I have observed two patterns chain of responsibility pattern and observer pattern.

As soon as the customer enters the restaurant he registers with Queue manager of the restaurant. Restaurant sends TableReady() flag to the QueueManager whenever there is a free table in the restaurant.

The restaurant class doesn't refer to a particular receiver class directly. Instead, it refers to QueueManager for assigning a Table (QueueManager.handleTable())), which makes the restaurant independent of which receiver handles the request. Since Restaurant is acting as a sender object and receivers (Receiver1 and Receiver2) are receiving objects this can be assumed as Chain of responsibility pattern. The request here is handled by chain of receivers and each receiver object is made aware of its successor.

Since the Restaurant class notifies the QueueHandler whenever there is a free table , QueueHandler can be treated as observer where restaurant is subject and observer pattern is noticed in this part.

**Summary :**

Customer registers with QueueManager with all the information and whenever there is a free table restaurant notifies that to QueueManager(Observer pattern) and QueueManager has chain of receiver objects which handles the request for allotment of table(Chain of Responsibility).

**CRC for Customer**

|  |  |
| --- | --- |
| Customer | |
| Register with name , phone number and number of people.  Confirm the table.  Leave the restaurant. | QueueHandler |

**CRC for QueueHandler**

|  |  |
| --- | --- |
| QueueHandler | |
| Send the request to receivers objects when table is ready.  Store the registered customer data. | Receiver (Receiver1 and Receiver2).  Customer |

**CRC for Restaurant**

|  |  |
| --- | --- |
| Restaurant | |
| Inform Queue Manager with table availability | QueueHandler |

**CRC for Receiver**

|  |  |
| --- | --- |
| Receiver | |
| Handle Table request by checking number of seats  Allot Table request. | QueueHandler |