1. **Identify Design Classes**: Begin by selecting the classes in the domain class diagram that are candidates for implementation.
2. **Add Design Attributes**: Define the data fields (attributes) for each design class.
3. **Add Design Methods**: Identify the methods (operations) based on the use cases, system behaviors, or class responsibilities.
4. **Refine Relationships**: Add necessary navigation arrows, multiplicity, and refine relationships (associations, inheritance, etc.) to reflect the design decisions.

Here's what the process will look like **step-by-step** for each of your **Domain Classes**:

1. Employee Class:

- **Step 1: Identify Design Class**: Employee
- **Step 2: Add Design Attributes**:
  - - EmployeeID : int
  - - Name : String
  - - Position : String
  - - Department : String
  - - Status : String
- **Step 3: Add Design Methods**:
  - + requestLeave() : void
  - + updateStatus() : void
  - + viewPayroll() : Payroll
- **Step 4: Refine Relationships**:
  - Association with LeaveRequest, Payroll, Attendance, PerformanceReview classes.
  - Navigation: Employee can access its related objects.

---

2. Department Class:

- **Step 1: Identify Design Class**: Department
- **Step 2: Add Design Attributes**:
  - - DepartmentID : int
  - - Name : String
- **Step 3: Add Design Methods**:
  - + assignSupervisor() : void
  - + viewEmployees() : List<Employee>
- **Step 4: Refine Relationships**:
  - Association with Supervisor and Employee classes.
  - Navigation: Department manages its employees.

---

3. Admin Class:

- **Step 1: Identify Design Class**: Admin
- **Step 2: Add Design Attributes**:
  - - AdminID : int
  - - Role : String
  - - Permission : String
- **Step 3: Add Design Methods**:
  - + manageEmployee() : void
  - + addHoliday() : void
  - + viewPermissions() : String
- **Step 4: Refine Relationships**:
  - Association with Employee and Holiday classes.
  - Navigation: Admin oversees and manages employees.

---

4. Supervisor Class:

- **Step 1: Identify Design Class**: Supervisor
- **Step 2: Add Design Attributes**:
    - - SupervisorID : int
    - - Permission : String
    - - Department : String
- **Step 3: Add Design Methods**:
    - + reviewEmployeePerformance() : PerformanceReview
    - + recommendCounseling() : void
- **Step 4: Refine Relationships**:
    - Association with Employee, Counseling, and ProbationStatus classes.

---

5. Payroll Class:

- **Step 1: Identify Design Class**: Payroll
- **Step 2: Add Design Attributes**:
    - - PayrollID : int
    - - Salary : Decimal
    - - Deduction : Decimal
    - - NetPay : Decimal
- **Step 3: Add Design Methods**:
    - + calculateNetPay() : Decimal
    - + viewPayrollDetails() : String
- **Step 4: Refine Relationships**:
    - Association with Employee class (one-to-one).

---

6. LeaveRequest Class:

- **Step 1: Identify Design Class**: LeaveRequest

- **Step 2: Add Design Attributes**:
    - - LeaveID : int
    - - Type : String
    - - Duration : int
    - - Status : String
- **Step 3: Add Design Methods**:
    - + submitRequest() : void
    - + checkRequestStatus() : String
- **Step 4: Refine Relationships**:
    - Association with Employee class (many-to-one).

---

7. PerformanceReview Class:

- **Step 1: Identify Design Class**: PerformanceReview
- **Step 2: Add Design Attributes**:
    - - ReviewID : int
    - - Date : Date
    - - Score : int
    - - Comments : String
- **Step 3: Add Design Methods**:
    - + generateReviewReport() : String
- **Step 4: Refine Relationships**:
    - Association with Employee class (many-to-one).

---

8. Counseling Class:

- **Step 1: Identify Design Class**: Counseling
- **Step 2: Add Design Attributes**:

    - - CounselingID : int
    - - Type : String
    - - Recommendation : String
    - - Outcome : String
- **Step 3: Add Design Methods**:
    - + conductCounseling() : void
- **Step 4: Refine Relationships**:
    - Association with Employee and Supervisor.

---

9. Recruiter Class:

- **Step 1: Identify Design Class**: Recruiter
- **Step 2: Add Design Attributes**:
    - - RecruiterID : int
    - - ManagedPositions : String
- **Step 3: Add Design Methods**:
    - + scheduleInterview() : void
    - + postJobAdvertisement() : void
- **Step 4: Refine Relationships**:
    - Association with JobAdvertisement, Candidate, and Interview.

---

10. Candidate Class:

- **Step 1: Identify Design Class**: Candidate
- **Step 2: Add Design Attributes**:
    - - CandidateID : int
    - - Name : String
    - - ApplicationStatus : String

    - - Position : String
- **Step 3: Add Design Methods**:
    - + applyForJob() : void
    - + attendInterview() : void
- **Step 4: Refine Relationships**:
    - Association with JobOffer, Interview, and JobAdvertisement.

---

11. Holiday Class:

- **Step 1: Identify Design Class**: Holiday
- **Step 2: Add Design Attributes**:
    - - HolidayID : int
    - - Date : Date
    - - Description : String
- **Step 3: Add Design Methods**:
    - + assignToEmployee() : void
- **Step 4: Refine Relationships**:
    - Association with Admin and Employee classes.

---

12. Attendance Class:

- **Step 1: Identify Design Class**: Attendance
- **Step 2: Add Design Attributes**:
    - - AttendanceID : int
    - - TardinessCount : int
    - - AbsenceRecord : String
- **Step 3: Add Design Methods**:
    - + markAttendance() : void

- + calculateTardiness() : int
- + viewAttendanceDetails() : String

- **Step 4: Refine Relationships**:
  - Association with Employee (one-to-one).
  - Navigation: Employee can access their Attendance.

---

13. Tardiness Investigation Class:

- **Step 1: Identify Design Class**: Tardiness Investigation
- **Step 2: Add Design Attributes**:
  - - InvestigationID : int
  - - Findings : String
  - - ActionTaken : String
- **Step 3: Add Design Methods**:
  - + conductInvestigation() : void
  - + recordFindings() : void
  - + viewOutcome() : String
- **Step 4: Refine Relationships**:
  - Association with Employee and Supervisor (many-to-one).
  - Navigation: Supervisor conducts investigations related to Employee.

---

14. Probation Status Class:

- **Step 1: Identify Design Class**: ProbationStatus
- **Step 2: Add Design Attributes**:
  - - ProbationID : int
  - - StartDate : Date
  - - EndDate : Date

    - + Status : String
- **Step 3: Add Design Methods**:
    - + assignProbation() : void
    - + updateProbationStatus() : void
    - + viewProbationDetails() : String
- **Step 4: Refine Relationships**:
    - Association with Employee and Supervisor (many-to-one).

---

15. Job Offer Class:

- **Step 1: Identify Design Class**: JobOffer
- **Step 2: Add Design Attributes**:
    - - OfferID : int
    - - Position : String
    - - Salary : Decimal
    - - Terms : String
- **Step 3: Add Design Methods**:
    - + createOffer() : void
    - + sendOfferToCandidate() : void
- **Step 4: Refine Relationships**:
    - Association with Candidate (one-to-one).
    - Navigation: Candidate receives job offers.

---

16. Interview Class:

- **Step 1: Identify Design Class**: Interview
- **Step 2: Add Design Attributes**:
    - - InterviewID : int

    - - Date : Date
    - - Feedback : String
    - - Status : String
- **Step 3: Add Design Methods**:
    - + scheduleInterview() : void
    - + provideFeedback() : void
    - + updateStatus() : void
- **Step 4: Refine Relationships**:
    - Association with Candidate and Recruiter (many-to-one).
    - Navigation: Recruiter schedules and conducts interviews for Candidate.

---

17. Onboarding Class:

- **Step 1: Identify Design Class**: Onboarding
- **Step 2: Add Design Attributes**:
    - - OnboardingID : int
    - - Task : String
    - - Status : String
- **Step 3: Add Design Methods**:
    - + assignTask() : void
    - + updateTaskStatus() : void
    - + viewOnboardingDetails() : String
- **Step 4: Refine Relationships**:
    - Association with Candidate (one-to-one).
    - Navigation: Candidate completes onboarding tasks.

---

18. Job Advertisement Class:

- **Step 1: Identify Design Class**: JobAdvertisement
- **Step 2: Add Design Attributes**:
    - - JobID : int
    - - Position : String
    - - Requirements : String
    - - DatePosted : Date
- **Step 3: Add Design Methods**:
    - + createJobAd() : void
    - + viewJobDetails() : String
- **Step 4: Refine Relationships**:
    - Association with Recruiter (many-to-one).
    - Navigation: Recruiter manages job advertisements.

Final Class Relationships Summary:

- **Employee** has relationships with:
    - LeaveRequest, Payroll, Attendance, PerformanceReview, ProbationStatus.
- **Supervisor** manages:
    - Employee, Counseling, ProbationStatus, TardinessInvestigation.
- **Admin** oversees:
    - Employee, Holiday.
- **Recruiter** interacts with:
    - JobAdvertisement, Interview, Candidate.
- **Candidate** relates to:
    - Interview, JobOffer, Onboarding.
- **JobAdvertisement** belongs to:
    - Recruiter.

19) User Class

1. **Create the User class**:
    - Write the class name in **italics** (*User*).
    - Add the common attributes: UserID, Name, and Role.
    - Add the common methods: login() and logout().
2. **Add the Subclasses**:
    - Place Employee, Admin, Supervisor, and Recruiter below the User class.
    - Use a **generalization arrow** (hollow triangle) pointing **upwards** from each subclass to User.
3. **Refactor Other Classes**:
    - Remove redundant attributes (e.g., Name and Role) from Employee, Admin, Supervisor, and Recruiter since they are inherited from User.