School Management System

About Our Project: In this project, a student can be admitted in class six to ten in various section. The school committee can appoint a new teacher and staff. The financial department maintain the payment of teacher and staff and collect fee from student. We store students, teachers, staff information in database system. The goal of making this project is application of five principles of solid. Described in belown:-

- ## Single Responsibility Principle (SRP):

  The Single Responsibility Principle (SRP) states that a class should have only one reason to change, meaning it should only have one responsibility or job, which promotes better modularity and maintainability in software design. We follow SRP in some classes which are Person, Address, Date, Student, Teacher, Staff class.

- ## Open Closed Principle (OCP):

  Open Closed Principle, is a fundamental concept in object-oriented programming that states that software entities (such as classes, modules, and functions) should be open for extension but closed for modification, promoting code reusability and reducing the risk of introducing bugs. We follow OCP in Appointment and EmployeeAppointment class. This class adheres to the Open/Closed Principle (OCP) as it allows for extension by accepting different types of EmployeeAppointment without requiring modification of its existing code. New employee appointment types can be added without changing this class.

- ## Liskov Substitution Principle (LSP):

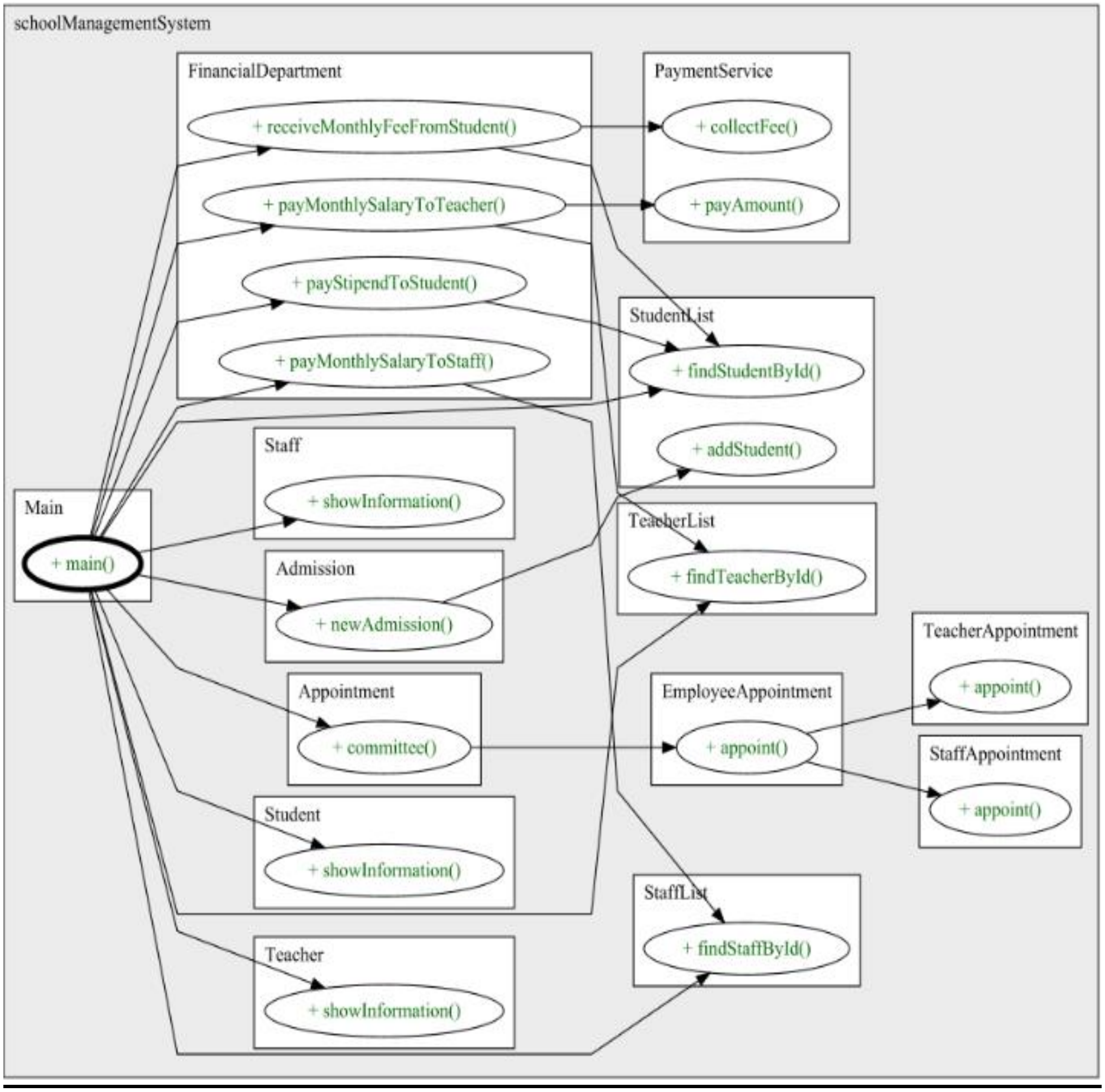
  The Liskov Substitution Principle (LSP) is a key concept in object-oriented design that states that objects of a superclass should be replaceable with objects of a subclass without affecting the correctness of the program, ensuring that subclasses maintain the behavior expected by clients of the superclass. We follow LSP in some classes which are Person, Student, Teacher, Staff class.

- ## Interface Segregation Principle (ISP):

  The Interface Segregation Principle (ISP) is a design principle in object-oriented programming that states that no client should be forced to depend on methods it does not use, promoting the creation of smaller, more specific interfaces rather than large, general-purpose ones. We maintain ISP in some classes which are TeacherAppointment, StaffAppointment, PaymentService class.

- ## Dependency Inversion Principle (DSP):

The Dependency Inversion Principle (DIP) is a software design principle that states that high-level modules should not depend on low-level modules; instead, both should depend on abstractions, and abstractions should not depend on details, but details should depend on abstractions, promoting loose coupling and enhancing code maintainability. We follow DIP in PaymentService Class.


schoolManagementSystem
FinancialDepartment
+ receiveMonthlyFeeFromStudent()
+ payMonthlySalaryToTeacher()
+ payStipendToStudent()
+ payMonthlySalaryToStaff()
PaymentService
+ collectFee()
+ payAmount()
StudentList
+ findStudentById()
+ addStudent()
Main
+ main()
Staff
+ showInformation()
Admission
+ newAdmission()
TeacherList
+ findTeacherById()
TeacherAppointment
+ appoint()
Appointment
+ committee()
EmployeeAppointment
+ appoint()
StaffAppointment
+ appoint()
Student
+ showInformation()
StaffList
+ findStaffById()
Teacher
+ showInformation()