# RAILWAY SYSTEM

Railway Reservation System is a system used for booking tickets over internet.Any Customer Can book tickets for different trains.Customer can book a ticket only if the tickets are available.Customer searches for the availability of tickets then if the tickets are available he books the tickets by initially filling details in a form.Tickets can be booked in two ways by i-ticket or by e-ticket booking.

In case of i-ticket booking customer can book the tickets online and the tickets are couriered to Particular customer at their address.But in case of e-ticket booking and cancelling tickets are booked and cancelled online sitting at the home and customer himself has to take print of the ticket but in both the cases amount for tickets are deducted from customers account.

For cancellation of ticket the customer has to go at reservation office than fill cancellation form and ask the clerk to cancel the ticket than the refund is transferred to customer account.After booking ticket the customer has to checkout by paying fare amount to clerk.

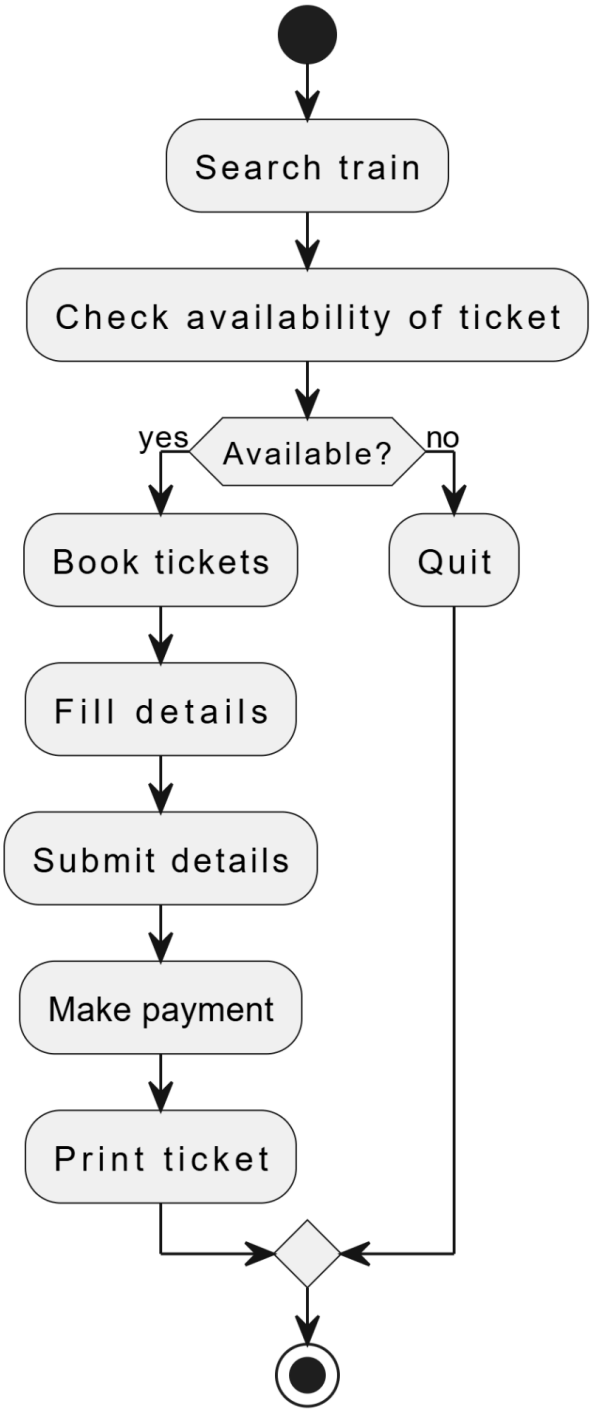
## UML diagrams:

### 1)ACTIVITY DIAGRAM FOR BOOKING TICKET

**CODE:**

```
@startuml
start
:Search train;
:Check availability of ticket;
if (Available?) then (yes)
  :Book tickets;
  :Fill details;
  :Submit details;
  :Make payment;
  :Print ticket;
else (no)
  :Quit;
endif
stop
@enduml
```

**Diagram:**

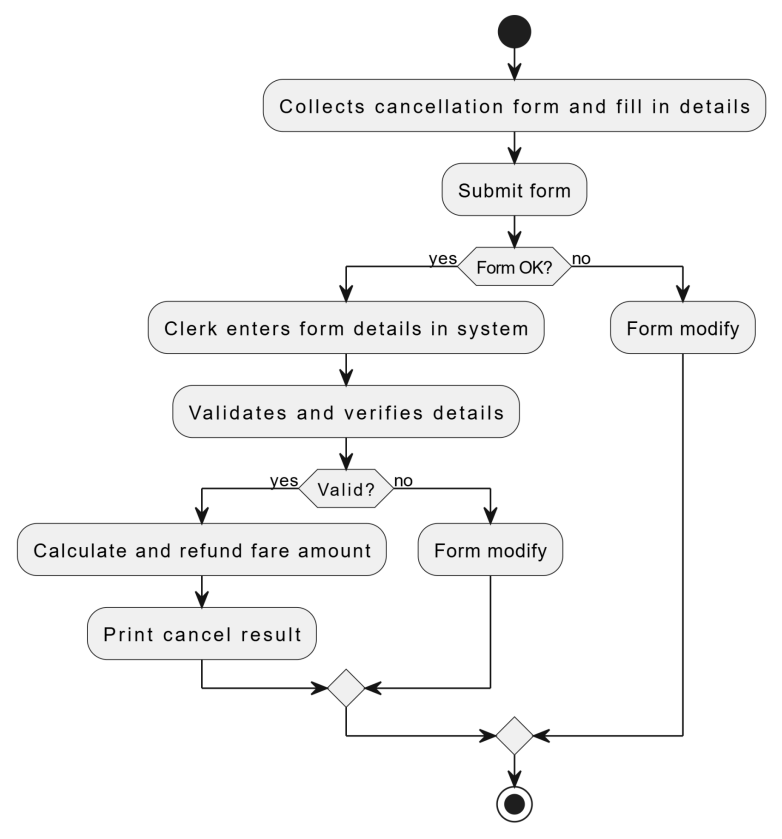

Search train
Check availability of ticket
yes
Available?
no
Book tickets
Quit
Fill details
Submit details
Make payment
Print ticket

**2)ACTIVITY DIAGRAM FOR CANCELLING TICKETS**

**CODE:**

```
@startuml
start
:Collects cancellation form and fill in details;
:Submit form;
if (Form OK?) then (yes)
  :Clerk enters form details in system;
  :Validates and verifies details;
  if (Valid?) then (yes)
    :Calculate and refund fare amount;
    :Print cancel result;
  else (no)
    :Form modify;
  endif
else (no)
  :Form modify;
endif
stop
@enduml
```

**DIAGRAM**



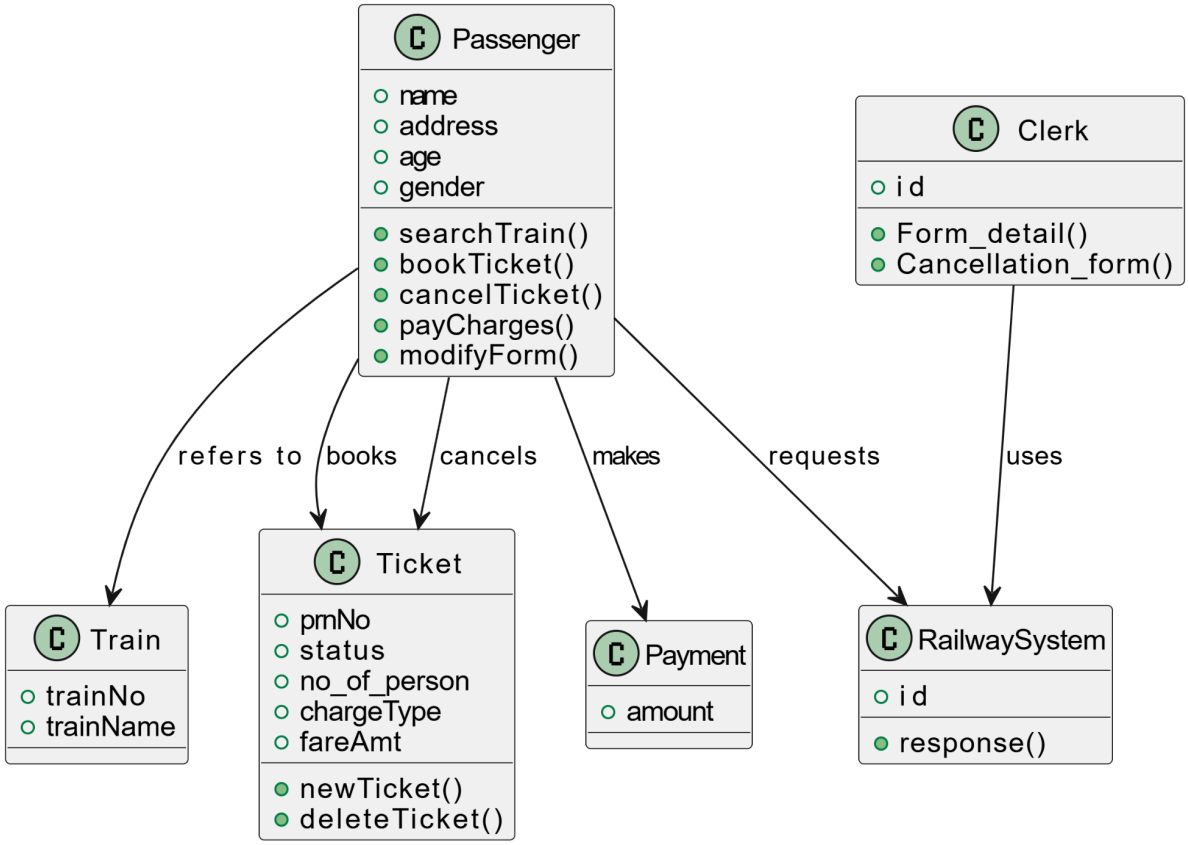
**3)Class diagram**

**Code:**

```
@startuml
class Train {
  +trainNo
  +trainName
}
class Passenger {
  +name
  +address
  +age
  +gender
  +searchTrain()
  +bookTicket()
  +cancelTicket()
  +payCharges()
  +modifyForm()
}
class Ticket {
  +prnNo
  +status
  +no_of_person
  +chargeType
  +fareAmt
  +newTicket()
  +deleteTicket()
}
class Clerk {
  +id
  +Form_detail()
  +Cancellation_form()
}

class Payment {
  +amount
}

class RailwaySystem {
  +id
  +response()
}
```

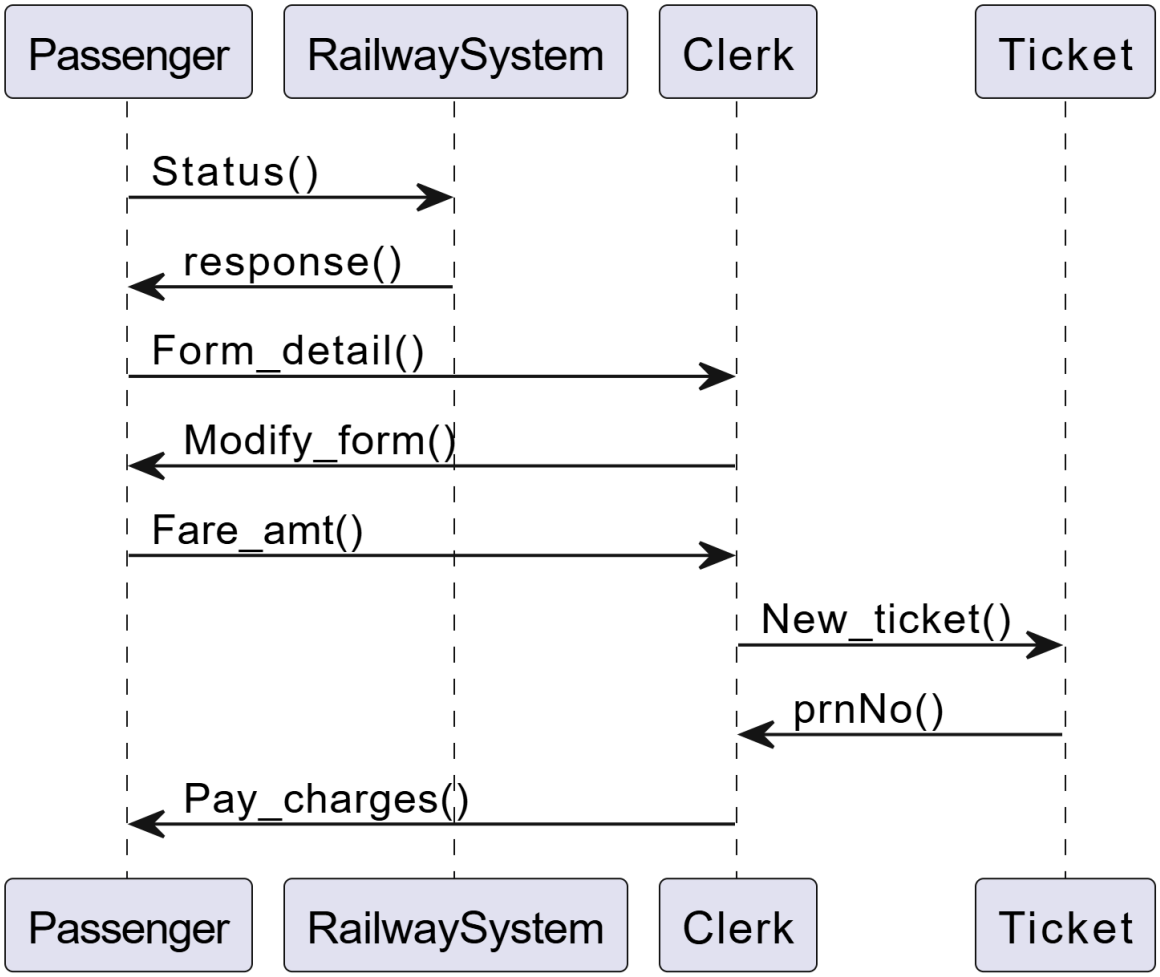
Passenger --> Ticket : books
Passenger --> Ticket : cancels
Passenger --> Payment : makes
Clerk --> RailwaySystem : uses
Passenger --> RailwaySystem : requests
Passenger --> Train : refers to
@enduml

**DIAGRAM:**




**4)SEQUENCE DIAGRAM:**
**CODE:**
@startuml
Passenger -> RailwaySystem : Status()
RailwaySystem -> Passenger : response()
Passenger -> Clerk : Form_detail()
Clerk -> Passenger : Modify_form()
Passenger -> Clerk : Fare_amt()
Clerk -> Ticket : New_ticket()

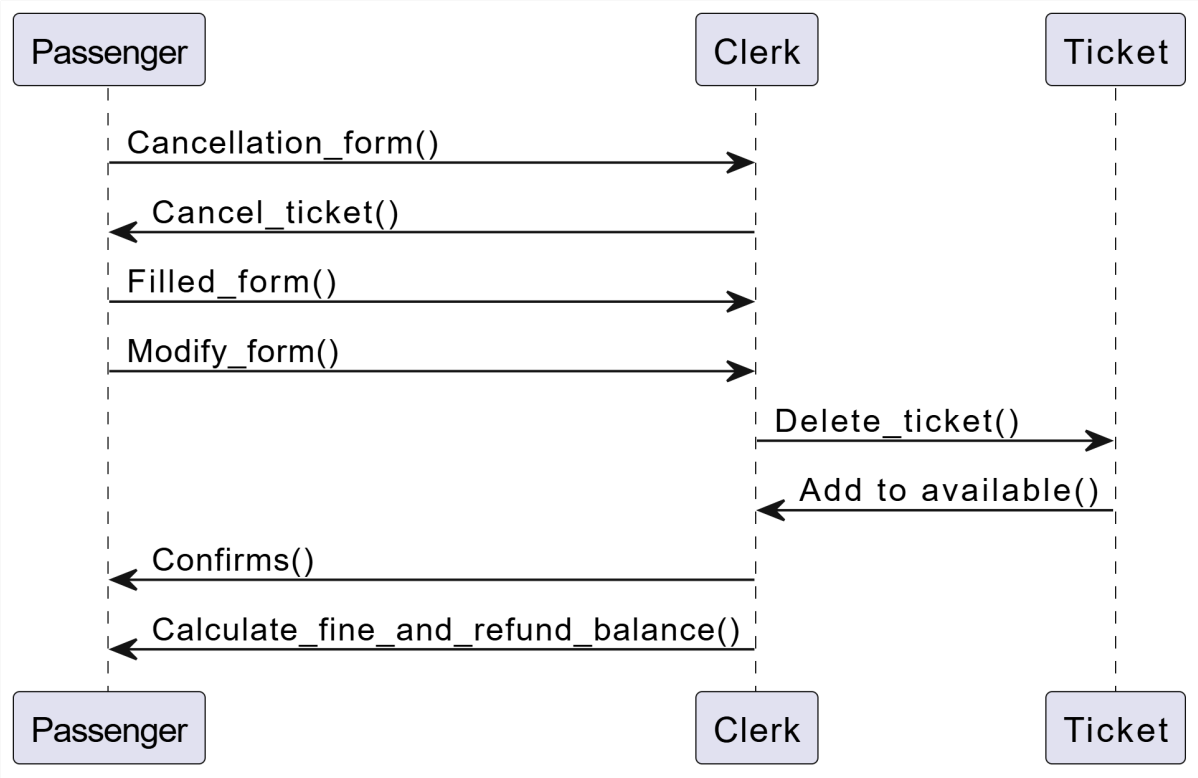
Ticket -> Clerk : prnNo()
Clerk -> Passenger : Pay_charges()
@enduml
**DIAGRAM**




**5)SEQUENCE DIAGRAM TO CANCEL TICKET**
**CODE:**
@startuml
Passenger -> Clerk : Cancellation_form()
Clerk -> Passenger : Cancel_ticket()
Passenger -> Clerk : Filled_form()
Passenger -> Clerk : Modify_form()
Clerk -> Ticket : Delete_ticket()
Ticket -> Clerk : Add to available()

Clerk -> Passenger : Confirms()
Clerk -> Passenger : Calculate_fine_and_refund_balance()
@enduml

**DIAGRAM**



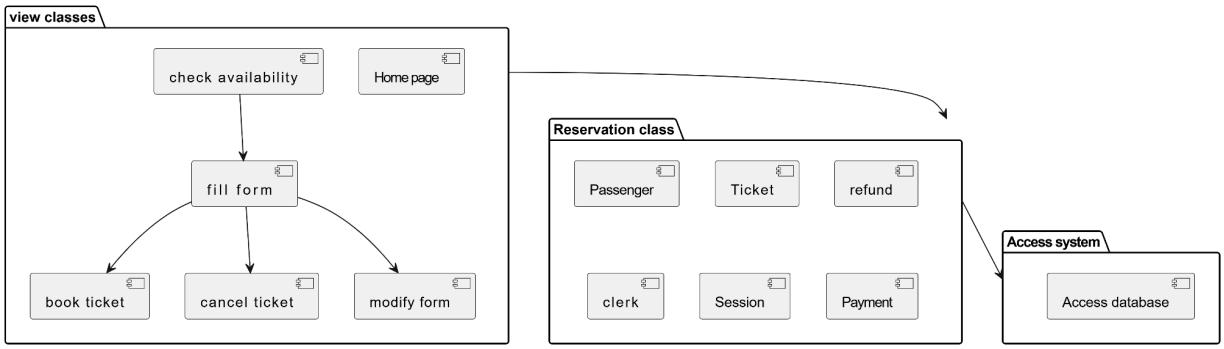

**6)Component Diagram**

**CODE:**

```
@startuml
package "view classes" {
  [Home page]
  [check availability] --> [fill form]
  [fill form] --> [book ticket]
  [fill form] --> [cancel ticket]
  [fill form] --> [modify form]
}

package "Reservation class" {
  [Passenger]
```

```
  [Ticket]
  [refund]
  [clerk]
  [Session]
  [Payment]
}
package "Access system" {
  [Access database]
}
[view classes] --> [Reservation class]
[Reservation class] --> [Access system]
@enduml
```

**DIAGRAM**



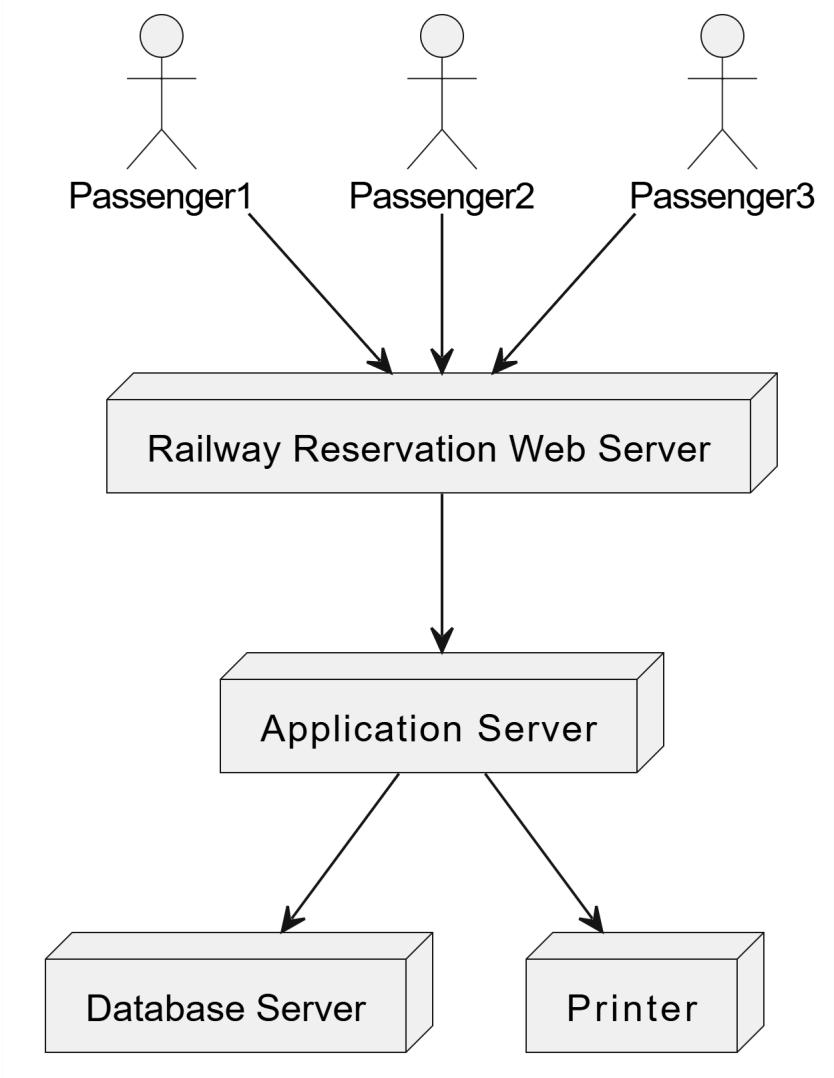

**7)DEPLOYMENT DIAGRAM FOR RAILWAY RESERVATION SYSTEM:**

**CODE:**

```
@startuml
node "Railway Reservation Web Server" {
}

node "Application Server" {
}

node "Database Server" {
}

node Printer {
}

actor Passenger1
```

```
actor Passenger2
actor Passenger3

Passenger1 --> "Railway Reservation Web Server"
Passenger2 --> "Railway Reservation Web Server"
Passenger3 --> "Railway Reservation Web Server"

"Railway Reservation Web Server" --> "Application Server"
"Application Server" --> "Database Server"
"Application Server" --> Printer
@enduml
```

**DIAGRAM**

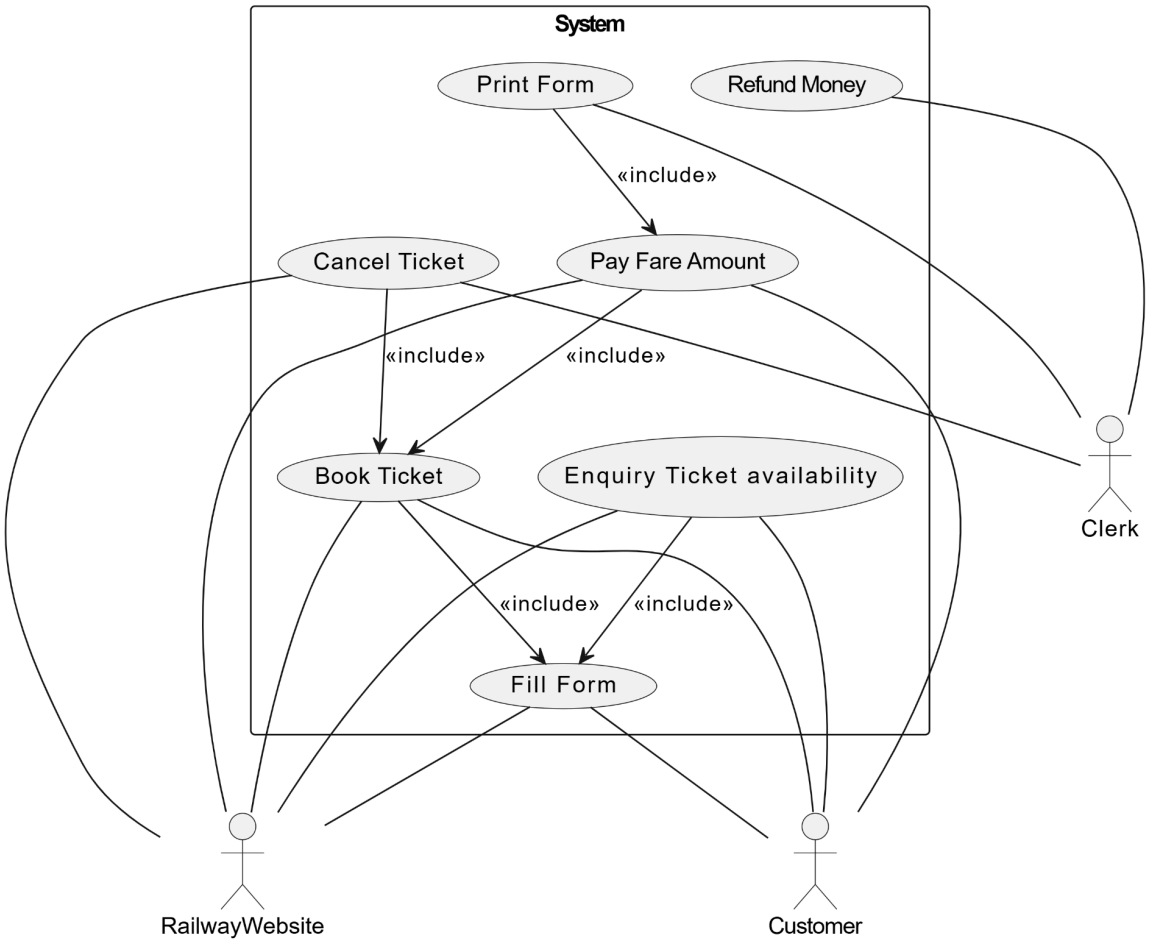


**8)USE CASE DIAGRAM**

**CODE:**

```
@startuml
actor Customer
actor Clerk
actor RailwayWebsite

rectangle System {
  (Enquiry Ticket availability) as Enquiry
  (Fill Form) as Fill
  (Book Ticket) as Book
  (Pay Fare Amount) as Pay
  (Print Form) as Print
  (Cancel Ticket) as Cancel
  (Refund Money) as Refund

  Enquiry -- Customer
  Fill -- Customer
  Book -- Customer
  Pay -- Customer
  Print -- Clerk
  Cancel -- Clerk
  Refund -- Clerk

  Enquiry -- RailwayWebsite
  Fill -- RailwayWebsite
  Book -- RailwayWebsite
  Cancel -- RailwayWebsite
  Pay -- RailwayWebsite

  Enquiry -down-> Fill : <<include>>
  Book -down-> Fill : <<include>>
  Pay -down-> Book : <<include>>
  Print -down-> Pay : <<include>>
  Cancel -down-> Book : <<include>>
```

**DIAGRAM:**


System
Print Form
Refund Money
«include»
Cancel Ticket
Pay Fare Amount
«include»
«include»
Book Ticket
Enquiry Ticket availability
Clerk
«include»
«include»
Fill Form
RailwayWebsite
Customer