DESIGN PATTERNS

Motivation:

Adding a Question feature from Question Bank Module of OAES. A question can be a multiple-choice type, fill in the blanks, match the following, descriptive, etc.

To represent and create all these type question with proper inputs, I have used Factory Pattern to design the feature along with Singleton Pattern to include logging mechanism in the feature.

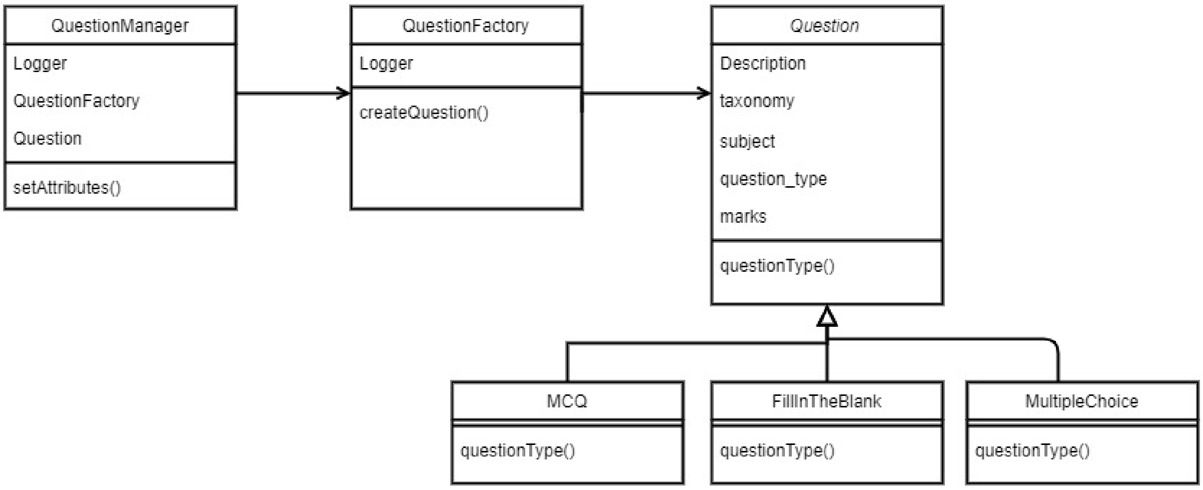
Design:

abstract Question<Description, taxonomy, subject, marks, question_type> is an abstract class with set ters for first 4 attributes and questionType() as abstract method.

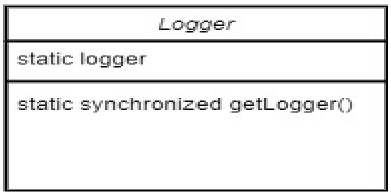
MCQ, FillInTheBlanks, MultipleChoice are the derived concrete product classes which overrides the questionType() method from Question class.

QuestionFactory creates the Question with proper question type and returns the Question object.

QuestionManager invokes QuestionFactory with question_type and calls the setters for the Question object for other attributes.



We need logging for exception handling and user activity monitoring. To avoid creating objects for logging in each of the above classes, I have created a Singleton pattern instance for Logger.


Logger
static logger
static synchronized getLogger()