

Laboratory Work Nr.0 SOLID PRINCIPLES

**Course: Tehnici și mecanisme de proiectare software**

**Author: Tofan Liviu**

**Group: FAF-223**

# Theory

SOLID is an acronym that stands for five key design principles:

* Single responsibility principle.
* Open-closed principle.
* Liskov substitution principle.
* Interface segregation principle.
* Dependency inversion principle.

SOLID principles seek to reduce dependencies so that engineers can change one area of the software without impacting others. They make it easier to understand, maintain, and extend designs. Software engineers avoid issues and build adaptive, effective, and agile software using these SOLID principles.

# Objectives:

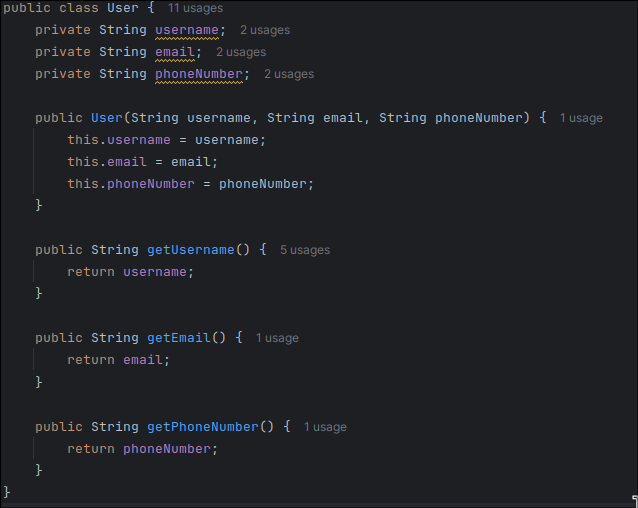
Implement 2 SOLID letters in a simple project.

# Implementation description

I have implemented a simple system where a user can log in, and upon login, the system can send a notification via different channels *(Email, SMS, Push Notification)*. I applied first 2 principles from SOLID, the **SRP** to separate responsibilities for authentication and notification sending. For **OCP**, I can created new notification methods like *Push Notifications* to be added without modifying the existing code.

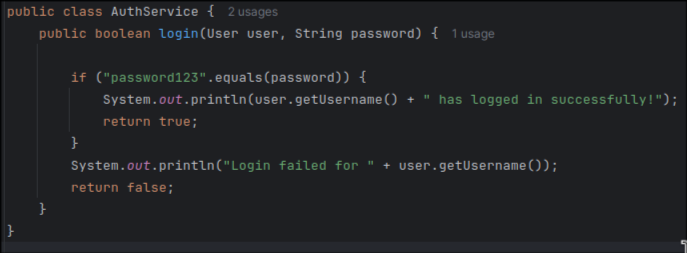
### Project Structure:

1. *User:* Represents the user entity.



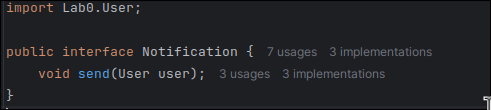
*Figure1. Class User*

1. *AuthService:* Handles user authentication using to **SRP**.

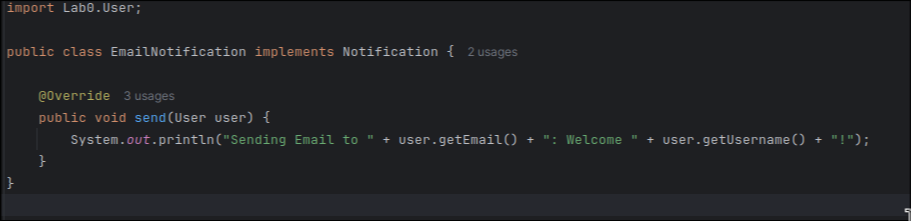


*Figure2. Class AuthService handling Authentification Process*

1. *Notification:* The base interface for sending notifications using tne **OCP**. It can be extended to support different types of notifications.



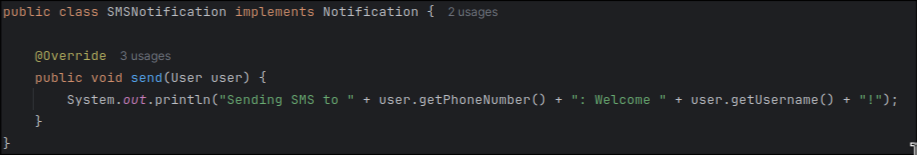
*Figure3. Interface Notification*



*Figure4. Email Notification*

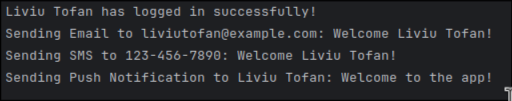


*Figure5. Push Notification*



*Figure6. SMS Notification*

# Results



*Figure7. Login Output*

# Conclusions

I used first two letters from SOLID and implemented them in a simple login/push notification

**SRP**: The AuthService class is only responsible for logging in the user, and the different notification classes are responsible for sending notifications. Each class does one thing and follows the Single Responsibility Principle.

**OCP**: The Notification interface is open for extension. We can add new notification types without modifying the existing notification classes.

So the SOLID principles are fundamental guidelines in software design that help developers create systems that are more maintainable, scalable, and flexible. Principles contribute to building systems that are more robust, easier to test, and adaptable to change.

# References

1. Solid Principles:  
   <https://www.bmc.com/blogs/solid-design-principles/>
2. Solid Principles:

<https://www.digitalocean.com/community/conceptual-articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design>

# References

1. Solid Principles:  
   <https://www.bmc.com/blogs/solid-design-principles/>
2. Solid Principles:

<https://www.digitalocean.com/community/conceptual-articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design>