```
Problem Statement: Design a Platform for Buying Tickets for Local Events

•    Behaviors and Key Objects:

-    Internet/Website:
    * Behavior: establishWebService, seatGeekResponse
    * Data: URL

-    Platform User:
    * Behavior: searchEvent, selectEvent, bookTickets, removeTickets, selectTicketType, selectTicketNumber, selectSeats
    * Data: User Name, User ID, User Password, User Email Address, User Phone, User Address, Number of Tickets, Seat Preference

-    Platform Login:
    * Behavior: logintoWebsite, createnewAccount, forgotPassword, validate(emailAddress, password)
    * Data: Email Address, Password, Phone Number

-    Ticket:
    * Behavior: isSoldOut, isAvailable, ticketType
    * Data: Ticket Event, Event Description, Ticket Price, Number of Total Tickets Available

-    Purchase:
    * Behavior: confirmPurchase, cancelPurchase
    * Data: Purchase ID, Purchase Date, Purchase Summary, Payment Method, Billing Address

-    Payment:
    * Behavior: makePayment
    * Data: Payment Method, Card Number

* Sequence of Flow – Invoke Objects with Behaviors:

User User
Platform SeatGeek


//Validate if the entered URL is valid
IF url.isValid = true
    boolean connection = seatGeek.establishWebService()
    //Authenticate the user if credentials are valid
    IF connection.isValid = true
        boolean authenticated = user.logintoWebsite(url, emailAddress, userPassword)
        seatGeekResponse response = seatGeek.validate(emailAddress, password)
        //SeatGeek validates the right user and allows the user to proceed further
        IF authenticated.isValid = true && seatGeek.validate = true
            //Website uses browser to request user location
            seatGeekResponse response = allowLocationDetection()
            userResponse response = user.selectLocationManually() || user.allowLocationDetection()
            user.searchItem(event)
                //Entered event is available for booking
                IF event.valid = true
                    //Ticket type can vary from Mobile to Physical tickets
                    user.selectEvent(event)
                    ticketType = user.selectTicketType(ticket)
                    IF ticket.isAvailable =  true
                        userResponse response = user.selectTicketNumber(n)
                            IF n<=count(ticket.isAvailable)
                                //User has to enter the seat preference for all tickets
                                Seat[] seats
                                //Seat count should not exceed the ticket count
                                WHILE count(seats)!= null && count(seats) !> n
                                userResponse user = selectSeats()
                                LOOP seat in seats
                                END
                                IF seats.isValid = true
                                    userResponse response = user.provideContactDetails()
                                    userResponse response = user.reviewSubTotal(totalSum)
                                    //User has to select the payment method; Card/Third Party
                                    userResponse response = user.selectPaymentMethod(Card) || user.selectPaymentMethod(thirdParty)
                                    user.makePayment(Card) || user.makePayment(thirdParty)
                                    seatGeekResponse response = SeatGeek.validatePayment()
                                    IF SeatGeek.validatePayment = true()
                                        seatGeekResponse response = confirmPurchase(purchaseSummary)
                                        seatGeekResponse response = sendTicketsToUser(ticketType)
                                    ELSE
                                        seatGeekResponse response = message.PaymentFailed(error)
                                        userResponse response = user.makePayment(Card || thirdParty) || userResponse response = user.cancelPurchase()
                                    END
                                ELSE
                                        seatGeekResponse response = message.invalidSeatSelection(error)
                                END
                            ELSE
                                //User can choose to edit the purchase : Remove certain number of tickets or cancel the selection
                                seatGeekResponse response = numberofticketsExceedsAvailable(error)
                                userResponse response = user.removeTickets(tickets)
                            END
                    ELSE
                        Boolean soldOut = SeatGeek.verify(ticket.isSoldOut)
                        if soldOut.isValid = true
                        seatGeekResponse response = print("Sorry the tickets have been sold out for this event")
                    END
                ELSE
                    //If the event is not found in search results
                    seatGeekResponse response = noSuchEventFound(error)
                END
            ELSE

            END
        ELSE
            //User has the ability to create new account, recover forgotten password
            seatGeekResponse response = user.createnewAccount || seatGeekResponse response = user.forgotPassword
        END
    ELSE

    END
ELSE

END
```