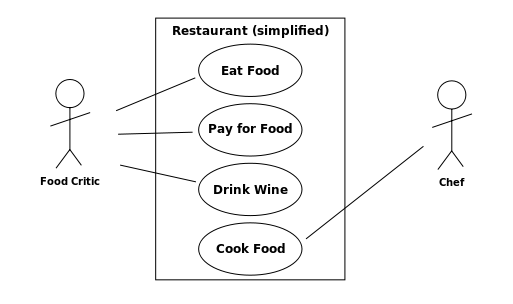
**1) List the different UML diagrams are used for structural modelling. Explain any four sample diagram that are based on behavior modelling.**

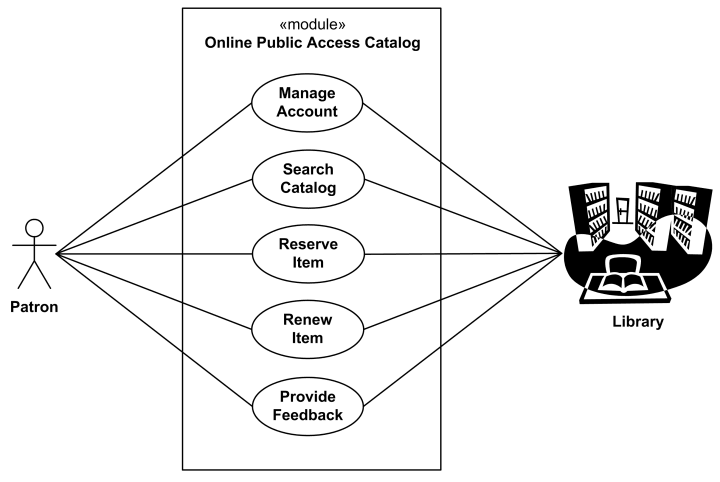
List of UML diagrams used for structural modelling are as follows:

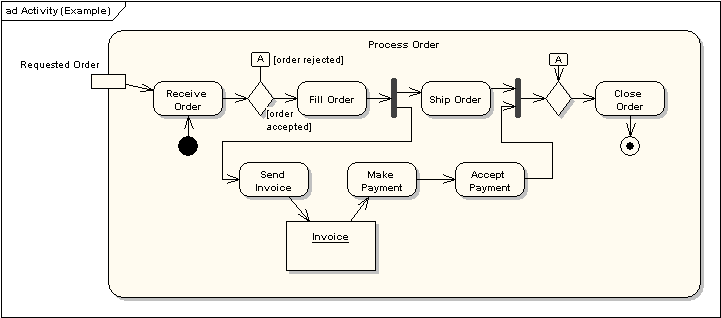
1. Classes diagrams
2. Objects diagrams
3. Deployment diagrams
4. Package diagrams
5. Composite structure diagram
6. Component diagram

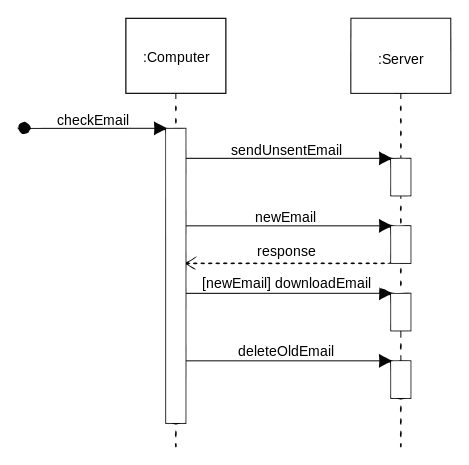
Diagrams based on behavioral modelling are as follows:

1. Activity diagrams
2. Interaction diagrams
3. Use case diagrams
4. State Machine diagrams









**2) Explain different GRASP pattern in detail.**

They are as follows

**1. Controller**

The Controller pattern assigns the responsibility of dealing with system events to a non-UI class that represents the overall system or a use case scenario. A Controller object is a non-user interface object responsible for receiving or handling a system event.

**2. Creator**

Creation of objects is one of the most common activities in an object-oriented system. Which class is responsible for creating objects is a fundamental property of the relationship between objects of particular classes. Simply, "Creator pattern is responsible for creating an object of class".

**3. High Cohesion**

High Cohesion is an evaluative pattern that attempts to keep objects appropriately focused, manageable and understandable. High cohesion is generally used in support of Low Coupling. High cohesion means that the responsibilities of a given element are strongly related and highly focused. Breaking programs into classes and subsystems is an example of activities that increase the cohesive properties of a system. Alternatively, low cohesion is a situation in which a given element has too many unrelated responsibilities. Elements with low cohesion often suffer from being hard to comprehend, hard to reuse, hard to maintain and averse to change.[3]

**4. Indirection**

The Indirection pattern supports low coupling (and reuse potential) between two elements by assigning the responsibility of mediation between them to an intermediate object. An example of this is the introduction of a controller component for mediation between data (model) and its representation (view) in the Model-view-controller pattern.

**5. Information Expert**

Information Expert (also Expert or the Expert Principle) is a principle used to determine where to delegate responsibilities. These responsibilities include methods, computed fields, and so on.

Using the principle of Information Expert, a general approach to assigning responsibilities is to look at a given responsibility, determine the information needed to fulfill it, and then determine where that information is stored.

**6. Low Coupling**

Low Coupling is an evaluative pattern, which dictates how to assign responsibilities to support:

lower dependency between the classes,

change in one class having lower impact on other classes,

higher reuse potential.

**7. Polymorphism**

According to Polymorphism, responsibility of defining the variation of behaviors based on type is assigned to the types for which this variation happens. This is achieved using polymorphic operations.

**8. Protected Variations**

The Protected Variations pattern protects elements from the variations on other elements (objects, systems, subsystems) by wrapping the focus of instability with an interface and using polymorphism to create various implementations of this interface.

**9. Pure Fabrication**

A Pure Fabrication is a class that does not represent a concept in the problem domain, specially made up to achieve low coupling, high cohesion, and the reuse potential thereof derived (when a solution presented by the Information Expert pattern does not). This kind of class is called "Service" in Domain-driven design.