems that occur at runtime and can be managed using exception handling (e.g., invalid input, missing files).

# Simple ATM withdrawal simulation with exception handling

balance = 5000 # Initial account balance

try:

amount = float(input("Enter amount to withdraw: "))

if amount <= 0:

raise ValueError("Amount must be greater than zero.")

if amount > balance:

raise Exception("Insufficient balance.")

balance -= amount

print(f"Withdrawal successful! Remaining balance: ₹{balance}")

except ValueError as ve:

print("Invalid input:", ve)

except Exception as e:

print("Transaction error:", e)

Enter amount to withdraw: 200

Withdrawal successful! Remaining balance: ₹4800.0

**File handling in Python**

File handling refers to the process of performing operations on a file, such as creating, opening, reading, writing and closing it through a programming interface. It involves managing the data flow between the program and the file system on the storage device, ensuring that data is handled safely and efficiently.

**Why do we need File Handling**

* To store data permanently, even after the program ends.
* To access external files like .txt, .csv, .json, etc.
* To process large files efficiently without using much memory.
* To automate tasks like reading configs or saving outputs.
* To handle input/output in real-world applications and tools.

**Opening a File**

To open a file, we can use [open()](https://www.geeksforgeeks.org/python/python-open-function/)function, which requires file-path and mode as arguments:

**Syntax:**

*file = open('filename.txt', 'mode')*

* **filename.txt:**name (or path) of the file to be opened.
* **mode:** mode in which you want to open the file (read, write, append, etc.).

***Note:*** *If you don’t specify the mode, Python uses* ***'r'*** *(read mode) by default.*

**Basic Example: Opening a File**

f = open("geek.txt", "r")

print(f)

Explanation: This code opens file **geek.txt** in read mode. If the file exists, it returns a file object connected to that file; if the file does not exist, Python raises a FileNotFoundError.

**Closing a File**

The file.close() method closes the file and releases the system resources. If the file was opened in write or append mode, closing ensures that all changes are properly saved.

file = open("geek.txt", "r")

# Perform file operations

file.close()

We will also see later how closing can be handled automatically using the with statement and how to ensure files close properly using exception handling.

**Checking File Properties**

Once the file is open, we can check some of its properties:

f = open("geek.txt", "r")

print("Filename:", f.name)

print("Mode:", f.mode)

print("Is Closed?", f.closed)

​

f.close()

print("Is Closed?", f.closed)

**Output:**

*Filename: geek.txt  
Mode: r  
Is Closed? False  
Is Closed? True*

**Explanation:**

* **f.name:** Returns the name of the file that was opened (in this case, "geek.txt").
* **f.mode:** Tells us the mode in which the file was opened. Here, it’s 'r' which means read mode.
* **f.closed:**Returns a boolean value- False when file is currently open otherwise True.

**Reading a File**

[Reading a file](https://www.geeksforgeeks.org/python/how-to-read-from-a-file-in-python/) can be achieved by **file.read()** which reads the entire content of the file. After reading, it’s good practice to close the file to free up system resources.

**Example:** Reading a File in Read Mode (r)

file = open("geek.txt", "r")

content = file.read()

print(content)

file.close()

**Output:**

*Hello world  
GeeksforGeeks  
123 456*

**Writing a File**

In Python, [writing to a file](https://www.geeksforgeeks.org/python/writing-to-file-in-python/) is done using the mode "w". This creates a new file if it doesn’t exist, or overwrites the existing file if it does. The write() method is used to add content. After writing, make sure to close the file.

**Example:** Writing to a file (overwrites if file exists)

with open("geek.txt", "w") as file:

file.write("Hello, Python!\n")

file.write("File handling is easy with Python.")

​

print("File written successfully")

**Output:**

*Hello, Python!  
File handling is easy with Python.*

**Explanation:**

* "w" mode opens the file for writing (overwrites existing content if the file already exists).
* write() method adds new text to the file.
* When using with, the file closes automatically at the end of the block.

**Using with Statement**

Instead of manually opening and closing the file, you can use the [with statement](https://www.geeksforgeeks.org/python/with-statement-in-python/), which automatically handles closing. This reduces the risk of file corruption and resource leakage.

**Example:**Let's assume we have a file named **geek.txt** that contains text "**Hello, World!**".

with open("geek.txt", "r") as file:

content = file.read()

print(content)

**Output:**

*Hello, World!*

**Handling Exceptions When Closing a File**

It's important to [handle exceptions](https://www.geeksforgeeks.org/python/python-exception-handling/) to ensure that files are closed properly, even if an error occurs during file operations. Here, the finally block ensures the file is closed even if an error occurs.

try:

file = open("geek.txt", "r")

content = file.read()

print(content)

finally:

file.close()

**Output:**

*Hello, World!*

**Explanation:**

* **try:** Starts the block to handle code that might raise an error.
* **open():** Opens the file in read mode.
* **read():** Reads the content of the file.
* **finally:** Ensures the code inside it runs no matter what.

# Writing to a file

with open("sample.txt", "w") as file:

file.write("Hello, this is the first line.\n")

file.write("File handling in Python is easy!\n")

# Reading from the file

with open("sample.txt", "r") as file:

content = file.read()

print("Reading file content:")

print(content)

# Appending to the file

with open("sample.txt", "a") as file:

file.write("This line was added later.\n")

# Reading again to confirm append

with open("sample.txt", "r") as file:

updated\_content = file.read()

print("\nUpdated file content:")

print(updated\_content)

Hello, this is the first line.

File handling in Python is easy!

This line was added later.

**List Comprehension**

List comprehension offers a shorter syntax when you want to create a new list based on the values of an existing list.

Example:

Based on a list of fruits, you want a new list, containing only the fruits with the letter "a" in the name.

Without List comprehension

# Input list of numbers

numbers = [1, 2, 3, 4, 5]

# Create an empty list to store squares

squares = []

# Use a for loop to calculate squares

for num in numbers:

squares.append(num \*\* 2)

# Print the result

print("Squares:", squares)

Squares: [1, 4, 9, 16, 25]

print("Even numbers from 1 to 11:")

for i in range(1, 12):

if i % 2 == 0:

print(i)

2

4

6

8

10

# Input string

text = "Hello, Python is fun!"

# Vowels to check

vowels = "aeiouAEIOU"

# Empty string to store vowels

found\_vowels = ""

# Loop through each character

for char in text:

if char in vowels:

found\_vowels += char

# Print the result

print("Vowels in the string:", found\_vowels)

Vowels in the string: eooiu

Flatten 2d array

matrix = [[1,2],[3,4],[5,6]]

flat = [num for row in matrix for num in row]

print(flat)

[1, 2, 3, 4, 5, 6]

C to F

celsius = [0,10,20,30]

farenheit = [(temp\* 9/5)+32 for temp in celsius]

print(farenheit)

[32.0, 50.0, 68.0, 86.0]

**Decorator**

In Python, decorators are flexible way to modify or extend behavior of functions or methods, without changing their actual code.

* A decorator is essentially a function that takes another function as an argument and returns a new function with enhanced functionality.
* Decorators are often used in scenarios such as logging, authentication and memorization, allowing us to add additional functionality to existing functions or methods in a clean, reusable way.

Logging Decorator example

import datetime

# Decorator to log function calls with timestamp

def log(func):

def wrapper(\*args, \*\*kwargs):

print(f"[{datetime.datetime.now()}] Calling {func.\_\_name\_\_}...")

return func(\*args, \*\*kwargs)

return wrapper

# Apply the decorator correctly

@log

def process\_data(data):

print(f"Processing {data}")

# Call the function

process\_data("User Records")

[2025-09-08 06:53:52.015296] Calling process\_data...

Processing User Records

**Unit Testing**

Unit Testing is the first level of software testing where the smallest testable parts of software are tested. This is used to validate that each software unit performs as designed. The unittest test framework is Python xUnit style framework. In this article, we will learn about unittest framework with the help of examples.

Single code

import unittest

# Functions to be tested

def add(a, b):

return a + b

def subtract(a, b):

return a - b

# Unit tests

class TestMathFunctions(unittest.TestCase):

def test\_add(self):

self.assertEqual(add(10, 5), 15)

self.assertEqual(add(-1, 1), 0)

self.assertEqual(add(0, 0), 0)

def test\_subtract(self):

self.assertEqual(subtract(10, 5), 5)

self.assertEqual(subtract(5, 10), -5)

self.assertEqual(subtract(-3, -2), -1)

# Run the tests

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

>..

----------------------------------------------------------------------

Ran 2 tests in 0.000s

OK

Code file n test file diff

# calculator.py

def add(a, b):

"""Returns the sum of a and b."""

return a + b

def subtract(a, b):

"""Returns the result of subtracting b from a."""

return a – b

# test\_calculator.py

import unittest

from calculator import add, subtract

class TestCalculator(unittest.TestCase):

def test\_add(self):

self.assertEqual(add(2, 3), 5)

self.assertEqual(add(-1, 1), 0)

self.assertEqual(add(0, 0), 0)

def test\_subtract(self):

self.assertEqual(subtract(5, 3), 2)

self.assertEqual(subtract(0, 5), -5)

self.assertEqual(subtract(-3, -2), -1)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**Python Libraries**

import matplotlib.pyplot as plt

# Sample data

x = [1, 2, 3, 4, 5]

y = [2, 4, 6, 8, 10]

# Create the plot

plt.plot(x, y, marker='o', linestyle='-', color='blue')

# Add labels and title

plt.xlabel('X Axis')

plt.ylabel('Y Axis')

plt.title('Simple Line Graph')

# Show the graph

plt.grid(True)

plt.show()



import pandas as pd

import matplotlib.pyplot as plt

# Load dataset from a URL

url = "https://raw.githubusercontent.com/mwaskom/seaborn-data/master/iris.csv"

data = pd.read\_csv(url)

# Display first few rows

print(data.head())

# Visualize: Scatter plot of petal length vs petal width

plt.figure(figsize=(8, 6))

plt.scatter(data['petal\_length'], data['petal\_width'], c='green', alpha=0.6)

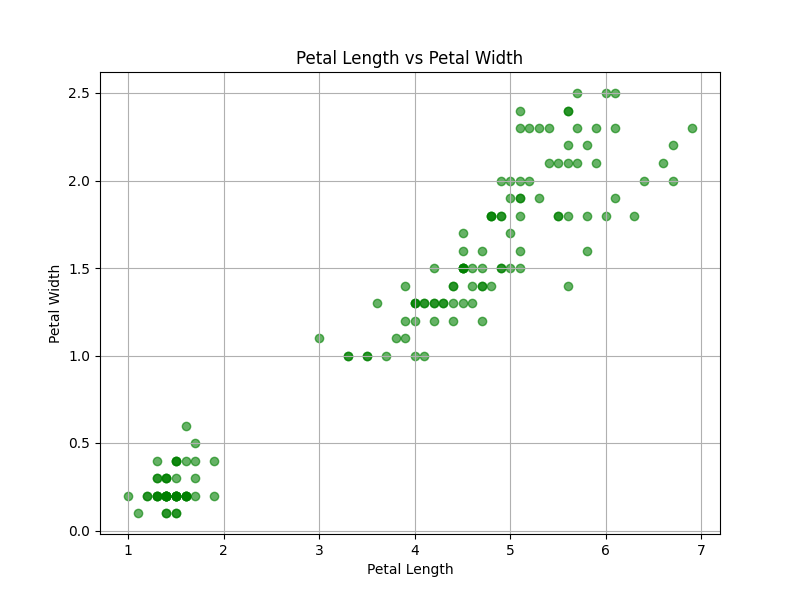
plt.title('Petal Length vs Petal Width')

plt.xlabel('Petal Length')

plt.ylabel('Petal Width')

plt.grid(True)

plt.show()



# Day 8 – Testing

Selenium - TestNg

Annotations

@Before test ( executes only once before test case)

@After test

Enable=false (disable any test)

@Before method (executes before every test case)

@After method

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Group  
it has test case in it, and when we specify the grp name, those test cases will run.

A screenshot of a computer

AI-generated content may be incorrect.

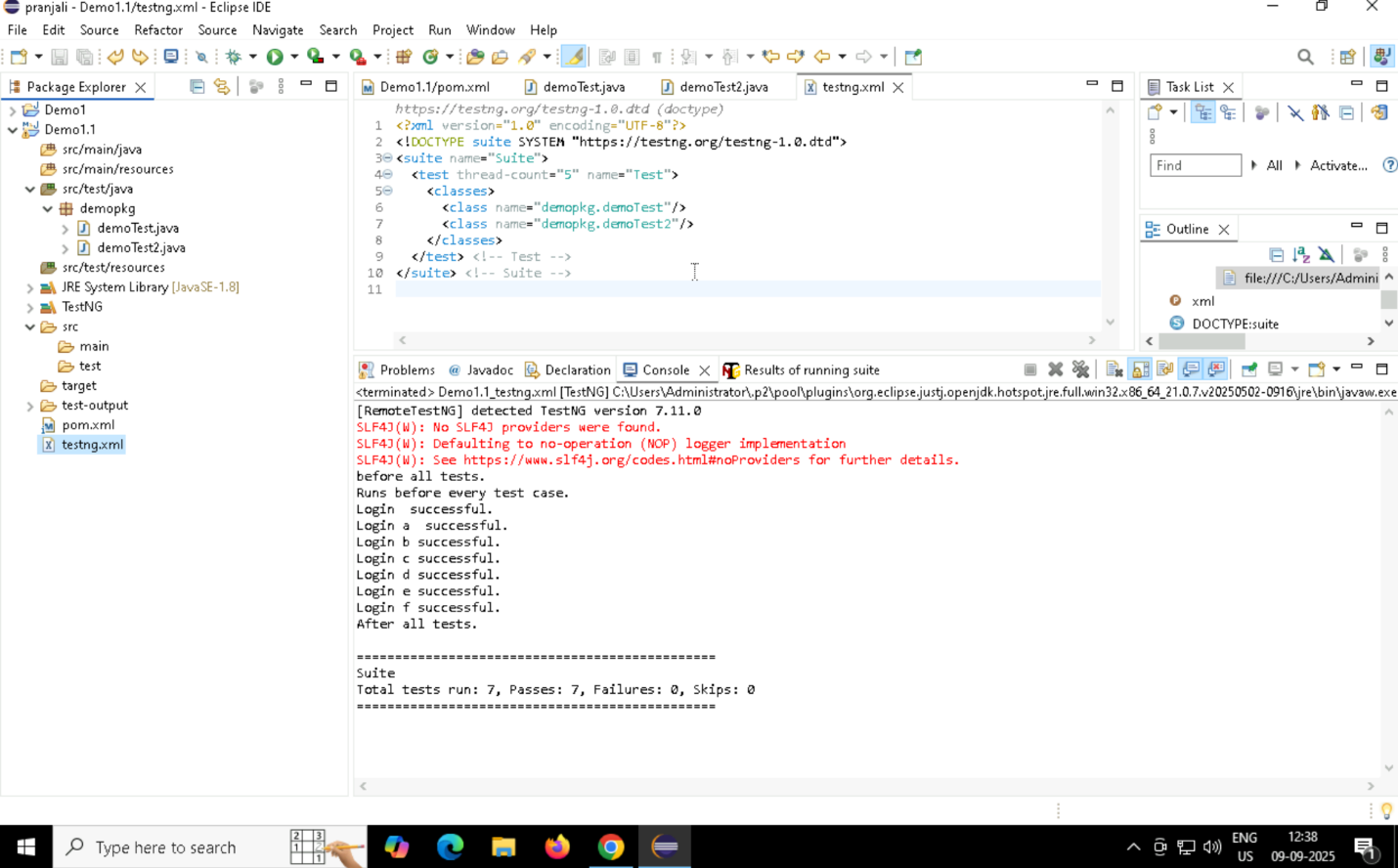
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Both classes together in the package



A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

Hard assert

Stops after failed test case.

Soft assert

Runs all test cases

# Day 9 – GIT

**What is Git?**

Git is a popular version control system.

It was created by Linus Torvalds in 2005, and has been maintained by Junio Hamano since then.

It is used for:

* Tracking code changes
* Tracking who made changes
* Coding collaboration

**Key Git Concepts**

* **Repository:** A folder where Git tracks your project and its history.
* **Clone:** Make a copy of a remote repository on your computer.
* **Stage:** Tell Git which changes you want to save next.
* **Commit:** Save a snapshot of your staged changes.
* **Branch:** Work on different versions or features at the same time.
* **Merge:** Combine changes from different branches.
* **Pull:** Get the latest changes from a remote repository.
* **Push:** Send your changes to a remote repository.

Working with Git

* Initialize Git on a folder, making it a **Repository**
* Git now creates a hidden folder to keep track of changes in that folder
* When a file is changed, added or deleted, it is considered **modified**
* You select the modified files you want to **Stage**
* The **Staged** files are **Committed**, which prompts Git to store a **permanent** snapshot of the files
* Git allows you to see the full history of every commit.
* You can revert back to any previous commit.
* Git does not store a separate copy of every file in every commit, but keeps track of changes made in each commit!

Why Git?

* Over 70% of developers use Git!
* Developers can work together from anywhere in the world.
* Developers can see the full history of the project.
* Developers can revert to earlier versions of a project.

GIT commands for set up

Python code on employee data for curd function

# Employee Management System (CRUD Operations)

employees = []

next\_id = 1 # Auto-increment employee ID

def add\_employee():

global next\_id

name = input("Enter employee name: ")

department = input("Enter department: ")

salary = float(input("Enter salary: "))

employee = {

'id': next\_id,

'name': name,

'department': department,

'salary': salary

}

employees.append(employee)

next\_id += 1

print("Employee added successfully!\n")

def view\_employees():

if not employees:

print("No employees found.\n")

return

print("\n Employee List:")

for emp in employees:

print(f"ID: {emp['id']}, Name: {emp['name']}, Dept: {emp['department']}, Salary: ${emp['salary']}")

print()

def update\_employee():

emp\_id = int(input("Enter employee ID to update: "))

for emp in employees:

if emp['id'] == emp\_id:

print(f"Editing Employee {emp\_id}: {emp['name']}")

emp['name'] = input("Enter new name: ") or emp['name']

emp['department'] = input("Enter new department: ") or emp['department']

salary\_input = input("Enter new salary: ")

if salary\_input:

emp['salary'] = float(salary\_input)

print(" Employee updated successfully!\n")

return

print(" Employee not found.\n")

def delete\_employee():

emp\_id = int(input("Enter employee ID to delete: "))

for emp in employees:

if emp['id'] == emp\_id:

employees.remove(emp)

print("🗑️ Employee deleted successfully!\n")

return

print(" Employee not found.\n")

def show\_menu():

print("======= Employee Management System =======")

print("1. Add Employee")

print("2. View Employees")

print("3. Update Employee")

print("4. Delete Employee")

print("5. Exit")

print("==========================================")

def main():

while True:

show\_menu()

choice = input("Enter your choice (1-5): ")

if choice == '1':

add\_employee()

elif choice == '2':

view\_employees()

elif choice == '3':

update\_employee()

elif choice == '4':

delete\_employee()

elif choice == '5':

print("Exiting the program.")

break

else:

print(" Invalid choice. Please try again.\n")

if \_\_name\_\_ == "\_\_main\_\_":

main()

======= Employee Management System =======

1. Add Employee

2. View Employees

3. Update Employee

4. Delete Employee

5. Exit

==========================================

Enter your choice (1-5): 1

Enter employee name: Abc

Enter department: It

Enter salary: 400000

Employee added successfully!

======= Employee Management System =======

1. Add Employee

2. View Employees

3. Update Employee

4. Delete Employee

5. Exit

==========================================

Enter your choice (1-5): 2

Employee List:

ID: 1, Name: Abc, Dept: It, Salary: $400000.0

======= Employee Management System =======

1. Add Employee

2. View Employees

3. Update Employee

4. Delete Employee

5. Exit

==========================================

Enter your choice (1-5): 3

Enter employee ID to update: 1

Editing Employee 1: Abc

Enter new name: Xyz

Enter new department: IT

Enter new salary: 39000

Employee updated successfully!

======= Employee Management System =======

1. Add Employee

2. View Employees

3. Update Employee

4. Delete Employee

5. Exit

==========================================

Enter your choice (1-5): 4

Enter employee ID to delete: 1

Employee deleted successfully!

======= Employee Management System =======

1. Add Employee

2. View Employees

3. Update Employee

4. Delete Employee

5. Exit

==========================================

Enter your choice (1-5): 5

Exiting the program.

# Day 10 – TDD

**Junit**

JUnit is a widely used testing framework in Java for writing and running unit tests. It helps developers ensure their code works as expected by testing individual components in isolation.

**Junit5 architecture**

1. JUnit Platform – the foundation that launches testing frameworks on the JVM.
2. JUnit Jupiter – provides new annotations and APIs for writing tests.
3. JUnit Vintage – supports running older JUnit 3 and 4 tests on the JUnit 5 platform.

**Common Junit Annotation**

* @Test – marks a method as a test method.
* @BeforeEach – runs before each test method.
* @AfterEach – runs after each test method.
* @BeforeAll – runs once before all tests.
* @AfterAll – runs once after all tests.
* @Disabled – disables a test method or class.

These annotations help manage test setup, teardown, and organization.

Difference in TDD and BDD

| **Behavior Driven Development** | **Test Driven Development** |
| --- | --- |
| Behavior Driven Development is a development technique which focuses more on a software application's behavior. | Test Driven Development is a development technique which focuses more on the implementation of a feature of a software application/product. |
| In BDD the participants are Developers, Customer, QAs. | In TDD the participants are developers. |
| Mainly it creates an executable specification that fails because the respective feature doesn't exist, then writing the simplest code that can make the specification pass and as a result we get the required behavior implemented in the system. | Mainly it refers to write a test case that fails because the specified functionality doesn't exist and after that update the code that can make the test case pass and as a result we get the feature implemented in the system. |
| Its main focus is on system requirements. | Its main focus is on unit test. |
| In BDD the starting point is a scenario. | In TDD the starting point is a test case. |
| It is a team methodology. | It is a development practice. |
| Here language used to write behavior/scenarios is simple English language. | Here language is used is similar to the one used for feature development like programming language. |
| In BDD collaboration is required between all the stakeholders. | In TDD collaboration is required only between the developers. |
| It is a good approach for project development which are driven by user actions. | It is a good approach for projects which involve API and third-party tools. |
| Some of the tools used are  Cucumber, Dave, JBehave, Spec Flow,  Concordian, BeanSpec etc. | Some of the tools used are  JBehave, JDave, Cucumber, Spec Flow, BeanSpec, FitNesse etc. |

# Day 11 – Webservices

**What is a web service?**

A web service is a software module that performs specific functions and enables communication between client and server applications over a network, typically the Internet. It uses standardized web protocols like HTTP/HTTPS to exchange data, often in formats like XML. Web services are platform-independent, allowing applications written in different programming languages to interact. In cloud computing, web services can be discovered and invoked over the network to provide functionality across systems, similar to how processes communicate within a single computer.

Why do we need webservice?

**Functions of Web Services**

* It's possible to access it via the internet or intranet networks.
* XML messaging protocol that is standardized.
* Operating system or programming language independent.
* Using the XML standard, it is self-describing.
* A simple location approach can be used to locate it.

**XML (Extensible Markup Language)** is a **markup language** used to **store and transport data** in a structured, readable, and platform-independent format.

**Key Features of XML:**

* **Extensible**: You can create your own custom tags to define data.
* **Self-descriptive**: The data is stored along with a description of what it is.
* **Platform-independent**: Works across different systems and technologies.
* **Text-based**: Data is stored as plain text, making it easy to read and share.

**SOAP-based Web Services**

SOAP (Simple Object Access Protocol) is a protocol for exchanging structured information in the implementation of web services. It relies on XML for message formatting and is based on a client-server model. SOAP web services are known for their robustness and standardization.**Key Characteristics:**

1. **Protocol and Standards:**

* SOAP is a protocol with a well-defined structure and interface, described using WSDL (Web Services Description Language). It includes standards such as WS-Security, WS-AtomicTransaction, and WS-ReliableMessaging
* SOAP messages are always in XML format, ensuring platform and language independence

**2.Message Exchange:**

* SOAP uses XML for both request and response messages. This ensures that the data is structured and can be validated against a schema
* In case of errors, SOAP provides detailed error information through SOAP Faults.

**3. Service Creation with Spring Boot:**

* Spring Boot provides a developer-friendly framework for creating SOAP web services. Developers can define SOAP-based services and generate Java code from XSD (XML Schema Definition) files
* The combination of Spring Boot, Spring Web Services, and JPA makes developing SOAP web services efficient and enjoyable

**Advantages:**

* **Standardization:** SOAP has a well-defined set of standards, making it suitable for enterprise-level applications that require robust security and transactional support.
* **Extensibility:** SOAP’s extensibility allows for the addition of new features and standards without breaking existing implementations.

**RESTful Web Services**

REST (Representational State Transfer) is an architectural style for designing networked applications. RESTful web services are known for their simplicity, scalability, and flexibility.**Key Characteristics:**

1. **Architectural Style:**

* REST is not a protocol but an architectural style that uses standard HTTP methods (GET, POST, PUT, DELETE) for communication
* Resources in RESTful services are identified by URIs (Uniform Resource Identifiers) and can be manipulated using standard HTTP methods.

2.**Message Formats:**

* RESTful services can use multiple formats for data exchange, including JSON, XML, HTML, and plain text. JSON is the most commonly used format due to its lightweight nature and ease of use.

3. **Service Creation with Spring Boot:**

* Spring Boot, in combination with Spring Web MVC (also known as Spring REST), simplifies the development of RESTful web services. It provides tools and annotations to create RESTful endpoints efficiently
* RESTful services are often the first step in developing microservices, making them a crucial part of modern application architectures

**Advantages:**

* **Simplicity:** RESTful services are simple to implement and use, making them ideal for web and mobile applications.
* **Scalability:** The stateless nature of RESTful services allows for easy scaling and load balancing.

A computer and a diagram

AI-generated content may be incorrect.

|  |  |
| --- | --- |
| **Advantages** | **Description** |
| Platform & Language Independence | Enables communication between applications written in different languages and platforms. |
| Interoperability | Allows different systems to work together, even if built with different technologies. |
| Reusability | Services can be reused across multiple applications, reducing duplication. |
| Modularity & Maintainability | Easier to update or replace specific functionality without affecting the entire system. |
| Scalability | Individual services can be scaled independently based on demand. |
| Loose Coupling | Changes in one part (e.g., server) don’t require changes in the client. |
| Accessibility Over the Internet | Services can be accessed from anywhere with a network connection. |

|  |  |
| --- | --- |
| **Disadvantages** | **Description** |
| Performance Overhead | XML/SOAP adds processing time compared to direct communication methods. |
| Complexity | Implementation can be complex, especially with protocols like SOAP. |
| Security Risks | Exposing services online increases vulnerability to attacks if not secured properly. |
| Lack of State Management | Most services are stateless, making it harder to maintain sessions. |
| Debugging Difficulty | Errors in distributed systems are harder to trace and resolve. |
| Dependency on Network | Requires stable network/internet connection to function reliably. |

WSDL (Web Services Description Language) and UDDI (Universal Description, Discovery, and Integration) are both XML-based standards used in the context of Web Services, but they serve distinct purposes:

WSDL (Web Services Description Language):

* **Purpose:**

WSDL is used to describe the technical interface of a web service. It defines what a web service does, how to access it, and what data formats it uses.

* **Content:**

A WSDL document specifies the operations offered by a web service, the input and output parameters for each operation, the data types involved, and the network protocols and message formats (bindings) used to communicate with the service.

* **Role:**

It acts as a contract between the service provider and the service consumer, enabling clients to understand how to interact with the service programmatically.

UDDI (Universal Description, Discovery, and Integration):

* **Purpose:**

UDDI serves as a distributed registry or directory for web services. It allows businesses to publish information about their web services and enables other businesses to discover and locate these services.

* **Content:**

A UDDI registry stores information about businesses (e.g., name, contact information), the services they offer (including their technical descriptions, often referencing WSDL documents), and categorization information to facilitate discovery.

* **Role:**

It facilitates the discovery of web services by providing a centralized or distributed repository where service providers can advertise their services and service consumers can search for services that meet their needs.

In summary:

* **WSDL describes a specific web service's interface and functionality.** It's like the detailed instruction manual for a particular appliance.
* **UDDI is a registry where businesses can publish and discover web services.** It's like a yellow pages directory where you can find different types of appliances and their respective instruction manuals (WSDLs).

What is API?

An API, or Application Programming Interface, is a set of rules and protocols that allows different software applications to communicate and interact with each other. It acts as an intermediary, defining how one piece of software can request services or data from another, and how the response will be delivered.

Types of APIs by Access:

* **Public APIs (Open APIs):**

These APIs are publicly available for anyone to use, often with minimal or no restrictions, and are typically well-documented. Examples include weather APIs or social media platform APIs.

* **Partner APIs:**

These APIs are exposed to specific business partners, requiring authentication and authorization. They facilitate integration between collaborating organizations.

* **Private APIs (Internal APIs):**

Designed for internal use within an organization, these APIs connect different systems or departments within the same company. They are not exposed to external users or partners.

* **Composite APIs:**

These APIs combine multiple service calls into a single request, simplifying complex interactions and improving efficiency for clients.

APIs work in a simple step-by-step process:

* **Request:** A client (user) sends a request through the API's URI (Uniform Resource Identifier).
* **Processing:** The API forwards the request to the server.
* **Response:** The server processes the request and sends the response back to the API.
* **Delivery:** The API returns the server's response to the client.

**Advantages of APIs**

* **Efficiency:**API produces efficient, quicker, and more reliable results than the outputs produced by human beings in an organization.
* **Flexible delivery of services:**API provides fast and flexible delivery of services according to developers' requirements.
* **Integration:**The best feature of API is that it allows the movement of data between various sites and thus enhances the integrated user experience.
* **Automation:**As API makes use of robotic computers rather than humans, it produces better and more automated results.
* **New functionality**: While using API the developers find new tools and functionality for API exchanges.

**Disadvantages of APIs**

* **Cost:**Developing and implementing API is costly at times and requires high maintenance and support from developers.
* **Security issues:**Using API adds another layer of surface which is then prone to attacks, and hence the security risk problem is common in APIs.

**HTTP METHODS**

* [**GET**](https://www.google.com/search?sca_esv=20f6847e24f004ff&rlz=1C1GCEA_enIN1177IN1177&q=GET&sa=X&ved=2ahUKEwiNuuu_ytKPAxWhg2MGHZR9DHIQxccNegQIIRAB&mstk=AUtExfDCVkwm7iYmfaDl2oLYK5U4dJwa85qS5BI3JmH7GecyXUTg3vFCnUQ3tmpvvMK214VPKfkmVQpaTHMvjAnrGd-TSWLyPh3QRRhE4P3OdlVZgz1_FPyZaXeQMI2GKS6upZ4&csui=3): Retrieves data from a specified resource.
* [**POST**](https://www.google.com/search?sca_esv=20f6847e24f004ff&rlz=1C1GCEA_enIN1177IN1177&q=POST&sa=X&ved=2ahUKEwiNuuu_ytKPAxWhg2MGHZR9DHIQxccNegQIIhAB&mstk=AUtExfDCVkwm7iYmfaDl2oLYK5U4dJwa85qS5BI3JmH7GecyXUTg3vFCnUQ3tmpvvMK214VPKfkmVQpaTHMvjAnrGd-TSWLyPh3QRRhE4P3OdlVZgz1_FPyZaXeQMI2GKS6upZ4&csui=3): Sends data to the server to create a new resource.
* [**PUT**](https://www.google.com/search?sca_esv=20f6847e24f004ff&rlz=1C1GCEA_enIN1177IN1177&q=PUT&sa=X&ved=2ahUKEwiNuuu_ytKPAxWhg2MGHZR9DHIQxccNegUIiQEQAQ&mstk=AUtExfDCVkwm7iYmfaDl2oLYK5U4dJwa85qS5BI3JmH7GecyXUTg3vFCnUQ3tmpvvMK214VPKfkmVQpaTHMvjAnrGd-TSWLyPh3QRRhE4P3OdlVZgz1_FPyZaXeQMI2GKS6upZ4&csui=3): Replaces an existing resource with the data provided in the request's body. If no resource exists, a new one may be created.
* [**PATCH**](https://www.google.com/search?sca_esv=20f6847e24f004ff&rlz=1C1GCEA_enIN1177IN1177&q=PATCH&sa=X&ved=2ahUKEwiNuuu_ytKPAxWhg2MGHZR9DHIQxccNegUIhgEQAQ&mstk=AUtExfDCVkwm7iYmfaDl2oLYK5U4dJwa85qS5BI3JmH7GecyXUTg3vFCnUQ3tmpvvMK214VPKfkmVQpaTHMvjAnrGd-TSWLyPh3QRRhE4P3OdlVZgz1_FPyZaXeQMI2GKS6upZ4&csui=3): Updates only a part of an existing resource with the data provided in the request.
* [**DELETE**](https://www.google.com/search?sca_esv=20f6847e24f004ff&rlz=1C1GCEA_enIN1177IN1177&q=DELETE&sa=X&ved=2ahUKEwiNuuu_ytKPAxWhg2MGHZR9DHIQxccNegUIjQEQAQ&mstk=AUtExfDCVkwm7iYmfaDl2oLYK5U4dJwa85qS5BI3JmH7GecyXUTg3vFCnUQ3tmpvvMK214VPKfkmVQpaTHMvjAnrGd-TSWLyPh3QRRhE4P3OdlVZgz1_FPyZaXeQMI2GKS6upZ4&csui=3): Removes the specified resource from the server.

Status Codes

* *1xx informational response* – the request was received, continuing process
* *2xx successful* – the request was successfully received, understood, and accepted
* *3xx redirection* – further action needs to be taken in order to complete the request
* *4xx client error* – the request contains bad syntax or cannot be fulfilled
* *5xx server error* – the server failed to fulfil an apparently valid request

200  (Success/OK)

301 (Permanent Redirect)

302 (Temporary Redirect)

304 (Not Modified)

400 (Bad Request)

401 (Unauthorized Error)

403 (Forbidden)

404 (Not Found)

500 (Internal Server Error)

501 (Not Implemented)

**Payload**

In the context of an API (Application Programming Interface), the payload refers to the actual data being sent or received within the body of an HTTP request or response. It is the core information that is being exchanged between the client (e.g., a web application) and the server.

Here's a breakdown:

* **Request Payload:**

When a client sends data to a server through an API request (e.g., creating a new user, updating a record), the information being sent in the request's body is the request payload. For example, when registering a new user, the payload might contain the username, email, and password.

* **Response Payload:**

When a server responds to an API request, the data returned in the response's body is the response payload. This could be the requested data (e.g., a list of products), a confirmation of a successful operation, or an error message.

Key characteristics of a payload:

* **Data Content:**

It contains the essential data relevant to the API operation.

* **Location:**

It resides within the body of an HTTP request or response, distinct from headers and other metadata.

* **Format:**

Payloads are typically structured in common data formats like JSON (JavaScript Object Notation) or XML (Extensible Markup Language) to ensure easy parsing and processing by both client and server.

* **Purpose:**

It enables the actual communication of information and instructions between applications, making the API functional.

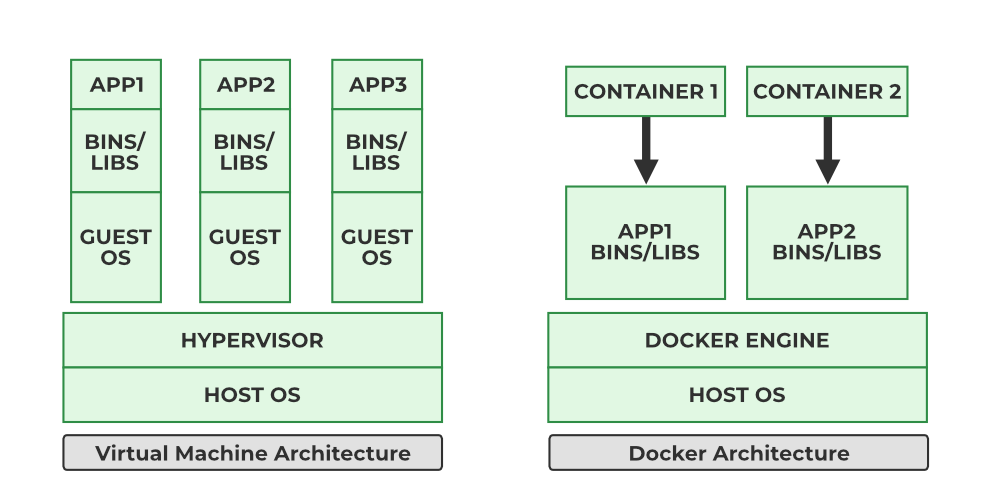
|  |  |  |
| --- | --- | --- |
| **Feature / Criteria** | **JSON (JavaScript Object Notation)** | **XML (eXtensible Markup Language)** |
| **Data Format** | Text-based, lightweight data-interchange format | Text-based, markup language |
| **Readability** | More human-readable and compact | More verbose and harder to read |
| **Syntax** | Uses key-value pairs, arrays, and objects | Uses custom tags with opening and closing elements |
| **Data Types Support** | Supports strings, numbers, arrays, booleans, null, objects | All values are treated as strings unless parsed |
| **Verbosity** | Less verbose (concise) | More verbose (lots of tags) |
| **Schema Support** | Optional, JSON Schema (less mature) | Strong schema support (XSD) |
| **Parsing Speed** | Faster to parse due to simpler structure | Slower parsing due to complex structure |
| **Namespaces** | Not supported | Fully supports namespaces |
| **Comments** | Not supported in standard JSON | Supports comments |
| **Transformation** | Limited transformation tools | Powerful transformation via XSLT |
| **Used In** | Web APIs, config files, AJAX, NoSQL databases | Web services (SOAP), config files, document storage |
| **Support in Browsers** | Native support in JavaScript | Requires parser like DOM or SAX |
| **Data Interchange** | Optimized for data interchange | Optimized for document structure and metadata |

# Day 12- Docker

**What is Docker?**

Docker is an OS‑level virtualization (or containerization) platform, which allows applications to share the host OS kernel instead of running a separate guest OS like in traditional virtualization. This design makes Docker containers lightweight, fast, and portable, while keeping themisolated from one another.

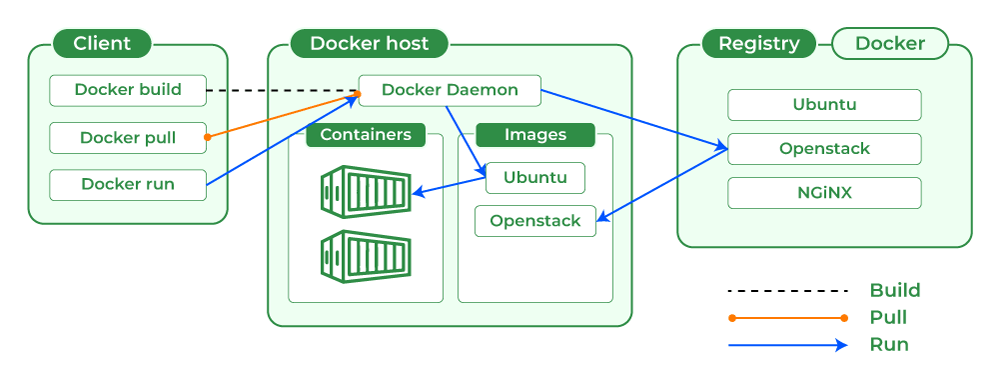
* Written in the **Go** programming language.
* Supports Windows, macOS, and Linux installations (Docker Engine runs natively on Linux).
* Solves the *“works on my machine”* problem by ensuring code runs identically across environments.
* Unlike VMware (hardware‑level virtualization), Docker operates at the OS level.



**Components of Docker**

The following are the some of the key components of Docker:

* **Docker Engine:**[Docker Engine](https://www.geeksforgeeks.org/devops/what-is-docker-engine/) has a core part docker daemon , that handles the creation and management of containers.
* **Docker Image:**[Docker Image](https://www.geeksforgeeks.org/devops/what-is-docker-image/)is a read-only template that is used for creating containers, containing the application code and dependencies.
* **Docker Hub:**It is a cloud based repository that is used for finding and sharing the container images.
* **Dockerfile:**It is a file that describes the steps to create an image quickly.
* **Docker Registry**: It is a storage distribution system for docker images, where you can store the images in both public and private modes.



Advantages and Disadvantages of docker

|  |  |
| --- | --- |
| **Docker Pros** | **Docker Cons** |
| **Cross-platform consistency:** Compatibility across a range of systems and environments makes developers’ jobs easier. | **Outdated documentation:** Docker’s extensive documentation doesn’t always keep pace with platform updates. |
| **Serverless storage:** Docker containers are cloud-based and don’t require tons of active memory to run reliably. | **Steep learning curve:** Developers transitioning from other infrastructure might find Docker easy to begin but hard to master. |
| **High-speed deployment:** Eliminating redundant installations and configurations makes deployment fast and easy. | **Security issues:** The lack of segmentation means that multiple containers can be vulnerable to host system attacks. |
| **Flexibility and scalability:** Developers can use any programming language and scale container resources up and down as needed. | **Limited orchestration:** It can be difficult to manage multiple containers at once without orchestration capabilities. |

**What is Kubernetes?**

Kubernetes is an open-source platform designed to automate deploying, scaling, and operating application containers. It simplifies the developer’s task of managing containerized applications. It solves many problems teams face during managment of containerized applications. Some of these challenges are as below:

**Challenges of Containerized Applications**

Managing containerized applications, whether using Docker containers or some other containers runtime, comes with its own set of challenges, such as:

* **Scalability**: As the number of containers grows, it becomes challenging to scale them effectively.
* **Complexity**: Managing numerous containers, each with its own role in a larger application, adds complexity.
* **Management**: Keeping track of and maintaining these containers, ensuring they are updated and running smoothly, requires significant effort.

**Kubernetes as a Solution**

Now that we have gone through the challenges of using containerized applications, let’s see how Kubernetes handle these challenges.

* Kubernetes steps in as a powerful platform to manage these complexities. It’s an open-source system designed for automating deployment, scaling, and operation of application containers across clusters of hosts.
* It simplifies container management, allowing applications to run efficiently and consistently.
* Kubernetes orchestrates a container’s lifecycle; it decides how and where the containers run, and manages their lifecycle based on the organization’s policies.

Learn more about the role of Kubernetes in modern development in our article [Why Kubernetes for Developers is the Next Big Thing](https://www.qovery.com/blog/why-kubernetes-for-developers-is-the-next-big-thing/).

**Benefits of Kubernetes**

Some important benefits of Kubernetes are listed below:

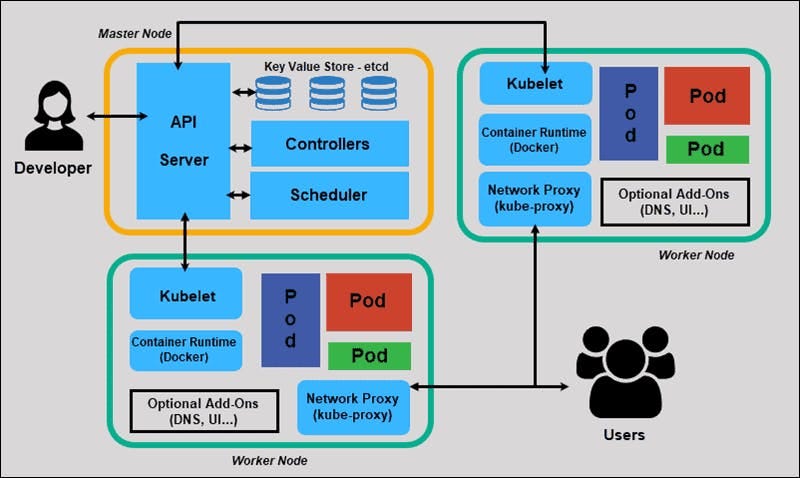
* **Efficiency**: Kubernetes optimizes the use of hardware resources, saving costs.
* **Reliability**: It ensures that the application services are available to users without downtime.
* **Flexibility and portability**: Kubernetes supports diverse workloads, including stateless, stateful, and data-processing workloads. Its flexibility allows it to run on various platforms, from physical machines to cloud infrastructure.
* **Security and resource management**: It provides robust security features and efficient management of resources, ensuring that the infrastructure is secure and the resources are used optimally.
* **Support for Docker and other container technologies**: Kubernetes works well with Docker and other container technologies, offering a wide range of options for containerization.
* **Open source community**: Being open source, Kubernetes benefits from a large community of developers and users who contribute to its continuous improvement.

**Key Concepts**

**Core Kubernetes components**

Below are some core components of Kubernetes.

1. **Pods:** The smallest deployable units created and managed by Kubernetes. A Pod represents a single instance of a running process in your cluster and can contain one or more containers.
2. **Nodes:** These are worker machines in Kubernetes, which can be either a physical or virtual machine, depending on the cluster. Each node runs Pods and is managed by the master.
3. **Deployments:** They describe the desired state of your application, like which images to use and the number of Pod replicas. Deployments update your application to the desired state at a controlled rate.
4. **Services:** They are an abstract way to expose an application running on a set of Pods as a network service. This decouples workloads from specific Pods, providing a consistent way to access the application.
5. **Ingress:** This manages external access to the services in a cluster, typically HTTP. Ingress can provide load balancing, SSL termination, and name-based virtual hosting.
6. **Namespaces:** Namespaces help split a Kubernetes cluster into sub-clusters, making it possible to divide resources between different projects or teams.
7. **Labels and Selectors:** They are powerful tools that allow you to organize and select subsets of objects, like Pods, based on key-value pairs for more precise resource management.



# Day 13 – Cloud

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. You typically pay only for cloud services you use, helping you lower your operating costs, run your infrastructure more efficiently, and scale as your business needs change.

**Key takeaways**

Types of cloud computing

* Discover the top benefits of cloud computing on cost, performance, security, and more.
* Understand the differences between public, private, and hybrid cloud deployment models.
* Learn about the four broad categories of cloud services: IaaS, PaaS, serverless, and SaaS.
* Explore cloud computing services from Microsoft—and find out which cloud computing option best suits your needs.

**Benefits**

Cloud computing is a big shift from the traditional way businesses think about IT resources. Here are seven common reasons organizations are turning to cloud computing services:

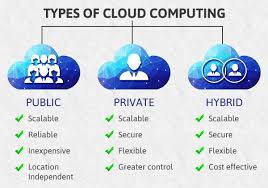
* **Infrastructure:**Cloud computing depends on remote network servers hosted on the Internet to store, manage, and process data.
* **On-Demand Access:** Users can access cloud services and resources on demand, scaling up or down without having to invest in physical hardware.
* **Types of Services:**Cloud computing offers various benefits such as cost saving, scalability, reliability, and accessibility. It reduces capital expenditures, and improves efficiency.



**Types of Cloud Computing Services**

The following are the types of Cloud Computing:

1. [Infrastructure as a Service (IaaS)](https://www.geeksforgeeks.org/cloud-computing/infrastructure-as-a-service-iaas/)
2. [Platform as a Service (PaaS)](https://www.geeksforgeeks.org/cloud-computing/platform-as-a-service-paas-and-its-types/)
3. [Software as a Service (SaaS)](https://www.geeksforgeeks.org/software-engineering/software-as-a-service-saas/)
4. [Function as as Service (FaaS)](https://www.geeksforgeeks.org/cloud-computing/google-cloud-platform-understanding-functions-as-a-service-faas/)



* 1. Infrastructure as a Service
* Infrastructure as a Service (IaaS) is a [cloud computing](https://www.geeksforgeeks.org/cloud-computing/cloud-computing/) service model that gives virtualized computing resources over the web, with IaaS, associations can get to and manage versatile infrastructure assets like [virtual machines](https://www.geeksforgeeks.org/operating-systems/types-of-virtual-machines/), storage, and networking administration parts without the need to put resources into or keep up with actual equipment.
* IaaS allows business to outsource their whole IT infrastructure to a cloud service provider, empowering them to arrange, deploy, and manage computing resources on-demand, this adaptability allows organizations to increase their infrastructure or down in view of fluctuating interest, pay just for the resources they consume, and keep away from the expenses and intricacies related with customary on-premises infrastructure.

1. Platform as a Service

Platform as a Service, also known as PaaS, is a [type of cloud computing service model](https://cloud.google.com/discover/types-of-cloud-computing) that offers a flexible, scalable cloud platform to develop, deploy, run, and manage apps. PaaS provides everything developers need for application development without the headaches of updating the operating system and development tools or maintaining hardware. Instead, the entire PaaS environment—or platform—is delivered by a third-party service provider via the cloud.

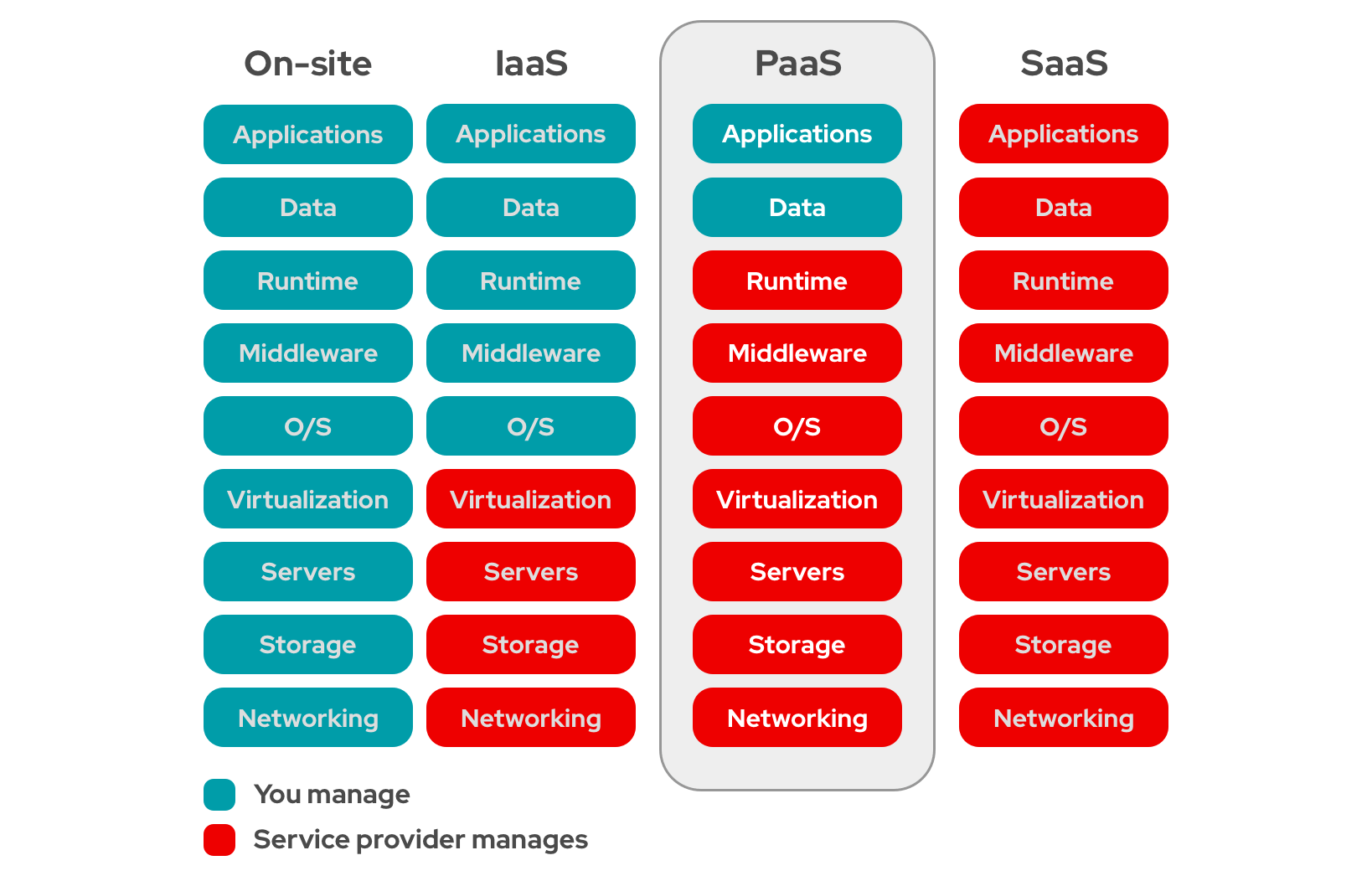
PaaS helps businesses avoid the hassle and cost of installing hardware or software to develop or host new custom applications. Development teams simply purchase pay-as-you-go access to everything they need to build custom apps, including infrastructure, development tools, operating systems, and more.

The result is simpler, faster, and secure app development that gives developers the freedom to focus on their application code.

Unlike IaaS or SaaS service models, PaaS solutions are specific to application and software development and typically include:

* **Cloud infrastructure**: Data centers, storage, network equipment, and servers
* **Middleware software**: Operating systems, frameworks, development kits (SDK), libraries, and more
* **User interface**: A graphical user interface (GUI), a command line interface (CLI), an API interface, and in some cases, all three

Platform as a Service is typically delivered as a secure online platform that developers can access over the internet, allowing them to work on projects from anywhere and collaborate freely with other members of their team. Applications are built directly on the PaaS system and can be immediately deployed once they are completed.



1. Saas

Software as a Service (SaaS) is a business model in which customers pay to access and use cloud-hosted software over the Internet rather than purchasing it outright. This differs from traditional software that you need to purchase and install yourself. Instead, SaaS provides access to apps through monthly or annual subscriptions, with common features such as multi-user accounts and pricing tiers.

Examples of popular SaaS platforms include Salesforce and Slack, which offer tools like CRM, communication, and collaboration through a simple pay-as-you-go model.

As such, SaaS makes it easy for users to connect to powerful applications from any internet-enabled device and pay for the level of service they need.

AWS

Amazon Web Services (AWS) was started in 2006 to help companies avoid the high cost and effort of buying and managing their servers. Before AWS, businesses had to set up physical computers and storage to run websites or apps, which took time and money. AWS came into the market to solve this problem by offering these resources over the internet.

Amazon Web Services (AWS) is a cloud platform offered by Amazon that lets people and companies use IT services like storage, servers, and databases through the internet. Instead of buying and maintaining physical computers, you can "rent" these services online and only pay for what you use.

AWS offers over 200 services including:

* **EC2:** Virtual servers for running applications.
* **S3:** Object storage for files and media.
* **RDS:** Managed relational databases.
* **Lambda:**Run code without managing servers.

From startups to large enterprises like **Netflix**, **Airbnb**, and **NASA**, AWS is widely adopted for its flexibility, scalability, and security.



Advantages



**1. Easy to Use**

AWS is designed for ease of use, making cloud management accessible even to those with minimal technical experience.

* The AWS Management Console offers a simple, web-based interface to manage cloud resources.
* Users can quickly set up servers, storage, and applications with just a few clicks.
* AWS provides detailed documentation, video tutorials, and community support, making it easy to learn.

Even businesses without a dedicated IT team can start using AWS without major technical challenges.

**2. Cost-Effective**

One of the biggest advantages of AWS is its pay-as-you-go pricing model, which eliminates the need for large upfront investments.

* Businesses only pay for the resources they use, reducing unnecessary costs.
* AWS offers Reserved Instances and Spot Instances, which provide discounts for long-term and flexible usage.
* Startups and small businesses can start with a low budget and scale up as needed.

By optimizing cloud usage, businesses can save a significant amount on IT infrastructure costs.

**3. Effortless Migration**

Migrating existing applications and data to AWS is easy because it supports multiple technologies.

* AWS Migration Hub helps businesses plan, track, and execute migrations smoothly.
* Amazon S3 (object storage) and Amazon EBS (block storage) provide scalable, secure storage solutions.
* AWS supports a wide range of operating systems, databases, and programming languages, so businesses don’t need to change their existing applications much.

This means businesses can move to AWS with minimal downtime and disruptions.

**4. Data Privacy**

AWS prioritizes security and ensures that customer data is protected at all levels.

* Data Encryption: AWS encrypts data both at rest (stored data) and in transit (data being transferred).
* Identity and Access Management (IAM): Allows businesses to control who has access to specific data.
* Automated Security Monitoring: Tools like AWS Security Hub and AWS Config help detect vulnerabilities.

AWS also complies with industry regulations (like GDPR, HIPAA, and ISO 27001), making it a secure option for businesses handling sensitive data.

**5. Reliability**

AWS is built for reliability, ensuring that applications and data remain available without major interruptions.

* It boasts over 38 global data centers strategically located to provide redundancy and disaster recovery capabilities.
* Amazon EC2 Auto Scaling automatically adjusts resources based on demand, preventing performance issues.
* Multi-region deployment ensures that even if one data center fails, services can continue from another location.

This means AWS can handle traffic spikes, prevent downtime, and maintain smooth business operations.

Disadvantages



**1. Confusing Bills**

AWS’s pricing structure can be difficult to understand, especially for businesses unfamiliar with cloud computing.

* Each AWS service has a different pricing model, which can lead to unexpected costs.
* Businesses need to actively monitor usage to avoid high bills.
* Many companies hire AWS experts or use third-party billing tools to manage costs.

Without proper cost management, businesses might end up paying more than expected.

**2. Limited Resources**

AWS allocates resources based on geographical data center locations, which can cause issues.

* Some AWS services are only available in specific regions, limiting options for businesses in certain locations.
* Businesses may need to request additional resources, which can cause delays.

This can be a problem for companies that need high-performance computing in a specific location but face service restrictions.

**3. Shortage of Experts**

AWS is a powerful but complex platform, requiring skilled professionals to manage it effectively.

* The demand for AWS-certified experts is high, making it difficult and expensive to hire the right talent.
* Businesses without an in-house AWS specialist may struggle to optimize cloud performance and security.
* Companies may need to invest in AWS training for existing employees, which can take time and resources.

Without proper expertise, businesses may not fully leverage AWS’s potential.

**4. Cloud Computing Concerns**

Although AWS is highly reliable, cloud computing is not 100% risk-free.

* Power outages, internet failures, or hardware issues can cause service disruptions.
* Businesses relying entirely on AWS must have a backup or disaster recovery plan.
* Cybersecurity risks, such as data leaks and hacking, require companies to continuously monitor and update security policies.

While AWS provides strong security tools, businesses must also take responsibility for protecting their own data.

S3 bucket

AWS offers a wide range of storage services that can be configured depending on your project requirements and use cases. AWS comes up with different types of storage services for maintaining highly confidential data, frequently accessed data, and often accessed storage data. You can choose from various storage service types such as O*bject Storage as a Service(Amazon S3), File Storage as a Service (Amazon EFS), Block Storage as a Service (Amazon EBS), backups,* and *data migration* options.

Amazon S3 is a Simple Storage Service in[AWS](https://www.geeksforgeeks.org/devops/aws-tutorial/)that stores files of different types like Photos, Audio, and Videos as Objects providing more scalability and security to. It allows the users to store and retrieve any amount of data at any point in time from anywhere on the web. It facilitates features such as extremely high availability, security, and simple connection to [other AWS Services.](https://www.geeksforgeeks.org/cloud-computing/top-aws-services/)

**What is Amazon S3 Used for?**

Amazon S3 is used for various purposes in the Cloud because of its robust features with scaling and Securing of data. It helps people with all kinds of use cases from fields such as Mobile/Web applications,[Big data,](https://www.geeksforgeeks.org/data-engineering/what-is-big-data/) [Machine Learning](https://www.geeksforgeeks.org/machine-learning/ml-machine-learning/) and many more. The following are a few Wide Usage of Amazon S3 service.

* **Data Storage:**Amazon s3 acts as the best option for scaling both small and large storage applications. It helps in storing and retrieving the data-intensitive applications as per needs in ideal time.
* **Backup and Recovery:**Many Organizations are using Amazon S3 to backup their critical data and maintain the data durability and availability for recovery needs.
* **Hosting Static Websites:**Amazon S3 facilitates in storing HTML, CSS and other web content from Users/developers allowing them for hosting Static Websites benefiting with low-latency access and cost-effectiveness. To know more detailing refer this Article - [How to host static websites using Amazon S3](https://www.geeksforgeeks.org/devops/how-to-host-static-website-using-aws-s3/)
* **Data Archiving:**[Amazon S3 Glacie](https://www.geeksforgeeks.org/cloud-computing/what-is-amazon-glacier/)r service integration helps as a cost-effective solution for long-term data storing which are less frequently accessed applications.
* **Big Data Analytics:**Amazon S3 is often considered as data lake because of its capacity to store large amounts of both structured and unstructured data offering seamless integration with other AWS Analytics and AWS Machine Learning Services.

**What is an Amazon S3 bucket?**

Amazon S3 bucket is a fundamental Storage Container feature in AWS S3 Service. It provides a secure and scalable repository for storing of Objects such as Text data, Images, Audio and Video files over AWS Cloud. Each S3 bucket name should be named globally unique and should be configured with ACL (Access Control List).

**Advantages**

**Pros of Amazon S3**

Here are some of the **main benefits of Amazon S3**:

**Ease of Use**

Amazon S3 cloud storage is really easy to use, great if you don’t have much experience using cloud services. It’s specifically designed for fast, secure access whenever and wherever you want. There’s also a wealth of documentation, videos, and information provided to help you get started using the service, and an intuitive interface that makes navigating the service a breeze.

**Incredibly Diverse Array of Tools**

Amazon S3 is just one part of the wider Amazon Web Services (AWS). This infrastructure as a service (IaaS) platform is a one-stop shop for all your IT requirements. AWS provides over 70 services including analytics, networking, mobile, database, and more. Tools and features you can take advantage of include versioning, which allows you to roll back to previous versions of a file or recover deleted objects, and managed IT services, designed to make running your business easier.

**Unlimited Server Capacity**

One of the main benefits of Amazon S3 is that it offers unlimited server capacity, so you can store your data without worrying about hard drive failure on macOS or service disruption. With unlimited bandwidth, secure email hosting, and more, Amazon S3 and AWS are reliable and scalable as your business expands or grows.

**Reliable Encryption and Security**

Storing stuff in the cloud is all well and good, but what if you have sensitive data that you need to ensure stays confidential? With Amazon S3, you don’t need to worry about your site being hacked or files stolen or deleted. AWS has 38 data centers worldwide, and [Amazon S3 encryption](https://cloudmounter.net/amazon-s3-encryption-software.html) uses the Transport Layer Security (TLS) protocol to encrypt data in motion (i.e. when you’re uploading or downloading files to or from the service).

**Cons of Amazon S3**

Every cloud storage service has its cons, and Amazon S3 is no exception. Here are some of the main **disadvantages of Amazon S3**:

**Billing Can be Confusing**

For all its benefits, Amazon S3 (and AWS’) billing can be confusing and complicated, particularly for small business owners who aren’t especially tech-savvy. By working with a reseller, you’ll eliminate this problem, so invoicing and billing will be clearer and easier to understand.

**Amazon’s S3 Limits**

When signing up for Amazon S3 and AWS, you choose which region to store your data in, and it’s usually recommended to choose your own region closest to your location. Unfortunately, one of the main disadvantages of Amazon S3 is that AWS limits resources by location, so some areas may have fewer resources than others. It can also be challenging to find out how many resources you’ve used, although you can request a resource increase if needed. In addition to this, new users are limited as to how many resources they can use and how much money they can spend until their membership is more established. This is to prevent hackers from using AWS’ resources for malicious attacks.

EC2 service

Amazon Elastic Compute Cloud (Amazon EC2) provides on-demand, scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 reduces hardware costs so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. You can add capacity (scale up) to handle compute-heavy tasks, such as monthly or yearly processes, or spikes in website traffic. When usage decreases, you can reduce capacity (scale down) again.

An EC2 instance is a virtual server in the AWS Cloud. When you launch an EC2 instance, the instance type that you specify determines the hardware available to your instance. Each instance type offers a different balance of compute, memory, network, and storage resources.

Amazon EC2 provides the following high-level features:

**Instances**

Virtual servers.

**Amazon Machine Images (AMIs)**

Preconfigured templates for your instances that package the components you need for your server (including the operating system and additional software).

**Instance types**

Various configurations of CPU, memory, storage, networking capacity, and graphics hardware for your instances.

**Amazon EBS volumes**

Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS).

**Instance store volumes**

Storage volumes for temporary data that is deleted when you stop, hibernate, or terminate your instance.

**Key pairs**

Secure login information for your instances. AWS stores the public key and you store the private key in a secure place.

**Security groups**

A virtual firewall that allows you to specify the protocols, ports, and source IP ranges that can reach your instances, and the destination IP ranges to which your instances can connect.

Amazon EC2 supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS). For more information about PCI DSS, including how to request a copy of the AWS PCI Compliance Package, see [PCI DSS Level 1](https://aws.amazon.com/compliance/pci-dss-level-1-faqs/)

**AWS Elastic Beanstalk**

AWS Elastic Beanstalk is a Platform as a Service (PaaS) offering by Amazon Web Services that simplifies the deployment and management of web applications and services. It provides a layer of abstraction over core AWS infrastructure, allowing developers to focus on their application code rather than managing underlying resources.

Key Services and Functionalities:

* **Application Deployment:**

Users upload their application code (e.g., Java, Python, Node.js, PHP, Ruby, .NET, Go, Docker), and Elastic Beanstalk automatically handles the deployment process.

* **Infrastructure Provisioning:**

It provisions and configures the necessary AWS resources, including Amazon EC2 instances, Elastic Load Balancers (ELBs), Auto Scaling groups, and security groups.

* **Load Balancing:**

Elastic Beanstalk integrates with ELBs to distribute incoming traffic across multiple EC2 instances, ensuring high availability and performance.

* **Auto Scaling:**

It automatically scales the application's capacity up or down based on predefined metrics and traffic patterns, ensuring optimal resource utilization.

* **Application Health Monitoring:**

Elastic Beanstalk monitors the health of the application and its underlying resources, providing insights and alerts on performance and availability.

* **Platform Management:**

It offers managed platforms for various programming languages and frameworks, handling updates and patches for the underlying operating system and application server.

* **Environment Configuration:**

Users can customize environment variables, database connections, and other settings to tailor the application's environment.

* **Integration with other AWS services:**

Elastic Beanstalk integrates seamlessly with other AWS services such as Amazon S3 for storage, Amazon RDS for databases, and AWS CloudWatch for monitoring.

Benefits:

* **Reduced Operational Overhead:**

Automates infrastructure management tasks, freeing up developers to focus on application development.

* **Fast Deployment:**

Enables quick and easy deployment of applications without manual configuration of servers and resources.

* **Scalability and High Availability:**

Provides built-in features for scaling and load balancing to handle varying traffic loads and ensure application availability.

* **Cost-Effective:**

Users only pay for the underlying AWS resources consumed, with no additional charge for Elastic Beanstalk itself.

* **Flexibility and Control:**

While managing infrastructure, it still allows users to access and customize the underlying AWS resources if needed.

Zones in Cloud computing

1. Availability Zone (AZ)
2. Fault Zone
3. Failure Domain
4. Update Domain
5. Placement Zone
6. Local Zone
7. Edge Zone
8. Proximity Zone
9. Security Zone
10. Geographic Zone

Region

Cloud regions are distinct geographical areas where cloud providers operate data centers. Availability zones (AZs) are logical data centers within a cloud region. Cloud regions usually have multiple isolated AZs, with each AZ comprising one or more data centers located in close proximity.

A cloud testing environment is a virtual space where software applications are tested utilizing cloud computing services and resources. This environment is designed to leverage the flexibility, scalability, and cost-optimization capabilities of the cloud, enabling teams to conduct various types of testing without requiring physical hardware.

Key aspects of testing environments in cloud computing:

* **Virtualization and On-Demand Resources:**

Cloud environments provide virtual machines, storage, and network resources on demand, allowing for rapid provisioning and de-provisioning of testing infrastructure.

* **Scalability and Elasticity:**

Testers can easily scale up or down resources based on testing needs, which is particularly beneficial for performance, load, and stress testing.

* **Accessibility and Collaboration:**

Cloud-based environments can be accessed remotely, facilitating collaboration among geographically dispersed testing teams.

* **Cost-Effectiveness:**

The pay-as-you-go model of cloud computing reduces the need for upfront investment in physical infrastructure and allows for cost optimization by paying only for resources consumed during testing.

* **Diverse Testing Capabilities:**

Cloud environments support a wide range of testing types, including functional, performance, security, compatibility, and scalability testing.

* **Types of Cloud Testing Environments:**
  + **Public Cloud:** Shared resources among multiple users, offering high scalability and cost-efficiency.
  + **Private Cloud:** Dedicated resources for a single organization, providing enhanced control and security.
  + **Hybrid Cloud:** Combines public and private cloud elements, balancing cost and control.
  + **Community Cloud:** Shared infrastructure for specific communities or organizations with common requirements.
* **Integration with CI/CD and DevOps:**

Cloud testing environments seamlessly integrate with Continuous Integration/Continuous Delivery (CI/CD) pipelines and DevOps practices, enabling automated and continuous testing throughout the software development lifecycle.

Types of Testing  
Cloud computing environments necessitate various types of testing to ensure application quality, performance, and security. These types of testing can be broadly categorized as follows:

1. Functional Testing:

* **System Testing:**

Validates that the entire application, including all its features and functionalities, operates according to the specified requirements within the cloud environment.

* **Interoperability Testing:**

Assesses the application's compatibility and seamless interaction with different cloud services, infrastructure components, and external systems.

* **Multi-tenancy Testing:**

Ensures that the application performs securely and efficiently when multiple users or tenants share the same cloud infrastructure, preventing data leakage or performance degradation between tenants.

* **Acceptance Testing:**

Verifies that the application meets the end-user's requirements and is ready for deployment in the cloud.

2. Non-Functional Testing:

* **Performance Testing:**

Evaluates the application's responsiveness, speed, and stability under various load conditions. This includes:

* + **Load Testing:** Measures application behavior under expected user loads.
  + **Stress Testing:** Determines the application's breaking point by pushing it beyond its normal operating limits.
* **Scalability Testing:**

Assesses the application's ability to handle increasing workloads and user demands by efficiently scaling resources up or down in the cloud.

* **Security Testing:**

Identifies vulnerabilities and weaknesses in the cloud application and infrastructure, ensuring data protection, access control, and compliance with security standards.

* **Availability Testing:**

Confirms that the application remains accessible and operational with minimal downtime, even during infrastructure changes or failures within the cloud environment.

* **Disaster Recovery (DR) Testing:**

Simulates disaster scenarios to verify the effectiveness of backup and recovery procedures, ensuring data integrity and business continuity in case of outages.

* **Compatibility Testing:**

Ensures the application functions correctly across different browsers, operating systems, devices, and cloud platforms.

3. Specific Cloud-Related Testing:

* **Browser Performance Testing:**

Focuses on the application's performance and responsiveness specifically within various web browsers.

* **SaaS Testing:**

Involves comprehensive functional and non-functional testing for applications delivered as Software as a Service (SaaS) in the cloud.

# Day 14 – selenium

Selenium is an open-source framework for automating web application testing across browsers and platforms. It offers tools like Selenium IDE for codeless testing and WebDriver for script-based automation in multiple programming languages (e.g., Java, Python, C#, etc.). Selenium supports cross-browser testing and works on Windows, Linux, and macOS. The Selenium Manager simplifies test setup by automatically handling browser driver management, removing the need for manual downloads and configurations.

**Key features and functionalities:**

* **Automated Driver Management:**

Selenium Manager automatically detects the installed browser (Chrome, Firefox, Edge, etc.) and downloads the compatible WebDriver executable (e.g., chromedriver, geckodriver) into a local cache.

* **Zero Configuration:**

With Selenium Manager, users no longer need to explicitly download and configure browser drivers or add them to the system PATH.

* **Browser Version Discovery:**

It identifies the version of the installed browser on the system to ensure the correct driver version is used.

* **Driver Version Resolution:**

Based on the discovered browser version, it resolves the appropriate driver version by consulting online metadata provided by browser vendors.

* **Driver Download and Cache:**

It downloads the resolved driver, uncompresses it, and stores it in a local cache (typically ~/.cache/selenium) for future use, improving efficiency.

* **Cross-Platform Compatibility:**

Implemented in Rust, Selenium Manager is a command-line interface (CLI) tool that supports various operating systems.

* **Integration with Selenium Bindings:**

It is integrated by default into Selenium 4.6.0 and later versions, making it accessible through standard Selenium code without requiring additional dependencies.

**Pros of the Selenium Automation Testing Tool**

**1. Free and Open Source**

* Selenium's popularity stems largely from its free and open-source nature.
* It allows companies of every type to use it for automated testing, and the community is constantly adding new functionality.
* This highlights its potential for automated testing and makes it available to a wide range of businesses, regardless of size or sector.
* Selenium has made a name for itself as the platform of choice for businesses looking for affordable and effective automated testing solutions because it is free and open source.

**2. Supports a Wide Number of Systems, Languages, and Browsers**

* no better tool than Selenium can automate testing across most major operating systems and browsers.
* The robust automation tool Selenium demonstrates remarkable adaptability by interacting with a wide variety of systems, browsers, and computer languages with ease.
* This feature allows QA specialists, developers, and testers to do thorough tests on several platforms with greater freedom.
* As it is cross-compatible, switching between separate tools for each platform is avoided, which saves a significant amount of time.
* Automating testing processes for popular operating systems and browsers is a simple way to increase productivity and accelerate the software development lifecycle. Within a quickly changing technological environment, selenium's flexibility plays a major role in upholding high levels of dependability and quality.

**3. Integrated with CI/CD**

* Another significant advantage is that Selenium is completely integrated with CI/CD—continuous integration /continuous deployment—platforms like Jenkins, Azure DevOps, Bamboo, Travis CI, Circle CI, GitLab, and many others.
* The exceptional interoperability of Selenium with numerous CI/CD platforms is one of the main advantages of using it for automated testing.
* Selenium's smooth interface with these systems facilitates automated deployments and continuous monitoring, which speeds up the testing process and increases the development team's productivity.

**4. Run Tests Concurrently**

* Selenium Grid allows you to run tests concurrently and manage execution from a single location. Web tests are known to take longer to run over time, and parallel execution provides significant benefits as your test suite grows.
* Selenium Grid, a powerful tool in the realm of automated testing, empowers users to execute test cases simultaneously while efficiently controlling the process from a centralized hub.
* In the landscape of web testing, where the duration of tests tends to increase as test suites evolve, utilizing parallel execution brings forth notable advantages. By harnessing Selenium Grid's capabilities to run tests concurrently, users can optimize test suite performance as it scales, enhancing productivity and reducing overall testing time.

**Cons of the Selenium Automation Testing Tool**

**1. Supports Web Applications Only**

* Selenium is only useful for web test automation.
* Although Selenium can only automate interactions within web browsers, it is unable to automate processes outside of the browser, such as desktop applications.

**2. Not Having an Image Comparison**

* Selenium does not have built-in image comparison functionality, so it is important to ensure that all images you display in your application are present and displayed correctly.
* A third-party library is required to achieve this.
* This restriction emphasizes how crucial it is to carefully verify that every image in the user interface of your application exists and is accurate. It becomes imperative to incorporate a third-party image comparison library into your testing suite to close this gap.

**3. Selenium cannot Create Reports**

* Selenium does not have a basic functionality called reports. To communicate test automation results to management, contractors, customers, and stakeholders, testers, and developers must generate all reports.
* Although Selenium lacks native reporting features, it offers integration support for several reporting frameworks and libraries.
* Test automation engineers can use Selenium WebDriver to create thorough test reports by utilizing third-party reporting technologies like TestNG, ExtentReports, Allure, or custom reporting solutions.

**4. No Technical Support will be Provided**

* Renowned for its powerful automated testing features, Selenium has a large community of seasoned specialists willing to share their vast experience and knowledge. Nevertheless, even with a large number of specialists, users who are just learning about Selenium's capabilities may find it difficult to navigate among the array of solutions accessible.
* One major challenge that many new users have is the lack of readily available Selenium-focused businesses or service providers.
* Lack of professional support can be quite difficult because it can be difficult to troubleshoot problems or seek advice when there aren't easily available resources.
* Due to this, learning Selenium and realizing its full potential for effective testing procedures may seem like a difficult and lonely road for individuals who choose to take it.

**Selenium WebDriver** is an open-source automation framework that enables programmatic control of web browsers. It functions as an object-oriented API that natively drives a browser, mimicking user interactions like clicking buttons, typing text, and navigating pages.

**Selenium architecture**



The architecture of Selenium WebDriver is structured to provide flexibility, speed, and scalability for test automation. It follows a client-server model, with several components that work together to send commands to the browser and get responses from the same.

Compared to Selenium WebDriver version 3, Selenium WebDriver 4 has seen significant improvements have been made in terms of functionality, performance, and ease of use. Selenium 4 introduces better support for modern browsers, improved W3C compliance, and new features like enhanced grid capabilities, which make it even more powerful for automating web applications.

