# COMP 3023 Assignment 2

# Calculate the patient alert levels

**A diagram of a computer

Description automatically generated**

Figure 1: Strategy Design Pattern

**Design Pattern:**  Strategy Design Pattern

The system must be able to utilise the correct algorithm to calculate a patient’s alert level depending on their primary disease. To address this, I chose to utilise the strategy design pattern, which would make it so that each disease alert level algorithm were interchangeable

This design is represented in Figure 1. The AlertCalcStrategy is the common interface that all the algorithms inherit from. This class is used by the AlertCalcContext, which sets and executes the chosen algorithm.

**How it works:**

1. A patient is selected to have new vitals declared for by the user
2. The PatientManagementSystem calls the addVitals() method for the selected patient, passing in a the new vitals class and a boolean set to true
3. The addVitals() method adds the inputted vitals to the patient
4. The status of the inputted Boolean is then checked, if true, addVitals() will set a new alert for the patient based on the inputted vitals by calling the function
5. The alertCalc context and the specific diagnosis’s alert calculation classes are defined
6. The patients primaryDiagnosis is checked, addVitals() calls the setStrategy() function from the context, passing in a reference to the diagnosis strategy
7. After the context is set, addVitals() calls the executeStrategy() function from the context, passing in a reference to itself (a reference to the patient)
8. AlertCalcContext calls the calculateAlert() function from the strategy, which is inherited from the AlertCalcStrategy.
9. The specific strategy checks the vitals, and then calls setAlertLevel() from the inputted patient, passing in the chosen AlertLevel.

**Git commits:**

# Calculate the Alert level for all diseases a patient has

**Design Pattern:** Composite Design Pattern

The composite pattern was chosen for Functional Requirement 2 because having all the diseases share a common interface will make it easy to calculate the alert level.

# Adapter Pattern

The adapter pattern was chosen so that existing patient files and new patient files can be loaded. Allowing possible incompatible interfaces to work together.

# Observer Pattern

The observer pattern was chosen, as it makes sense for an alert system to be flexible and scalable. This makes it so that new stakeholders can be added to the system easily.