**Classes and Solid Principles (Tracing the code)**

1. **EmployeeDetails Class** (Single Responsibility Principle - SRP)
   * **Purpose**: This class have one responsibility ( holding basic information about an employee ).
   * **Attributes**:
     + string name: The name of the employee.
     + int age: The age of the employee.
     + string position: The position or job title of the employee.
   * **Method**:
     + printDetails(): Prints the employee's details to the console.
2. **EmployeeManagement Class** (Single Responsibility Principle - SRP)
   * **Purpose**: Manages actions related to employees, such as hiring and firing.
   * **Methods**:
     + hireEmployee(EmployeeDetails& Emp): Prints a message indicating the hiring of the specified employee.
     + fireEmployee(EmployeeDetails& Emp): Prints a message indicating the firing of the specified employee.
3. **Manager Class** (Inheritance and SRP)
   * **Purpose**: Extends EmployeeDetails to represent a managerial role, including team management responsibilities.
   * **Attributes**:
     + int teamSize: Represents the number of people in the manager's team.
   * **Methods**:
     + manageTeam(): Prints a message about managing a team.
     + addTeamMember(): Prints a message about adding a team member.
     + removeTeamMember(): Prints a message about removing a team member.
     + scheduleMeeting(): Prints a message about scheduling a meeting.
     + reviewPerformance(): Prints a message about reviewing team performance.
4. **Report Class** (Open/Closed Principle - OCP)
   * **Purpose**: An abstract base class that defines a contract for generating different types of reports.
   * **Method**:
     + generateReport(): A pure virtual function that derived classes must implement.
5. **PDFReport Class** (Open/Closed Principle - OCP)
   * **Purpose**: A concrete implementation of the Report class for generating PDF reports.
   * **Method**:
     + generateReport(): Implements the generation of a PDF report.
6. **WordReport Class** (Liskov Substitution Principle - LSP)
   * **Purpose**: A concrete implementation of the Report class for generating Word reports.
   * **Method**:
     + generateReport(): Implements the generation of a Word report.
   * **LSP**: This class can be used wherever a Report is expected, substituting the PDFReport class without altering the behavior of the code.
7. **PayrollSystem Class** (Single Responsibility Principle - SRP)
   * **Purpose**: Manages payroll operations.
   * **Methods**:
     + calculatePay(EmployeeDetails& Emp): Prints a message indicating pay calculation for the specified employee.
     + processPayroll(): Prints a message about processing payroll.
     + generatePayslip(): Prints a message about generating a payslip.
     + sendPayslip(): Prints a message about sending a payslip.
     + handleEmployeeBenefits(): Prints a message about handling employee benefits.
     + handleEmployeeTaxes(): Prints a message about handling employee taxes.
8. **ILogger Interface** (Interface Segregation Principle - ISP)
   * **Purpose**: A general logging interface that defines a contract for logging messages.
   * **Method**:
     + log(string message): A pure virtual function for logging messages.
9. **Specific Logger Classes** (ISP)
   * **ErrorLogger**: Logs error messages.
   * **DebugLogger**: Logs debug messages.
   * **WarnLogger**: Logs warning messages.
   * **InfoLogger**: Logs informational messages.
   * **TraceLogger**: Logs trace messages.
   * **ISP**: Each logger class implements only the logging functionality it needs, ensuring that clients depend only on the specific logger they require.