INHERITANCE UML

BOOKINGS

- bookingID: int
- customerID: int
- customerName: String
- numberOfGuests: int
- dateNTime: datetime
- email : String

+ addBookings(customerID,bookingID,firstName, lastName, numberOfGuests,time,duration,approval, customerAddress,date) : void
+ cancelBooking(BookingID) : void
+ waitlist(): void
+ bookEvents (customerID,firstName,lastName, numberOfGuests,time,duration,approval,customerAddress, date) : void
+ bookDining (customerID,firstName,lastName, numberOfGuests,time,duration,approval,customerAddress, date) : void
+ isBookingAvailable() :boolean
+ isExtendRequest(customerId: int): boolean
+ sendAllocationRequest (eventTablesId : int, dineInTablesId : int): void

DINING

- bookingDuration: int

+ isExtendRequest(customerId: int): boolean

EVENTS

- eventTitle: String
- eventDescription: string
- eventType: String
- eventDuration: int

- additionalEventInfo() : string
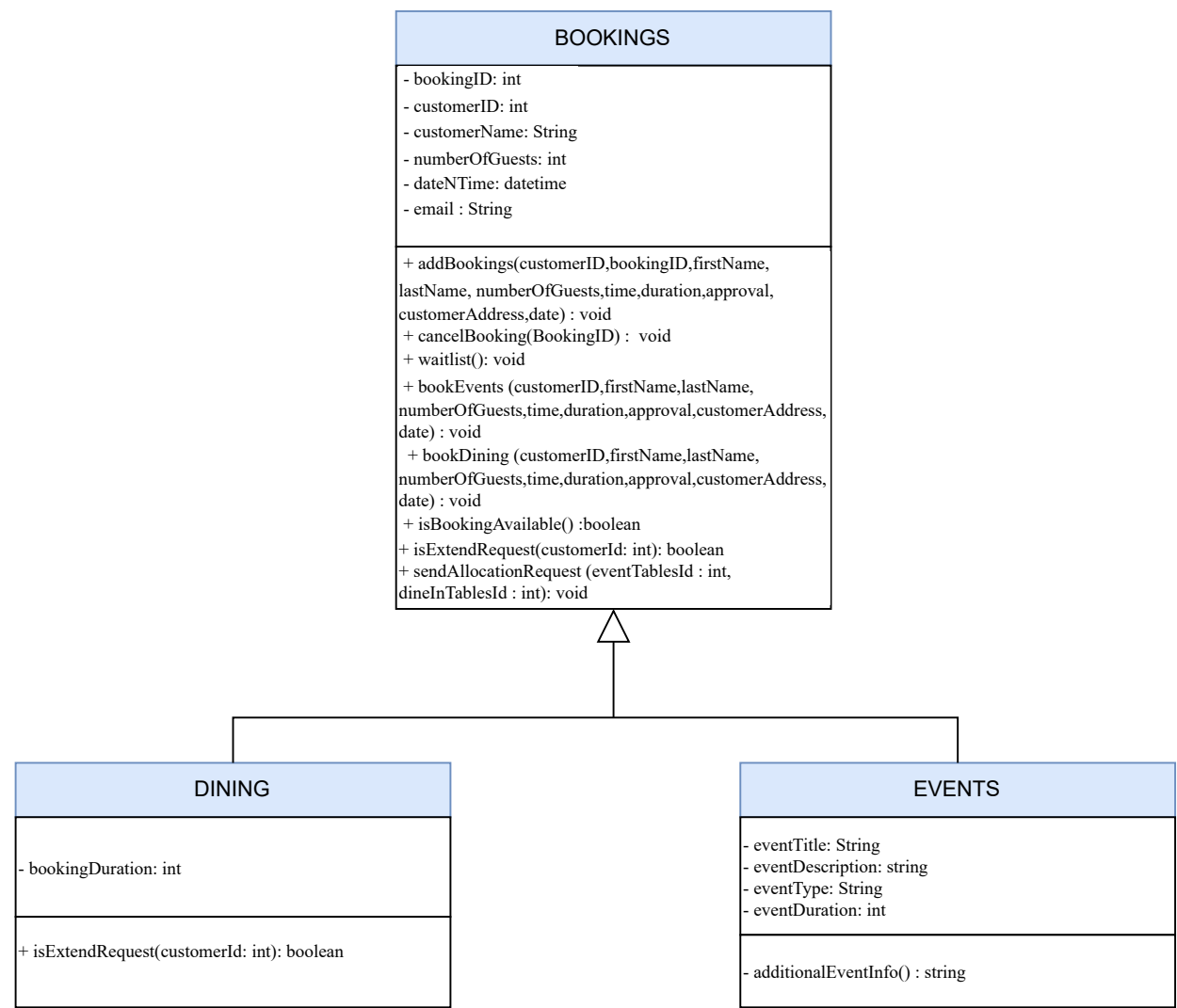
Description

The booking class is responsible for managing the reservations for customers. Placing the Booking class as a parent with Dining and Events as child classes in an inheritance hierarchy is advantageous as it promotes code reusability by allowing the child classes to inherit common properties and methods from the Booking class, reducing redundancy and improving maintainability. It also provides a clear and structured organization of classes, enhancing readability and facilitating modifications if needed. Additionally, this design choice promotes polymorphism, enabling flexibility in handling different types of bookings while maintaining a consistent interface.