Software design master rubric

The rubric below collects the design principles from CSSE220 and CSSE374. None of these items are new to students; they've all seen these very criteria and know exactly what is expected of them by each checkbox.

In 374, individual grade items on the list were generally graded using a **pass/fail basis**, with the bar being about **80% correctness for each checkbox**. Grade items were arranged in the rubric in increasing order of sophistication akin to the style of "mastery" assessment.

To earn a given letter grade, students should demonstrate proficiency in the work presented under that letter grade and all lower letter grades.

D: Deficient

- ☐ Complete hand-made UML class diagram describing the program design of the project.
- ☐ Design satisfies the most basic CSSE220 design principles:
  1) Design allows proper functionality
     a) Must be able to store required information
     b) Must be able to access the required information to accomplish tasks
     c) Data should not be duplicated
  2) Structure design around the data to be stored
     a) Nouns should become classes
     b) Classes should have intelligent behaviors that encapsulate their data.

C: Satisfactory

- ☐ Design separate what changes from what stays the same.
- ☐ Design follows the Single Responsibility Principle.
- ☐ README.MD documenting how to run the program.
- ☐ Each student demonstrated his or her individual contributions to the code by committing under his or her GitLab login.

B: Very Good

- ☐ Design favors composition over inheritance.
- ☐ Design programs to interfaces, not implementations.

A: Excellent

- ☐ Strive for loosely coupled designs between objects that interact.
  - Don't().use().method().chains()
- ☐ Classes should be open for extension but closed for modifications.