Este documento fue creado con el fin de justificar y explicar el uso de patrones de diseño, en este caso patrones GoF

Creational:

Builder: La clase constructor cliente normal y los constructores de ingredientes implementan este patrón, esta clase tiene el propósito de crear una nueva instancia de una clase hija de usuario. Su único propósito es crear este objeto.

Factory: Puedes ver este patrón implementado en tres clases ahora, con la ampliación se usan en fabrica ingrediente, fabrica mesa y fabrica motivo reserva, estas clases son factory, porque ellas pueden instanciar múltiples tipos de clases de una misma abstracción

Behavior

Visitor. Nosotros decidimos implementar visitor para extender la funcionalidad de hacer reportes de la misma clase sin extender su comportamiento. O sea mantener publico y ordenado el acceso al información relevante de la clase.

Strategy: En la nueva ampliación para uso de inventario, tenemos solidos no contables y solidos contables, donde su cantidad será obtenida de maneras distintas, aquí es donde tenemos el método “obtenerCantidad” que según sea solido contable o noContable se aplicara la estrategia correspondiente.

Structural

Facade: Queremos esconder la lógica de negocio, para eso hicimos una capa de servicios, así los controladores, no conocen el negocio, solo conocen su servicio correspondiente y hacen uso de estos para obtener la data de presentación necesaria en sus respectivo controllers.