**Factory Pattern Description**

Factory pattern defines an interface for creating an object, but subclasses were given the responsibility of instantiation by factory methods. To keep it simple we will create an object that can create other objects. Where it delegates the instantiation responsibility to subclasses.

For example, the user must deal with multiple classes depending on the role of Organizer or Attendee

Like

Despite common information that can be called as user, user as an attendee can request Event Id in search to get the list of possible events he prefers to attend and should have details of events he is attending, while User as Organizer should get details like organizing event ID’s, attendees list…et.

**In Project**

Participants in Factory Pattern:

User, organizer, attendee, events

User: A Generic class that have multiple generic methods that returns multiple outputs.

Organizer: extended from user where it has a unique getOrganizedEvents () method returns list of events.

Attendee: extended from user where it has a unique getAttendeeEventids () method return list of events id’s that attendee is attending.

Events: Depending on calling function returns information related to events.