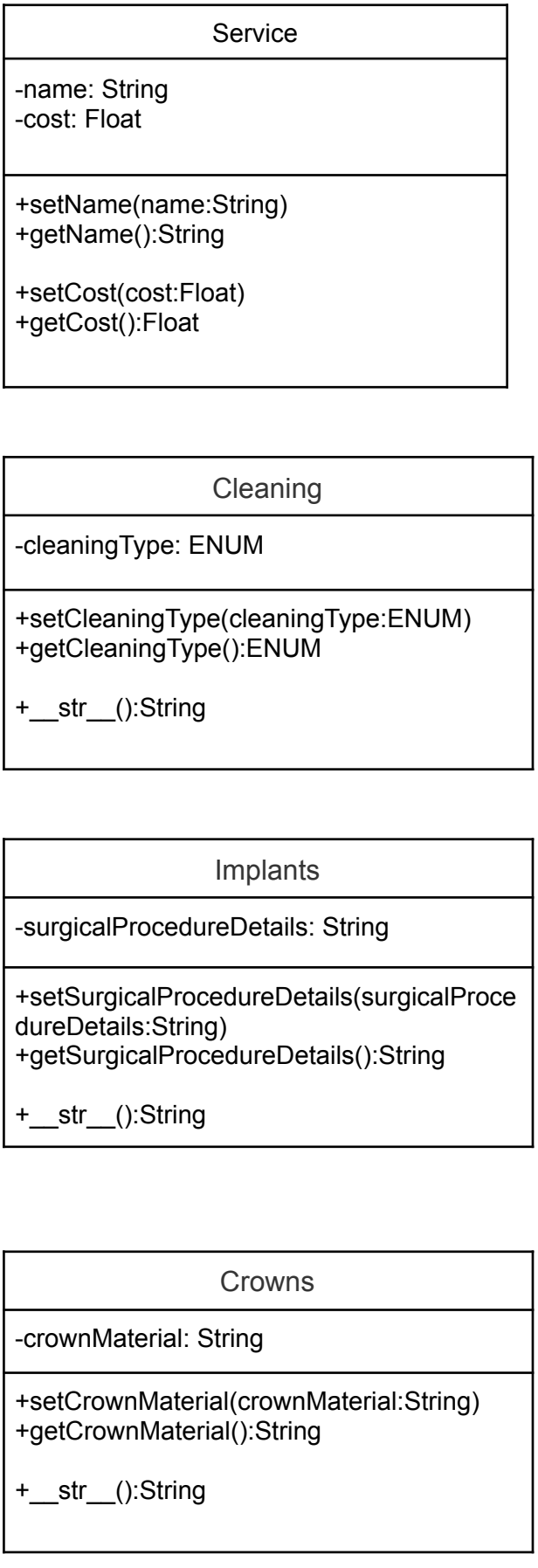
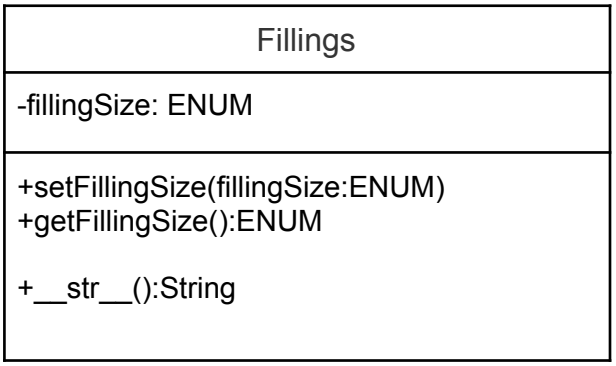
Assignment 2: Software Implementation - OO Project with GUI

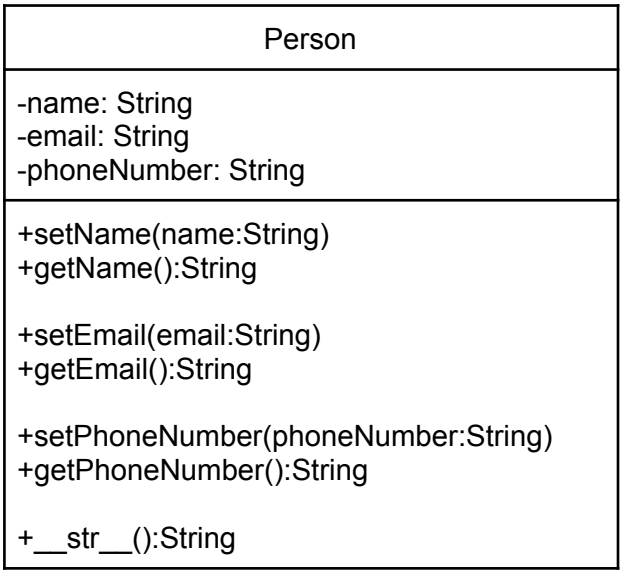
1. **UML Class** Diagram and Description

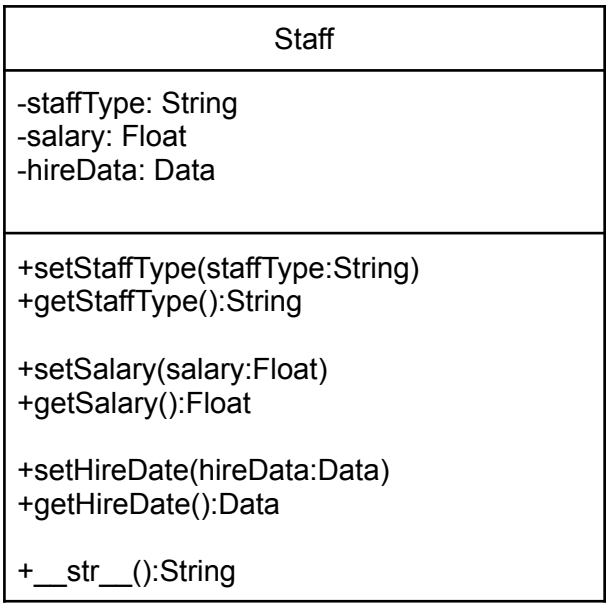
DentalCompany
-name: String -branches: List
+setName(name:String) +getName():String +addBranch(branches:List) +getBranch():List

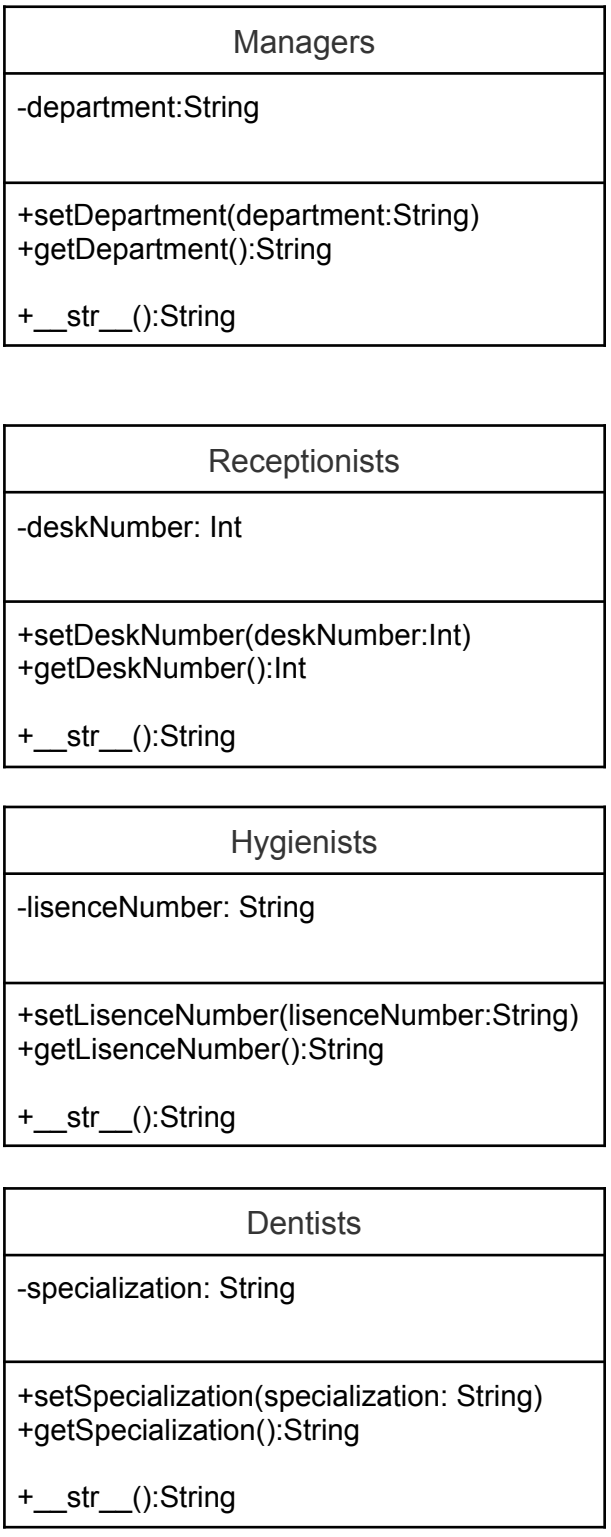
DentalBranch
-address: String -phoneNumber: String -manager: String -staff: List -Service: List -patient: List -appointment: List
+setAddress(address:String) +getAddress():String +setPhoneNumber(phoneNumber:String) +getPhoneNumber():String +setManager(manager:String) +getManager():String +addStaff(staff: List) +getStaff():List +addService(Service:List) +getService():List +addPatient(patient:List) +getPatient():List +addAppointment(appointment:List) +getAppointment():List

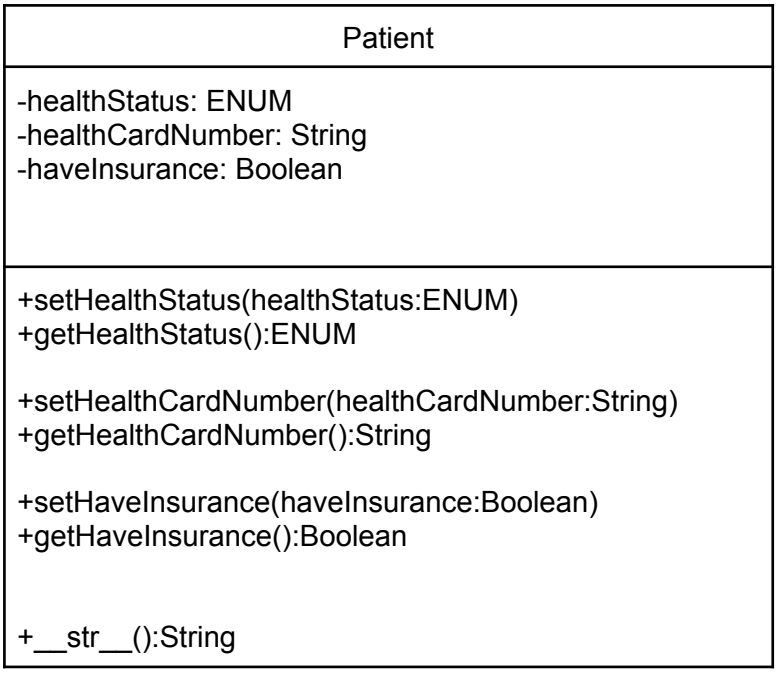

Service
-name: String
-cost: Float
+setName(name:String)
+getName():String
+setCost(cost:Float)
+getCost():Float
Cleaning
-cleaningType: ENUM
+setCleaningType(cleaningType:ENUM)
+getCleaningType():ENUM
+__str__():String
Implants
-surgicalProcedureDetails: String
+setSurgicalProcedureDetails(surgicalProcedureDetails:String)
+getSurgicalProcedureDetails():String
+__str__():String
Crowns
-crownMaterial: String
+setCrownMaterial(crownMaterial:String)
+getCrownMaterial():String
+__str__():String

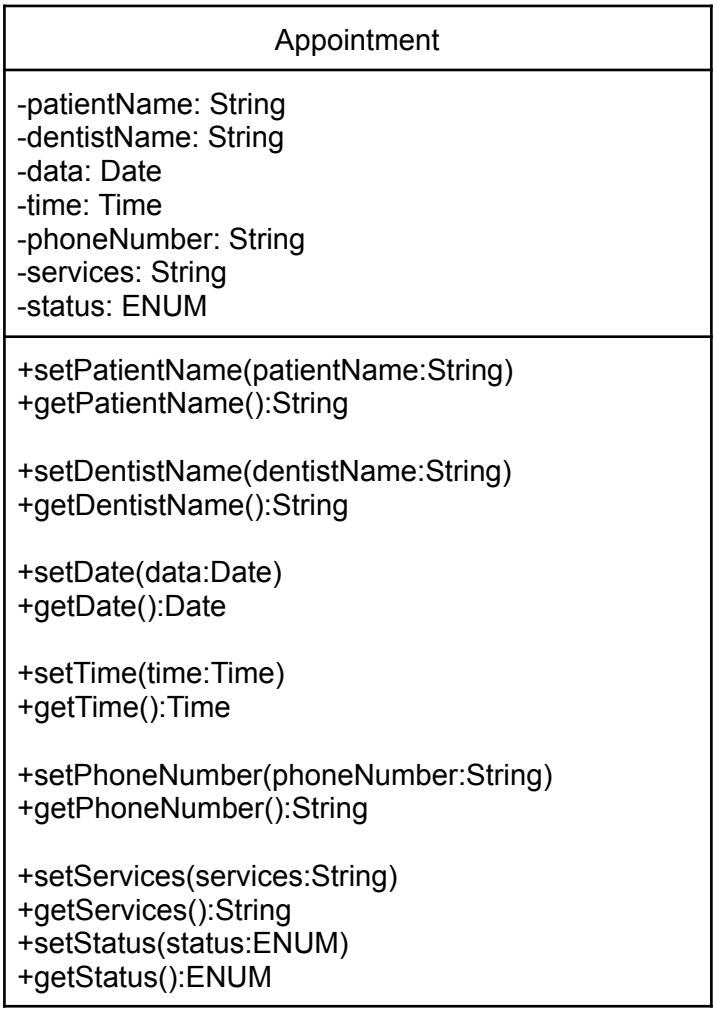

Fillings
-fillingSize: ENUM
+setFillingSize(fillingSize:ENUM)
+getFillingSize():ENUM
+__str__():String

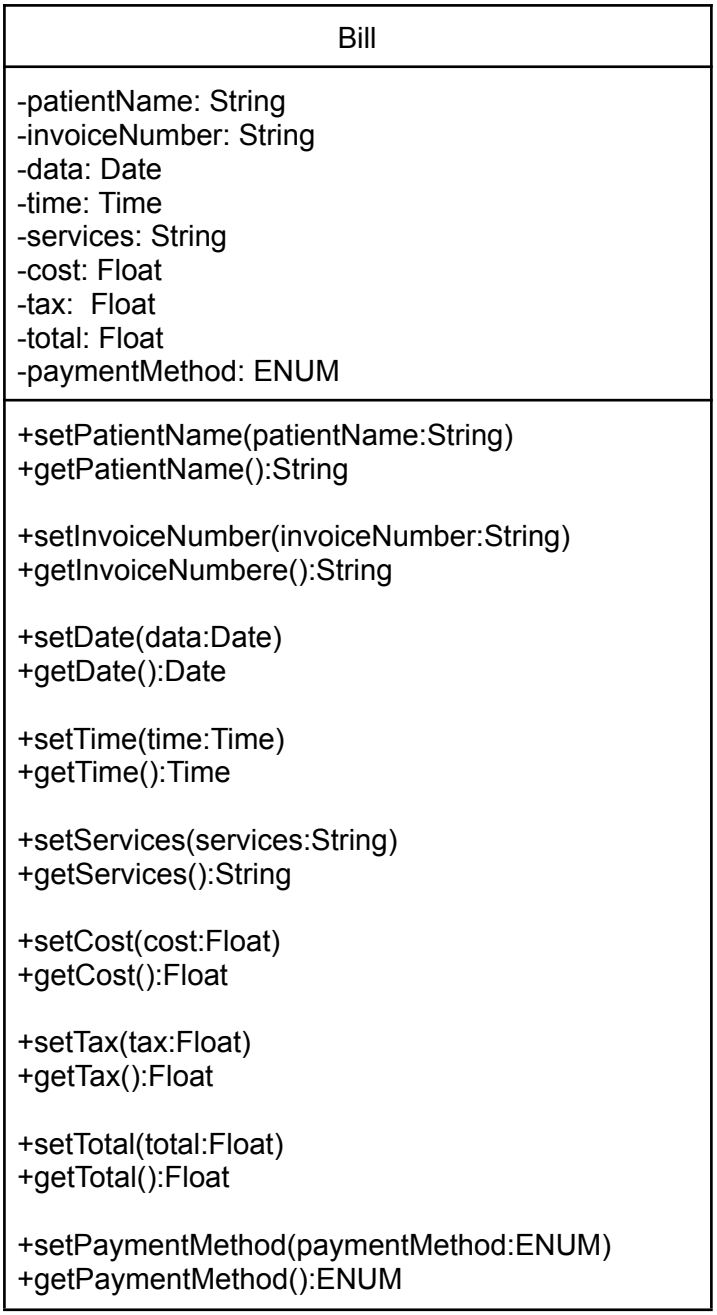

Person
-name: String
-email: String
-phoneNumber: String
+setName(name:String)
+getName():String
+setEmail(email:String)
+getEmail():String
+setPhoneNumber(phoneNumber:String)
+getPhoneNumber():String
+__str__():String

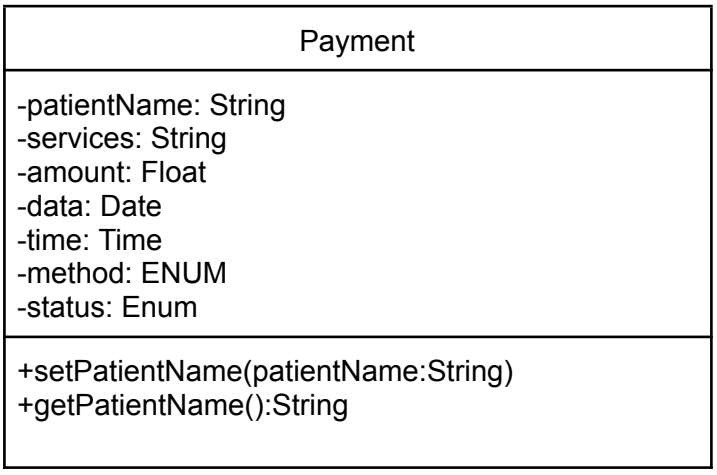

Staff
-staffType: String
-salary: Float
-hireData: Data
+setStaffType(staffType:String)
+getStaffType():String
+setSalary(salary:Float)
+getSalary():Float
+setHireDate(hireData:Data)
+getHireDate():Data
+__str__():String


Managers
-department:String
+setDepartment(department:String)
+getDepartment():String
+__str__():String
Receptionists
-deskNumber: Int
+setDeskNumber(deskNumber:Int)
+getDeskNumber():Int
+__str__():String
Hygienists
-lisenceNumber: String
+setLisenceNumber(lisenceNumber:String)
+getLisenceNumber():String
+__str__():String
Dentists
-specialization: String
+setSpecialization(specialization: String)
+getSpecialization():String
+__str__():String


Patient
-healthStatus: ENUM
-healthCardNumber: String
-haveInsurance: Boolean
+setHealthStatus(healthStatus:ENUM)
+getHealthStatus():ENUM
+setHealthCardNumber(healthCardNumber:String)
+getHealthCardNumber():String
+setHaveInsurance(haveInsurance:Boolean)
+getHaveInsurance():Boolean
+__str__():String


Appointment
-patientName: String
-dentistName: String
-data: Date
-time: Time
-phoneNumber: String
-services: String
-status: ENUM
+setPatientName(patientName:String)
+getPatientName():String
+setDentistName(dentistName:String)
+getDentistName():String
+setDate(data:Date)
+getDate():Date
+setTime(time:Time)
+getTime():Time
+setPhoneNumber(phoneNumber:String)
+getPhoneNumber():String
+setServices(services:String)
+getServices():String
+setStatus(status:ENUM)
+getStatus():ENUM


Bill
-patientName: String
-invoiceNumber: String
-data: Date
-time: Time
-services: String
-cost: Float
-tax: Float
-total: Float
-paymentMethod: ENUM
+setPatientName(patientName:String)
+getPatientName():String
+setInvoiceNumber(invoiceNumber:String)
+getInvoiceNumbere():String
+setDate(data:Date)
+getDate():Date
+setTime(time:Time)
+getTime():Time
+setServices(services:String)
+getServices():String
+setCost(cost:Float)
+getCost():Float
+setTax(tax:Float)
+getTax():Float
+setTotal(total:Float)
+getTotal():Float
+setPaymentMethod(paymentMethod:ENUM)
+getPaymentMethod():ENUM


Payment
-patientName: String
-services: String
-amount: Float
-data: Date
-time: Time
-method: ENUM
-status: Enum
+setPatientName(patientName:String)
+getPatientName():String

+setServices(services:String)
+getServices():String

+setAmount(total:Float)
+getAmountl():Float

+setDate(data:Date)
+getDate():Date

+setTime(time:Time)
+getTime():Time

+setStatus(status:ENUM)
+getStatus():ENUM

Relationships:
dentalCompany has one to many association with **dentalBranch**. One dentalCompany can have many branches, while each dentalBranch is related to one dentalCompany. It is a composition as if the dentalCompany was removed, the branch cannot still exist independently since it is related and managed through the company.
—----------
dentalBranch has one-to-many association with **Staff**. One dentalBranch can have a lot of staff, while each staff is related to one dentalBranch. It is a composition since the branch cannot exist without staff, and staff should be related to a branch. The staff would be impacted and unable to continue operating in the same capacity as before if the branch were to be eliminated or ceased to exist. The branch would also be impacted if its staff were to be fired or depart and would have to hire new employees to fill their positions.
—----------
dentalBranch has many-to-many relationship with **Service.** One dentalBranch can have many services, and each service can be related to multiple dental branches. The same service may be provided by more than one dentalBranch and vice versa. There is no ownership or containment link between the entities, and each one may interact with several instances of the other while remaining separate from it.
—----------
dentalBranch has one-to-many association with **Pateint**. One dentalBranch can have a lot of patients, while each patient is related to one dentalBranch. It is a composition since the branch cannot exist without patients, and a patient should be related to a branch. The patients would be impacted and unable to receive treatment if the branch were to be eliminated or ceased to exist. The branch would also be impacted if it has no patients, it won't be able to gain profit or continue operating.
—----------
dentalBranch has a one-to-many association with **Appointment**. One dentalBranch can have a lot of appointments, while each appointment is related to one dentalBranch. It is a composition since the branch cannot exist without appointments, and an appointment should be related to a branch. The appointment would be impacted if the branch were to be eliminated or ceased to exist. The branch would also be impacted if it has no appointments, it won't be able to gain profit or continue operating, also there will be disorganization.
—----------

The relationship between an **appointment** and a **service** is many-to-one since an appointment may be for a single service but a service may be provided to multiple appointments.

—----------

There is a many-to-one relationship between the **patient** and the **Appointment** because a patient can make several appointments yet only one patient can book an appointment. We can describe it as a unary association indicating that a patient can have an association with multiple appointments.

—----------

The relationship between an **appointment** and a staff is a binary association. Each Appointment is associated with one or more Staff members who are in charge of delivering the services necessary for it to take place. A number of staff members may be involved in delivering a service for a specific appointment, and each staff member may be involved in a number of appointments, making this association a many-to-many relationship.

—----------

There is a one-to-one relationship between **Payment** and **Appointment** as a Payment can be made for a single Appointment but an Appointment can only have one Payment.

—----------

The class **Person** is the parent class and Both **Staff** and **Patient** inherit from it. It is a hierarchical inheritance, where more than one child(Staff and Patient) class is derived from a single-parent class(Person).

—----------

The classes **cleaning, implants, crowns, and fillings** inherit from the class **Service** which is considered a hierarchical inheritance as well.

—----------

The classes **managers, receptionists, hygienists, and dentists** inherit from the class **Staff** which is considered a hierarchical inheritance as well. Each one of those classes inherits from **staff** that inherits already inherits from the class **Person** which is a multilevel inheritance as it is a chain of classes.