# Assignment 2 - Fall 2021
# CSCI 5408 - Advance Topics in Software Development

**1.1 Assignment by:**

Full Name: Adesh Nalpet Adimurthy

Banner Number: B00886154

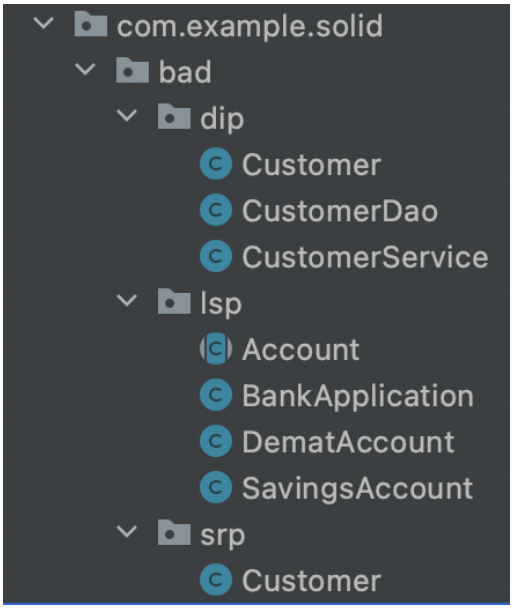
Dal FCS CSID: adimurthy

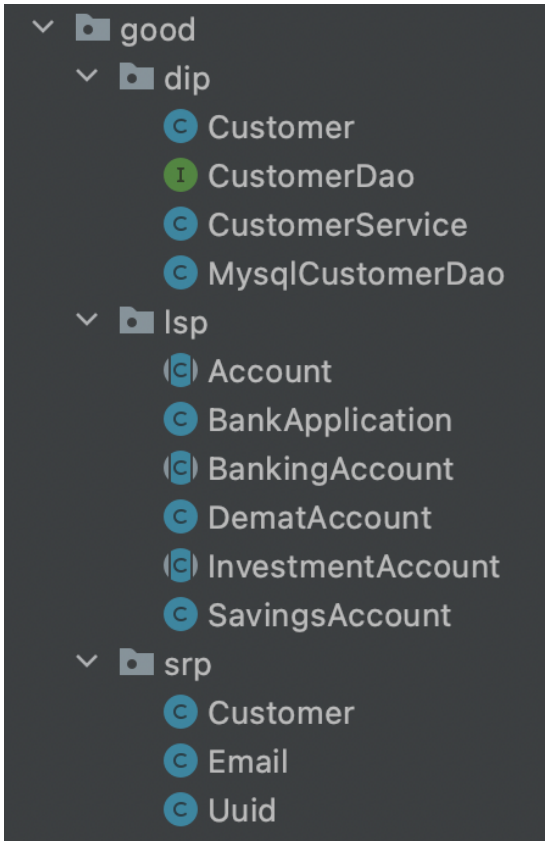
Gitlab Profile: https://git.cs.dal.ca/adimurthy

**1.2 Assignment Gitlab Repository (A2):**

https://git.cs.dal.ca/courses/2021-fall/csci-5308/assignments/adesh_n/-/tree/master/A2

**1.3 Project Structure:** bad and good packages in S.O.L.I.D



**2.1 Models:**

To explain the use of some of the SOLID principles, I have taken the example of customers using a banking application for investments and savings. Hence, in the low-level design, the key classes are Customer, Account, and Application.

In the A2 project sub-directory, there are examples of violations of the Single Responsibility Principle, Dependency Inversion Principle, and Liskov Substitution Principle.

**3.1 Single Responsibility Principle:**

**Violation**: Consider the Customer class in the package "srp," with id, name, and email. There are validations for the format of the id and email, as defined in the regex pattern. While this is valid for the Customer class, it violates the Single Responsibility Principle. Performing validations on the customer entity fields in the same class does not scale and are not the primary functionality of the customer class. The class will eventually end up holding too many responsibilities.

**Solution**: I introduced new classes, Email, and Uuid, which have the validation details. Any additional features and changes in validations would be in their respective classes and not in the customer class. Hence, the responsibility of the Email and Uuid validations are segregated and out of the Customer class. Furthermore, it improves the flexibility and reusability of code.

**3.2 Dependency Inversion Principle:**

**Violation**: Taking the example of the Customer entity again, the CustomerDao is the data access object which fetches/retries data from the datastore (MySQL, for example); however, the CustomerService class, which has the business logic, directly depends on CustomerDao, resulting in higher coupling between low-level and high-level modules. Hence violates the Dependency Inversion Principle.

**Solution**: Introducing abstraction between the low-level and high-level modules is the best way to tackle the violation. So I introduced an interface called CustomerDao and further implemented the MysqlCustomerDao; this way, the coupling between the high and low-level modules is reduced and gives better flexibility to not depend on the internals of the storage layer.

### 3.3 Liskov Substitution Principle:

**Violation**: Taking the example of a Banking application with features to redeem, invest and deposit. The Account class offers all these three functionalities. However, there can be different types of accounts, such as an investment account with no option to "deposit" or savings account without implementation for "invest"; this results in having unsupported operations for sub-classes, refusing to bequest, clearly violating the Liskov Substitution Principle.

**Solution**: To solve the violation, it's important to understand the different types of Account that can exist; at times, solving such problems might require business context. In this case, we can segregate the Account subtypes (abstractions) as Investment Account and Banking Account. An example of an investment account is a Demat Account and Savings Account for Banking. By doing so, the objects of a superclass are replaceable with objects of its subclasses without breaking the application.

**General References:**


1. Dependency Inversion Principle. Accessed on: Nov 3, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Dependency_inversion_principle
2. Single Responsibility Principle. Accessed on: Nov 3, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Single-responsibility_principle
3. Maven Surefire Plugin. Accessed on: Nov 6, 2021. [Online]. Available: https://maven.apache.org/surefire/maven-surefire-plugin
4. Maven Surefire Plugin. Accessed on: Nov 6, 2021. [Online]. Available: https://maven.apache.org/surefire/maven-surefire-plugin
5. Gitlab CI Maven Example. Accessed on: Nov 7, 2021. [Online]. Available: https://docs.gitlab.com/ee/ci/examples/