Day 13 – python Training

Today's session was focused on understanding the core concepts of **Object-Oriented Programming (OOPs)** in Python. It was an important topic that helped us shift from procedural programming to thinking in terms of objects and classes.

✅ **Key Concepts Covered:**

- **Introduction to OOPs:**
  Learned why OOP is useful for organizing complex programs and managing code more efficiently through real-world modeling.
- **Class and Object:**
  Understood the basic building blocks of OOP — a **class** is a blueprint, and an **object** is an instance of that blueprint.
- **Constructor (__init__ method):**
  Learned how constructors are used to initialize object properties when an object is created.
- **Self Parameter:**
  Understood the use of self to refer to the instance of the class and access its attributes and methods.
- **Encapsulation:**
  Learned how to hide internal object details using private and protected members, and why it's important for data protection.
- **Inheritance:**
  Explored how one class can inherit features from another to promote code reuse and structure.
- **Polymorphism:**
  Understood how methods can behave differently based on the object calling them (method overriding and operator overloading).
- **Abstraction:**
  Discussed how abstraction helps in hiding unnecessary implementation details and showing only essential features.
- **Access Modifiers:**
  Learned about public, protected, and private members in Python and how naming conventions are used to implement them.
- **Real-life Examples:**
  Saw how OOP is used in real applications like creating user profiles, managing systems like banking, student records, etc.