

# Object-Oriented Programming System

Class creation

## Amazon

Employee Class

Code:

```
class Employee
{
    private int employeeId;
    private string employeeName;
    private short employeeAge;
    private string employeeEmailId;
    private float employeeSalary;
    private long employeeMobile;

    public void AddEmployeeDetails()
    {
        //TODO
    }

    public void DisplayEmployeeDetails()
    {
        //TODO
    }

    public void EditEmployeeDetails()
    {
        //TODO
    }

    public void DeleteEmployeeDetails()
    {
        //TODO
    }
}
```

UML Diagram:

Employee		
-	employeeId	: int
-	employeeName	: string
-	employeeAge	: short
-	employeeEmailId	: string
-	employeeSalary	: float
-	employeeMobile	: long
+	AddEmployeeDetails()	: void
+	DisplayEmployeeDetails()	: void
+	EditEmployeeDetails()	: void
+	DeleteEmployeeDetails()	: void

## Customer Class

### Code:

```
class Customers
{
    private int customerId;
    private string customerName;
    private string customerEmailId;
    private string customerAddress;
    private long customerMobile;

    public void AddCustomersDetails()
    {
        //TODO
    }

    public void DisplayCustomersDetails()
    {
        //TODO
    }

    public void EditCustomersDetails()
    {
        //TODO
    }

    public void DeleteCustomersDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Customer		
-	customerId	: int
-	customerName	: string
-	customerEmailId	: string
-	customerAddress	: string
-	customerMobile	: string
+	AddCustomerDetails()	: void
+	DisplayCustomerDetails()	: void
+	EditCustomerDetails()	: void
+	DeleteCustomerDetails()	: void

## Order Class

### Code:

```
class Order
{
    private long orderId;
    private int customerId;
    private long shippingId;
    private double OrderPrice;
    private string customerAddress;

    public void AddOrderDetails()
    {
        //TODO
    }

    public void DisplayOrderDetails()
    {
        //TODO
    }

    public void EditOrderDetails()
    {
        //TODO
    }

    public void DeleteOrderDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Order		
-	orderId	: long
-	customerId	: int
-	shippingId	: long
-	orderPrice	: double
-	customerAddress	: string
+	AddOrderDetails()	: void
+	DisplayOrderDetails()	: void
+	EditOrderDetails()	: void
+	DeleteOrderDetails()	: void

## Product Class

### Code:

```
class Product
{
    private long productId;
    private string productName;
    private string productDescription;
    private double productPrice;
    private long productStock;

    public void AddProductDetails()
    {
        //TODO
    }

    public void DisplayProductDetails()
    {
        //TODO
    }

    public void EditProductDetails()
    {
        //TODO
    }

    public void DeleteProductDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Product		
-	<b>productId</b>	: long
-	<b>productName</b>	: string
-	productDescription	: string
-	productPrice	: double
-	<b>productStock</b>	: long
+	AddProductDetails()	: void
+	DisplayProductDetails()	: void
+	EditProductDetails()	: void
+	DeleteProductDetails()	: void

## ShoppingCart Class

### Code:

```
class ShoppingCart
{
    private long shoppingCartId;
    private int customerID;
    private string customerName;
    private string productName;

    public void AddShoppingCartDetails()
    {
        //TODO
    }

    public void DisplayShoppingCartDetails()
    {
        //TODO
    }

    public void EditShoppingCartDetails()
    {
        //TODO
    }

    public void DeleteShoppingCartDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

ShoppingCart		
-	shoppingCartId	: long
-	customerID	: int
-	customerName	: string
-	productName	: string
+	AddShoppingCartDetails()	: void
+	DisplayShoppingCartDetails()	: void
+	EditShoppingCartDetails()	: void
+	DeleteShoppingCartDetails()	: void

# Apollo

## Hospital Class

### Code:

```
class Hospital
{
    private int hospitalId;
    private string hospitalName;
    private string hospitalType;
    private string hospitalAddress;
    private string doctorName;

    public void AddHospitalDetails()
    {
        //TODO
    }

    public void DisplayHospitalDetails()
    {
        //TODO
    }

    public void EditHospitalDetails()
    {
        //TODO
    }

    public void DeleteHospitalDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Hospital		
-	hospitalId	: int
-	hospitalName	: string
-	hospitalType	: string
-	hospitalAddress	: string
-	doctorName	: string
+	AddHospitalDetails()	: void
+	DisplayHospitalDetails()	: void
+	EditHospitalDetails()	: void
+	DeleteHospitalDetails()	: void

## Doctor Class

### Code:

```
class Doctor
{
    private int doctorId;
    private string doctorName;
    private string doctorType;
    private long doctorMobile;
    private string doctorEmailId;
    private string doctorAddress;

    public void AddDoctorDetails()
    {
        //TODO
    }

    public void DisplayDoctorDetails()
    {
        //TODO
    }

    public void EditDoctorDetails()
    {
        //TODO
    }

    public void DeleteDoctorDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Doctor		
-	doctorId	: int
-	doctorName	: string
-	doctorType	: string
-	doctorMobile	: long
-	doctorEmailId	: string
-	doctorAddress	: string
+	AddDoctorDetails()	: void
+	DisplayDoctorDetails()	: void
+	EditDoctorDetails()	: void
+	DeleteDoctorDetails()	: void

## Patient Class

### Code:

```
class Patient
{
    private int patientId;
    private string patientName;
    private long patientMobile;
    private string patientEmailId;
    private string patientCity;

    public void AddPatientDetails()
    {
        //TODO
    }

    public void DisplayPatientDetails()
    {
        //TODO
    }

    public void EditPatientDetails()
    {
        //TODO
    }

    public void DeletePatientDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Patient		
-	patientId	: int
-	patientName	: string
-	patientMobile	: long
-	patientEmailId	: string
-	patientCity	: string
+	AddPatientDetails()	: void
+	DisplayPatientDetails()	: void
+	EditPatientDetails()	: void
+	DeletePatientDetails()	: void



## Appointment Class

### Code:

```
class Appointment
{
    private int appointmentId;
    private string appointmentType;
    private string appointmentdate;
    private int doctorId;

    public void AddAppointmentDetails()
    {
        //TODO
    }

    public void DisplayAppointmentDetails()
    {
        //TODO
    }

    public void EditAppointmentDetails()
    {
        //TODO
    }

    public void DeleteAppointmentDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Appointment		
-	appointmentId	: int
-	appointmentType	: string
-	appointmentdate	: string
-	doctorId	: int
+	AddAppointmentDetails()	: void
+	DisplayAppointmentDetails()	: void
+	EditAppointmentDetails()	: void
+	DeleteAppointmentDetails()	: void

## Medicine Class

### Code:

```
class Medicine
{
    private int medicineId;
    private string medicineName;
    private string medicineCompany;
    private string MedicineCost;

    public void AddMedicineDetails()
    {
        //TODO
    }

    public void DisplayMedicineDetails()
    {
        //TODO
    }

    public void EditMedicineDetails()
    {
        //TODO
    }

    public void DeleteMedicineDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Medicine		
-	medicineId	: int
-	medicintname	: string
-	medicineCompany	: string
-	medicineCost	: string
+	AddMedicineDetails()	: void
+	DisplayMedicineDetails()	: void
+	EditMedicineDetails()	: void
+	DeleteMedicineDetails()	: void

# Police Station

## Police Class

### Code:

```
class Police
{
    private int policeId;
    private string policeName;
    private string policeDesignation;
    private long policeMobile;
    private string policeEmailId;
    private string policeAddress;

    public void AddPoliceDetails()
    {
        //TODO
    }

    public void DisplayPoliceDetails()
    {
        //TODO
    }

    public void EditPoliceDetails()
    {
        //TODO
    }

    public void DeletePoliceDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Police		
-	policeId	: int
-	policeName	: string
-	policeDesignation	: string
-	policeMobile	: long
-	policeEmailId	: string
-	policeAddress	: string
+	AddPoliceDetails()	: void
+	DisplayPoliceDetails()	: void
+	EditPoliceDetails()	: void
+	DeletePoliceDetails()	: void

## PoliceStation Class

### Code:

```
class PoliceStation
{
    private int policeStationId;
    private long policeStationMobile;
    private string policeStationEmailId;
    private string policeStationCity;
    private string policeStationAddress;

    public void AddPoliceStationDetails()
    {
        //TODO
    }

    public void DisplayPoliceStationDetails()
    {
        //TODO
    }

    public void EditPoliceStationDetails()
    {
        //TODO
    }

    public void DeletePoliceStationDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

PoliceStation		
-	policeStationId	: int
-	policeStationMobile	: long
-	policeStationEmailId	: string
-	policeStationCity	: string
-	policeStationAddress	: string
+	AddPoliceStationDetails()	: void
+	DisplayPoliceStationDetails()	: void
+	EditPoliceStationDetails()	: void
+	DeletePoliceStationDetails()	: void

## FIR Class

### Code:

```
class FIR
{
    private int firNumber;
    private string firName;
    private string firDescription;
    private string firType;

    public void AddFIRDetails()
    {
        //TODO
    }

    public void DisplayFIRDetails()
    {
        //TODO
    }

    public void EditFIRDetails()
    {
        //TODO
    }

    public void DeleteFIRDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

FIR		
-	<b>firNumber</b>	: int
-	<b>firName</b>	: string
-	<b>firDescription</b>	: string
-	<b>firtype</b>	: string
+	AddFIRDetails()	: void
+	DisplayFIRDetails()	: void
+	EditFIRDetails()	: void
+	DeleteFRIDetails()	: void

## Incident Class

### Code:

```
class Incident
{
    private int incidentId;
    private string incidentType;
    private string incidentLocation;
    private string incidentTime;

    public void AddIncidentDetails()
    {
        //TODO
    }

    public void DisplayIncidentDetails()
    {
        //TODO
    }

    public void EditIncidentDetails()
    {
        //TODO
    }

    public void DeleteIncidentDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Incident		
-	incidentId	: int
-	incidentType	: string
-	incidentLocation	: string
-	incidentTime	: string
+	AddIncidentDetails()	: void
+	DisplayIncidentDetails()	: void
+	EditIncidentDetails()	: void
+	DeleteIncidentDetails()	: void

## Prison Class

### Code:

```
class Prison
{
    private string prisonName;
    private string prisonLocation;
    private int prisonCapacity;
    private string prisonSupervisorName;

    public void AddPrisonDetails()
    {
        //TODO
    }

    public void DisplayPrisonDetails()
    {
        //TODO
    }

    public void EditPrisonDetails()
    {
        //TODO
    }

    public void DeletePrisonDetails()
    {
        //TODO
    }
}
```

### UML Diagram:

Prison		
-	<b>prisonName</b>	: string
-	<b>prisonLocation</b>	: string
-	<b>prisonCapacity</b>	: int
-	<b>prisonSupervisorName</b>	: string
+	AddPrisonDetails()	: void
+	DisplayPrisonDetails()	: void
+	EditPrisonDetails()	: void
+	DeletePrisonDetails()	: void