Assignment

Student Name : Likhil N Maiya **UID:** 23BCS11938
Branch: BE-CSE **Section/Group:** KRG_1B
Semester: 6th **Date of Performance:** 31/1/26
Subject Name: System Design
Subject Code: 23CSH-314

Aim:

Q1. Explain SRP and OCP in detail with proper examples.
Q2. Discuss in detail about the violations in SRP and OCP along with their fixes.
Q3. Design an HLD for an Online Examination System applying these principles.

Objectives:

1. To understand SOLID principle inorder to explain what is SRP and OCP
2. To make an .io file representing the Online Examination System.

Single Responsibility Principle (SRP) - The S in SOLID principle

- One class should have only one job.
- If a class does multiple things, changes in one concern can break others.

Real-world analogy: Restaurant

Imagine **one employee** who:

- cooks food
- takes orders
- handles payments
- cleans tables

If the payment system changes → the cook's job breaks.
That's bad design.

Instead:

- **Chef** → cooking
- **Waiter** → orders
- **Cashier** → payments

Each role has **one responsibility**.

Example 2:

In case of notification in an application
if only one method is defined for email sharing, and we need to make email sharing for SMS and whatsapp as well then It might be a problem in the code pasted below
public void sendOtp(String method) {

```
        if (method.equals("EMAIL")) {
            // email API logic
        }
    }
```

rather make an interface of notification and implement classes of SMS email and whatsapp in the application
like this

```
class NotificationService {

    public void sendOtp(String method) {

        if (method.equals("EMAIL")) {
            // send OTP using email API
        }
        else if (method.equals("MOBILE")) {
            // send OTP using mobile API (Twilio, etc.)
        }
    }
}
```

Violation in SRP

A class should have only one responsibility and one reason to change.
One class doing multiple businesses

How to fix that?

Just make different classes for different tasks.

Open Closed Principle (OCP) – O of SOLID

It states that according to any new requirement, the state can only be extended and not be altered or modified.
Class A was originally present
We can make a Class B and extend A into it and use the objects of class B

Example:
when taking a loan, we have different interest rates for different types of loan.
Rather than used if else for each type and then modifying the id else ladder statement inorder to add more loan type we can make one loan interface and separate class for types of loans

Wrong way

```
class LoanService {

    public double getInterestRate(String loanType) {

        if (loanType.equals("HOME_LOAN")) {
            return 7.5;
        }
        else if (loanType.equals("PERSONAL_LOAN")) {
            return 12.0;
        }
        else if (loanType.equals("CAR_LOAN")) {
            return 9.0;
        }

        return 0;
    }
}
```

Right way

```
interface Loan {
    double getInterestRate();
}

class HomeLoan implements Loan {
    public double getInterestRate() {
        return 7.5;
    }
}
```

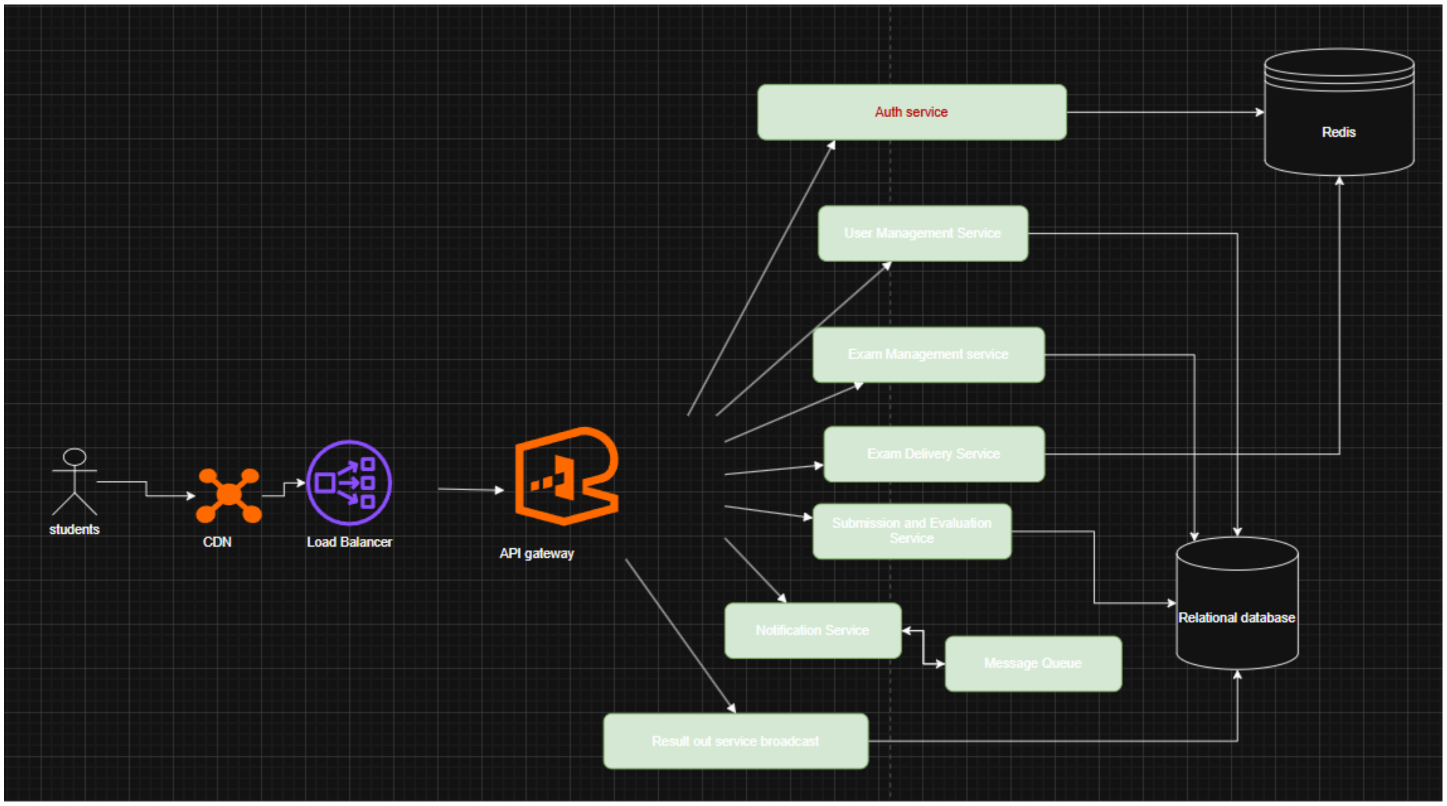
```
class PersonalLoan implements Loan {
    public double getInterestRate() {
        return 12.0;
    }
}

class CarLoan implements Loan {
    public double getInterestRate() {
        return 9.0;
    }
}
```

Violation in OCP

Use of interface and class instead of if else to prevent modification of a case when we need to upgrade or add some feature into a class

How to Fix that?
Use interface and make separate class and implement that main interface class into all those classes.

High level diagram of Online exam system


students
CDN
Load Balancer
API gateway
Auth service
Redis
User Management Service
Exam Management service
Exam Delivery Service
Submission and Evaluation Service
Notification Service
Message Queue
Result out service broadcast
Relational database