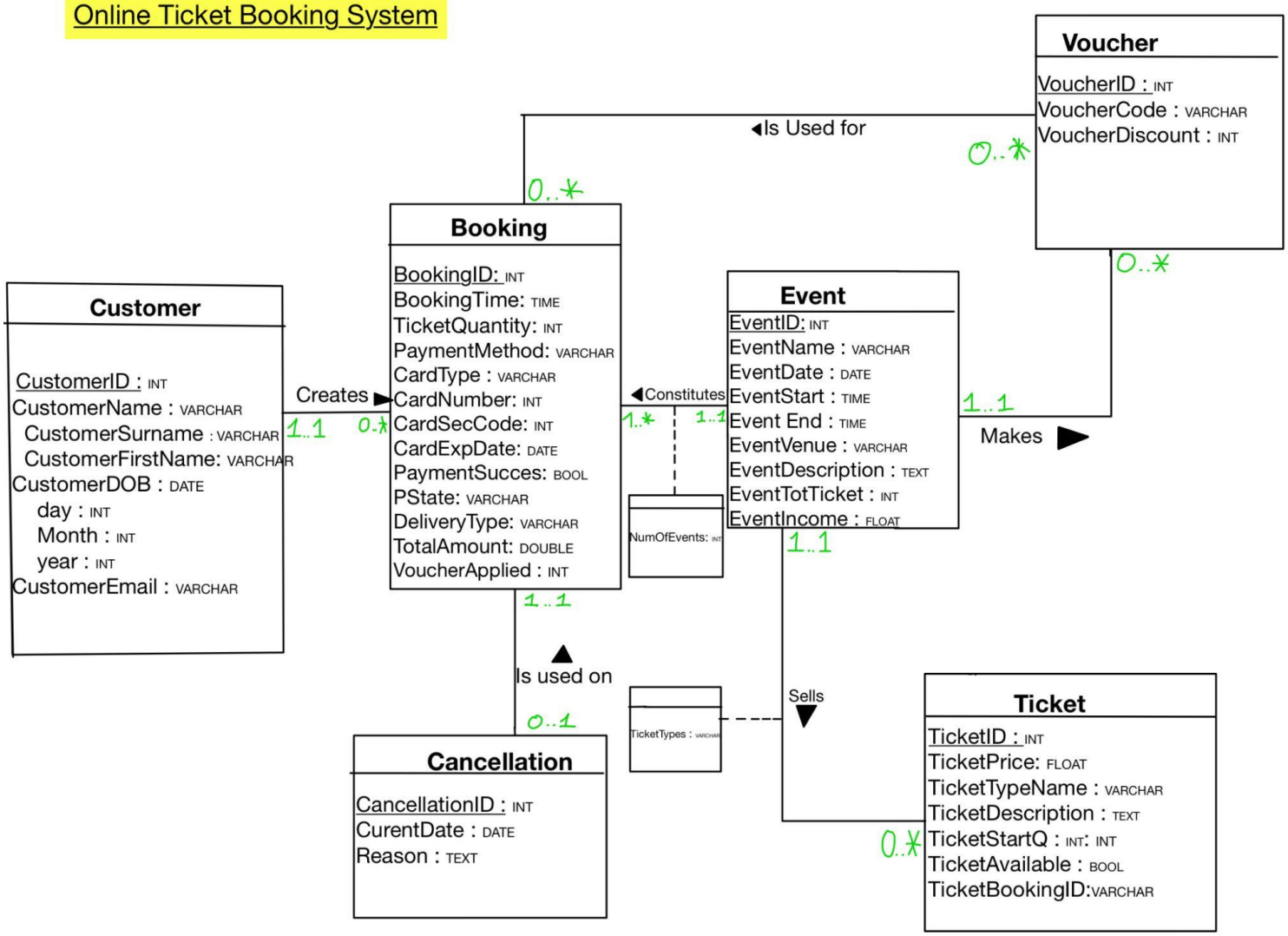
Online Ticket Booking System



ERD using UML notation and Specialisation of a Online Ticket Booking System

Comprehensive description and rationale for the design:

1. **Events (Entity)**:
   - Attributes: EventID (Primary Key), EventName, EventDate, EventStart, EventEnd,EventVenue, EventDescription,EventTotTickets EventIncome

   - Rationale: The "Event" entity represents the events available for booking by a customer. Each event is uniquely identified by an "EventID." The “EventStart” and “EventEnd” attributes are to keep track of starting and ending times.The “EventIncome” attribute is the sum of the price of all ticket types sold. “EventVenue” is the place where the event takes place.Finally “EventTotTickets” is the num of tot tickets regardless of type for the event.

2. **Ticket (Entity):**
- Attributes: TicketID (Primary Key), EventID (Foreign Key), TicketTypeName, TicketPrice,TicketDescription, TicketStartQ, TicketAvailable,TicketBookingID.

- Rationale: The "Ticket" entity represents the various ticket types available for each event. It is related to the "Event" entity via the "EventID" foreign key. This design allows for events with multiple ticket types and different prices. The “TicketStartQ " attribute is the starting quantity of tickets available. “TicketTypeName” is the name of the ticket category. Finally “TicketAvailable” is used to see if the individual ticket is already sold.”TicketBookingID” Is The PK of Booking that will bind tickets to Bookings.

3. **Customer (Entity)**:
- Attributes: CustomerID (Primary Key), CustomerName made of CustomerSurname and CustomerFirstname,CustomerDOB made of day, month and year, CustomerEmail.

- Rationale: The "Customer" entity represents the customers who book tickets. Each customer is uniquely identified by the "CustomerID." The “CustomerEmail” attribute is used in case the Email delivery method is chosen. “Customer DOB” is used to identify the customer and is used to place him categories when ticket types are based on age.

4. **Booking (Entity):**
- Attributes: BookingID (Primary Key), CustomerID (Foreign Key),EventID (Foreign Key), VoucherID(Foreign Key),BookingTime, TotalAmount,TicketQuantity, PaymentSuccess, PaymentMethod, CardType, CardNumber, CardSecCode, CardExpDate, PState, DeliveryType, VoucherApplied

- Rationale: The "Booking" entity represents individual bookings made by customers. It is related to the "Event", "Customer” and “Voucher” entities through foreign keys. The "PaymentSucces" attribute can be used to track whether the booking is successful or not. Payment details are stored in this entity for reference. “TicketQuantity” will be used to keep track of the number of tickets for sale. “PState” will be used to show if a booking is canceled or if it is active. “DeliveryType” is used to see how the customer wants to get their booking. “VoucherApplied” is used to tell how much discount is applied on the booking.

5. **Voucher (Entity):**
- Attributes: VoucherID (Primary Key), EventID (Foreign Key), VoucherCode, VoucherDiscount.

- Rationale: The "Voucher" entity represents voucher codes associated with specific events. It is related to “Event” entities through a foreign key These codes can be used to apply discounts to bookings for events. “VoucherCode” is the code one should enter to get the discount.

6. **Cancelation (Entity)**:
   - Attributes: CancellationID (Primary Key), BookingID (Foreign Key), CurrentDate, Reason

   - Rationale: The "Cancellation" entity is introduced to record cancellations. When a customer requests to cancel a booking, a new record is created. It is related to “Booking” entities through a foreign key. The "CurrentDate" provides information about when the cancellation was requested, and the "Reason" attribute can be used to capture the reason for the cancellation.

Relationships and Cardinality:

Event to Ticket Relationship:

- An "Event" can Sell many "Tickets" associated with it, indicating that multiple tickets can be available for a single event. The cardinality of the relationship is typically represented as **"One-to-Many" (1 to *)**, indicating that one event can have multiple tickets.

Ticket to Event Relationship:

- Conversely, each "Ticket" is associated with only one "Event." Each ticket is specific to a particular event. The cardinality of the relationship is typically represented as **"One-to-One" (1 to 1)**, indicating that one ticket is linked to a single event.

Customer to Booking Relationship:

- A "Customer" can create zero to many "Bookings." This means that a customer can make multiple bookings. The cardinality in this direction can be represented as **"One-to-Many" (1 to *)**, indicating that one customer can have multiple bookings.

Booking to Customer Relationship:

- Conversely, a "Booking" is associated with one and only one "Customer." Each booking is made by a specific customer. The cardinality in this direction is **"One-to-One" (1 to *)**, indicating that one booking is linked to one customer.

Booking to Event Relationship:

- A "Booking" can be associated with one "Event." Each booking can be made for a single event, so the cardinality in this direction is **"One-to-One" (1 to 1)**, indicating that one booking is linked to one event.

Event to Booking Relationship:

- An "Event" can constitute many "Bookings." Multiple customers can make bookings for the same event, so the cardinality in this direction is **"One-to-Many" (1 to *)**, indicating that one event can have multiple bookings

Event to Voucher Relationship:

- An "Event" can make zero to many "Voucher" which means that each event can be associated with multiple voucher codes. This relationship cardinality is typically represented as **"One-to-Many" (1 to *)**, indicating that one event can have zero to multiple voucher codes.

Voucher to Event Relationship:

- Conversely, a "Voucher" can be associated with only one "Event." Each voucher code is specific to a particular event. This relationship cardinality is typically represented as **"One-to-One" (1 to 1)**, indicating that one voucher is linked to a single event.

Booking to Voucher Relationship:

- A "Booking" can be associated with zero to many "Vouchers." This means that multiple vouchers can potentially be applied to a single booking. The cardinality in this direction can be represented as **"One-to-Many" (1 to *)**, indicating that one booking can have multiple vouchers, and one voucher can be associated with multiple bookings.

Voucher to Booking Relationship:

- Conversely, a "Voucher" is used for a single "Booking." One voucher can be used for one Booking to apply discounts. The cardinality in this direction is **“One-to-One" (1 to 1)**, indicating that one voucher can be used for one booking.

Booking to Cancellation Relationship:

- A "Booking" can be associated with zero or more "Cancellations." Not all bookings may have associated cancellations, and a booking can potentially have one cancellation request. The cardinality in this direction can be represented as **"One-to-One" (1 to 1)**, indicating that one booking can have a single cancellation, or zero if not canceled.

Cancellation to Booking Relationship:

- Conversely, a "Cancellation" can be used with exactly one "Booking." Each cancellation request is linked to a single booking. The cardinality in this direction is **"One-to-One" (1 to 1)**, indicating that one cancellation is associated with one booking.

Logical Relational Instance and Schema Representation

Diagram Of relation Instance below (at the end of page 5)

Event (EventID, EventName, EventDate, EventStart, EventEnd,EventVenue, EventDescription, EvenTotTIcket, EventIncome)
PrimaryKey: EventID

Ticket (TicketID, EventID , TicketTypeName, TicketPrice,TicketDescription, TicketStartQ,, TicketAvailable,TicketBookingID)
PrimaryKey: TicketID
Foreign Key: EventID references the “Event” entity’s EventID.

Customer (CustomerID, CustomerName(CustomerSurname and CustomerFirstname), CustomerDOB(day, month and year) CustomerEmail)
Primary Key: CustomerID
Alternate Key : CustomerEmail

Booking (BookingID , TicketID, CustomerID ,EventID, VoucherID,BookingTime, TotalAmount,TicketQuantity, PaymentSuccess, PaymentMethod, CardType, CardNumber, CardSecCode, CardExpDate, PState, DeliveryType, VoucherApplied)
PrimaryKey: BookingID
Foreign Key: TicketID references the "Ticket" entity's TIcketID.
CustomerID references the "Customer" entity's CustomerID.
EventID references the "Event" entity's EventID.
VoucherID references the "Voucher" entity's VoucherID.

Voucher (VoucherID, EventID, VoucherCode, VoucherDiscount)
PrimaryKey: VoucherID
Foreign Key: EventID references the "Event" entity's EventID.

Cancellation (CancellationID, BookingID, CurrentDate, Reason)
PrimaryKey: CancellationID
Foreign Key: BookingID references the "Booking" entity's BookingID.

Online Ticket Booking System

Customer

CustomerID	CustomerFName	CustomerLName	Day	Month	Year	CustomerEmail
Primary Key						

Cancellation

CancellationID	CurrentDate	Reason	BookingID
Primary Key			Foreign Key

Ticket

TicketID	TicketTypeName	TicketPrice	TicketDescription	TicketStartQ	TicketCurrrentQ	TicketAvalable	TicketBookingID	EventID
Primary Key								Foreign Key

Booking

BookingID	BookingTime	TotaAmount	TicketQuantity	PaymentSucces	CardType	PayementMethod	CardNum	CardExpDate	CardSecCode	State	DeliveryType	VoucherApplied	CustomerID	EventID	VoucherID
Primary Key													Foreign Key	Foreign Key	Foreign Key

Event

EventID	EventName	EventDate	EventStart	EventEnd	EventVenue	EventDescription	EventTotTickets	EventIncome
Primary Key								

Voucher

VoucherID	VoucherCode	VoucherDiscount	EventID
Primary Key			Foreign Key