

### **Scenarios**

**Create Account**: <u>User</u> can become a <u>Registered User</u> by <u>creating an account</u> and entering their name, username, password, email and credit card number. Their information will be <u>saved</u> in the database.

**Login**: When <u>User</u> visits the movie app, they can <u>Login</u> if they have an existing <u>Registered User</u> account. Or continue as a Guest User.

**Search Movies**: User can view movies by browsing through a dropdown list.

**Select Movie**: User can <u>select</u> a movie from the movies dropdown list.

**Select Theatre**: <u>User</u> can <u>select</u> a <u>theater</u> from the theater dropdown list.

**Select Showtime**: User can <u>select</u> a showtime from the showtimes dropdown list.

**View Seats**: <u>User</u> can <u>view</u> available, selected and reserved <u>seats</u> graphically.

**Select Seats**: <u>User</u> can <u>enter</u> the row and column number for the desired available <u>seat</u>. The database will <u>update</u> accordingly to show the seat is selected.

**Make payment**: <u>User</u> can <u>proceed to checkout</u> and make <u>payment</u>. If it is <u>Registered User</u> there will be an option to <u>use stored banking information</u> from the database. If it is a <u>Guest User</u>, they will be required to enter their name, email and credit card number. The entered information will be <u>validated</u>. Payment will be <u>processed</u>.

**View Booking**: A <u>receipt</u> will pop up on the screen when the <u>payment</u> is processed. It will <u>display</u> details of the <u>payment</u>: Ticket ID, Movie Name, Theatre Name, Screening Room, and Seat.

**Cancel Reservation**: User can <u>cancel</u> a <u>reservation</u> up to 72 hours prior to the <u>showtime</u> by entering the ticket ID. A <u>Registered User</u> will receive 100% refund. A <u>Guest User</u> will receive an 85% voucher for a future purchase which is valid for 1 year. Database will be <u>updated</u> accordingly upon cancellation.

**Send Reservation/Cancellation Notification**: <u>User</u> will <u>receive</u> <u>emails</u> with their <u>tickets</u> and <u>receipts</u> for a <u>reservation</u>. For cancellations, <u>Registered users</u> will be <u>notified</u> of the 100% refund and Guest Users will be <u>emailed</u> a voucher.

Send Movie Announcement: Register Users will be emailed about new movie announcements.

**Add Movie**: Admin can add a new movie to the database.

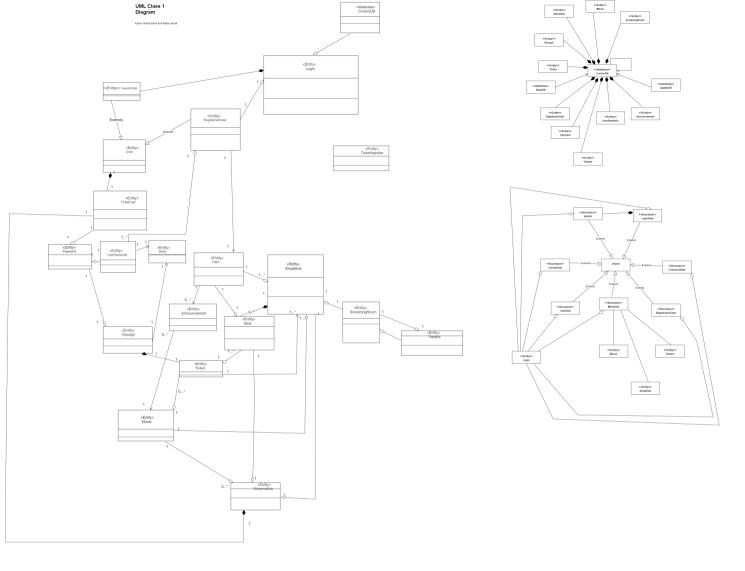
**Modify Movie**: Admin can change the details of a movie like the description.

**Remove Movie**: Admin can <u>remove</u> a <u>movie</u> from the database.

**Add Show**: Admin can add a showtime for a movie and update the database.

**Modify Show**: Admin can <u>modify</u> a showtime for a movie and update the database.

**Remove Show**: Admin can remove a showtime for a movie and update the database.



\*Entity\* Theatre

### Class Diagrams: Major Attributes and Operations for Entity Classes

«Entity» «Entity» «Entity» Date Movie Login - ID of movie:int - dataController:ControlDB - day:int - month:int - counter for movie:int = 200 - currentRegisteredUser:RegisteredUser - currentGuestUser:GuestUser year:int title of movie:String genre of movie:String + is before CurrentDate():boolean + questUser():void poster:String + login verification(username:String,password:String):RegisteredUser + toString():String price:double + logout user():void synopsis:String + user\_registration(username:String,password:String,f\_name:String,l\_name:String,email:String,name:String,cardNum:String):void - movieAnnouncement:Announcement + check if user is existing(username:String):boolean + toString():String «Entity» «Entity» «Entity» «Entity» «Entity» Theatre Bank GuestUser Payment Receipt - ID\_for\_payment:int - counter for theatre:int = 400 - name:String - ID of receipt:int - bankAccounts:ArrayList<UserBankInfo> - name of theatre:String - payment calculated:double - number of receipt:int = 600 - ID for theatre:int - card info:UserBankInfo - payment:Payment + account addition(ba:UserBankInfo):void - auditoriums list:ArrayList<ScreeningRoom> - date:Date - pCounter:int = 500 + CardInfo\_verification(name:String,num:String):boolean - ticket list:ArrayList<Ticket> + printReceipt():void + receiptToString():String «Entity» «Entity» «Entity» «Entity» «Entity» RegisteredUser ScreeningRoom Reservation Showtime TicketCart - username:String - theatre:Theatre - movie to book:Movie - ID for showtime:int pay:Payment - showing\_time:Showtime - Date of show:Date - password:String - auditoriumID:int totalCost:float - time in hours:int - seat\_to\_book:Seat - first name:String - seats available:int items:ArrayList<Reservation> - time in minutes:int - last name:String - number of rows:int - bookingID:int - audi:ScreeningRoom + addToCart(item:Reservation):void - gmail of user:String number of columns:int count bookings:int = 0 - number\_of\_rows:int + removeFromCart(item Reservation):void - auditoriumCounter:int = 300 - date of feeDeposit:Date - number of columns:int + calculatetotalCost():float - bankinfo:UserBankinfo - seats:Seat[][] - number\_of\_seats\_available:int + isFeePayed():boolean - movie:Movie - showtimeCounter:int = 700 + getSeatAvailability(r:int,c:int):Boolean + bookSeat(r:int,c:int):Boolean + vacantSeat(r:int,c:int):void + toString():String «Entity» «Entity» «Entity» «Entity» «Entity» UserBankInfo Announcement Ticket Seat User - ID for announcement:int Name of the customer:String - userCounter:int = 0 - seat\_select:Seat - selected\_row:int - counter for announcement:int=800 - ID for bank:int - ID of ticket:int - selected column:int - ID\_of\_User:int prAnnounceDate:Date - seat\_booked\_or\_not:boolean - counter for bank:int = 100 - movie:Movie - reg or unreg user:String puAnnounceDate:Date - payment card number:String - cart:TicketCart - showtime:Showtime - announce movie:Movie + showSeatSelection():void receipt:Receipt + bookSeat():void

+ vacateSeat():void

+ toString():String

Author: Farica Mago

- ticketCounter:int = 900

+ isPrivate():boolean

+ isPublic():boolean

### Class Diagrams: Major Attributes and Operations for Boundary Classes

#### <<Boundary>> MainView

- logoutButton:JLabel
- cartButton:JLabel
- annualFeeButton:JLabel
- cancelTicketButton:JLabel
- rowTextField:JTextField
- colTextField:JTextField
- selectRowLabel:JLabel
- selectColLabel:JLabel
- invalidSeatErrorLabel:JLabel
- takenSeatErrorLabel:JLabel
- selectedSeatErrorLabel:JLabel
- addedToCartLabel:JLabel
- seatGraphicLabel:JTextArea
- AddToCartButton:JLabel
- selectSeatLabel:JLabel
- selectShowtimeLabel:JLabel
- showtimeDetailsLabel:JTextArea
- showtimeSelectComboBox:JComboBox
- selectTheatreLabel:JLabel
- theatreSelectComboBox:JComboBox
- selectMovieLabel: JLabel
- movieDetailsLabel:JTextArea
- movieSynopsisLabel: JTextArea
- posterCard: JLabel
- movieSelectComboBox:JComboBox
- welcomeLabel:JLabel
- homeBackground:JLabel
- currentMovie:Movie
- currentTheatre:Theatre
- currentShowtime:Showtime
- +createSeatGraphic(frame:JFrame,backend:Login):void

# <<Boundary>> CheckoutView

- registerButton:JLabel
- titleLabel:JLabel
- emailTextField:JTextField
- emailLabel:JLabel
- payment\_card\_numberTextField:JTextField
- payment card numberLabel:JLabel
- userStoredBankingButton:JLabel
- nameTextField:JTextField
- nameLabel:JLabel
- backButton:JLabel
- + processPayment(frame:JFrame,backend:Login,costAmount:double,bankingInfo:UserBankInfo):void

# <<Boundary>> CartView

- backButton:JLabel
- checkoutButton :JLabel

# <<Boundary>> CancelView

- registerButton:JLabel
- ticketCodeTextField:JTextField
- ticketNumberLabel:JLabel
- backButton:JLabel

# <<Boundary>> LoginView

- titleLabel:JLabel
- infoLabel:JLabel
- usernameTextField:JTextField
- usernameLabel:JLabel
- passowrdField:JPasswordField
- passowrdLabel:JLabel
- submitLoginButton:JLabel
- registerButton:JLabel
- guestButton:JLabel

# <<Boundary>> RegistrationView

- fullNameTextField:JTextField
- fullNameLabel:JLabel
- usernameTextField:JTextField
- usernameLabel:JLabel
- passwordField:JPasswordField
- passwordLabel:JLabel
- emailTextField:JTextField
- emailLabel:JLabel
- cardNumberTextField:JTextField
- cardNumberLabel:JLabel
- registerButton:JLabel
- backButton:JLabel

# <<Boundary>> SetGUI

- MainFrame:JLabel
- loginPanel:LoginView
- + addSubmitLoginMouseClicked(e:MouseListener):void
- + displayGUI():void

### Class Diagrams: Major Attributes and Operations for Control Classes

«Control» Date

- frontEnd:SetGUI
- + main(args:String∏):main
- + runClient():void

### Class Diagrams: Major Attributes and Operations for Database Classes

Farica | December 5, 2022

#### ControlDB

- object:ControlDB
- list of users:ArrayList<RegisteredUser>
- list of announcements: ArrayList < Announcements >
- list of auditoriums: ArrayList < ScreeningRoom >
- list of pay:ArrayList<Payment>
- list of receipts:ArrayList<Receipt>
- list of movies:ArrayList<Movie>
- list of showtimes:ArrayList<Showtime>
- list of theatres: ArrayList < Theatre>
- list of banks:ArrayList<UserBankInfo>
- list of tickets:ArrayList<Ticket>
- inst:Bank
- + findMovie(title of movie:String):Movie
- + findTheatre(name:String):Theatre
- + findAllShowtime(searchMovie:Movie,searchTheatre:Theatre):ArrayList<Showtime>
- + findShowtime(searchMovie:Movie,searchTheatre:Theatre,date:Date):Showtime
- + findTheatresPlayingMovie(searchMovie:Movie):ArrayList<Theatre>
- + searchMovie(id:int):Movie
- + searchTheatre(id:int):Theatre
- + searchAuditorium(id:int):ScreeningRoom
- + searchBankingInfo(id:int):UserBankInfo
- + searchShowtimeInfo(id:int):Showtime
- + addMovie(movie:Movie):void
- + addShowtime(showtime:Showtime):void
- + addTheatre(theatre:Theatre):void
- + addTicket(ticket:Ticket):void
- + removeTicket(ticket:Ticket):void
- + addUser(user:RegisteredUser):void
- + addAnnoucement(announcement:Announcement):void
- + addAuditorium(auditorium:ScreeningRoom):void
- + addBankingInfo(b:UserBankInfo):void
- + addPayment(p:Payment):void
- + addReceipt(list of receipts:Receipt):void

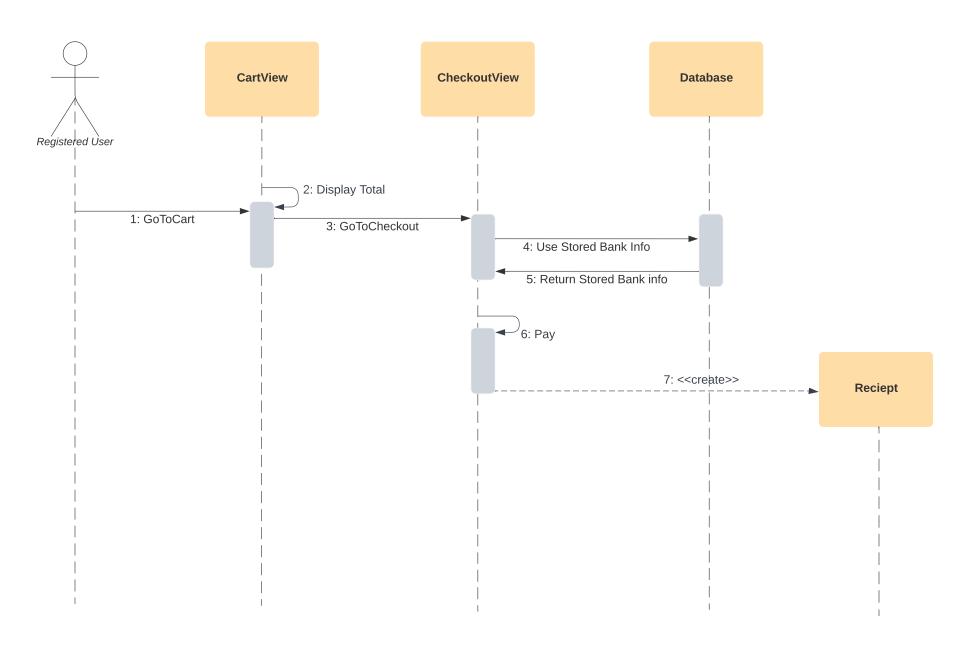
#### ReadDB

- ~ databse control:ControlDB
- + loadDatabase():void
- + gettingMovie():void
- + gettingTheatres():void
- + gettingAuditoriums():void
- + gettingShowtimes():void
- + gettingUsers():void
- + gettingAnnouncements():void
- + gettingBankInfo():void

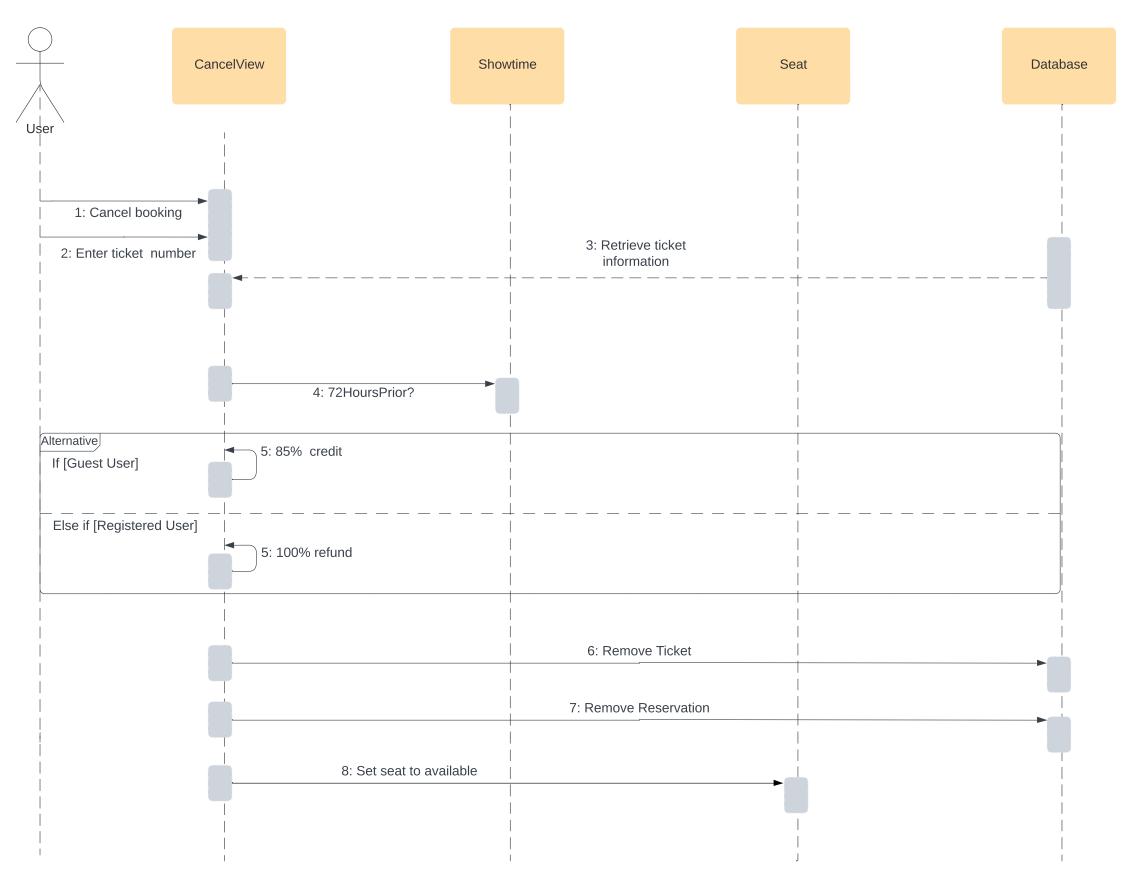
#### UpdateDB

- ~ databaseController:ControlDB
- + write\_to\_Database():void
- + writeRegisteredUser():void

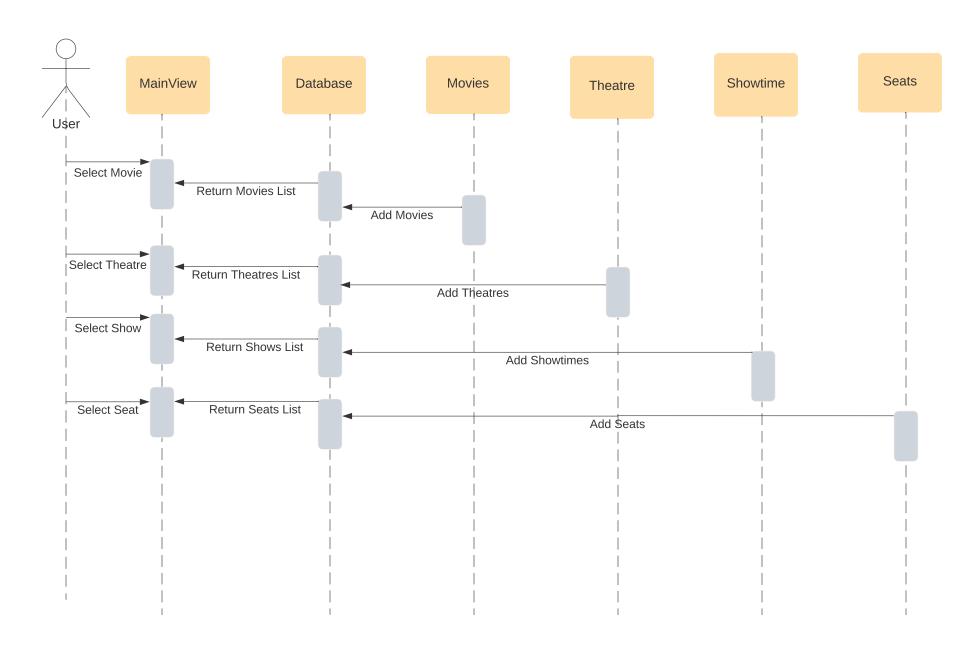
## **Registered User Makes a Payment**



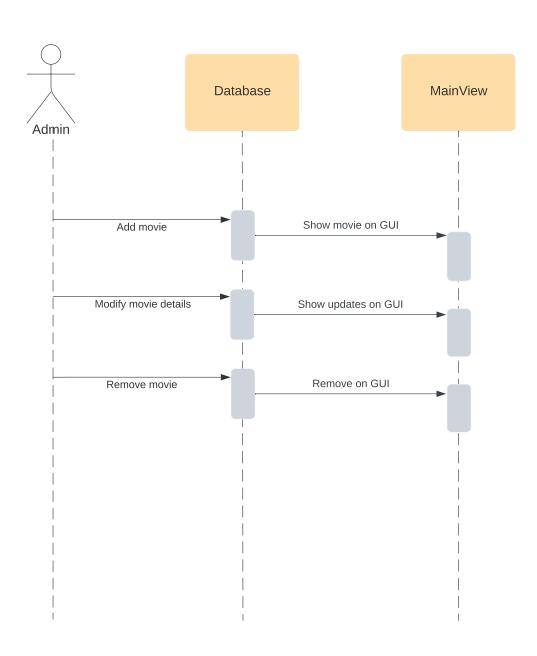
# User Cancels a Ticket



## User selects movie, theatre, showtime and seat

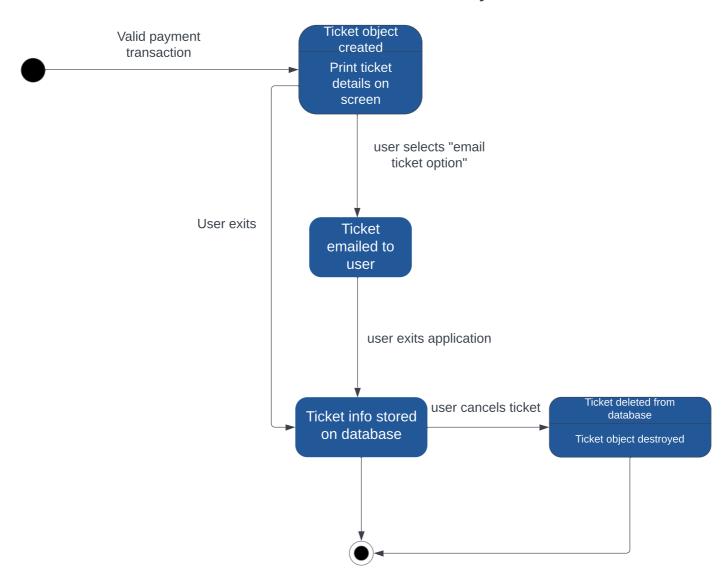


## Admin Adds , Modifies and Removes Movies

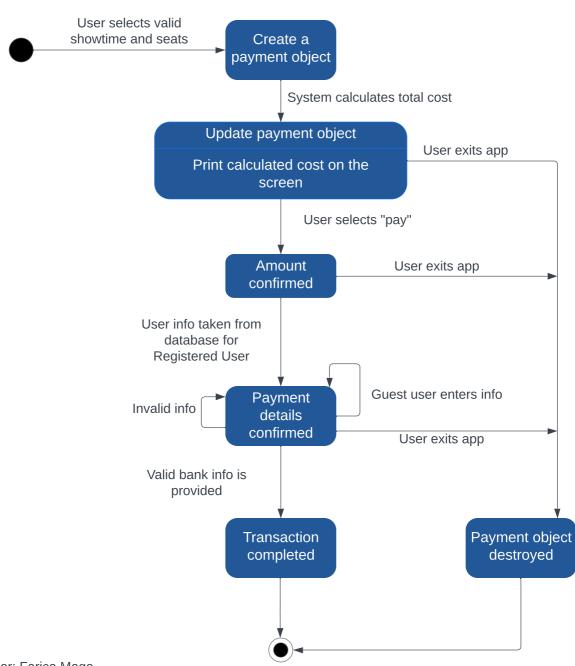


**Author: Hridika Banik** 

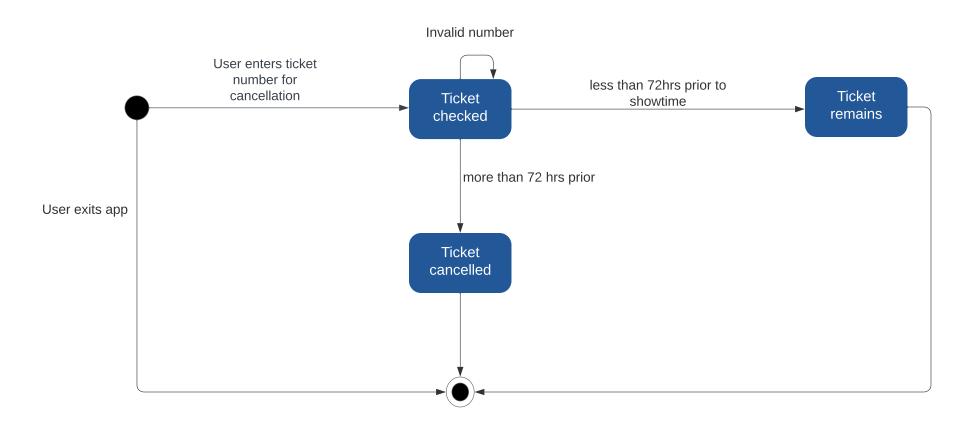
## **State Transition: Ticket Object**



## **State Transition: Payment Object**

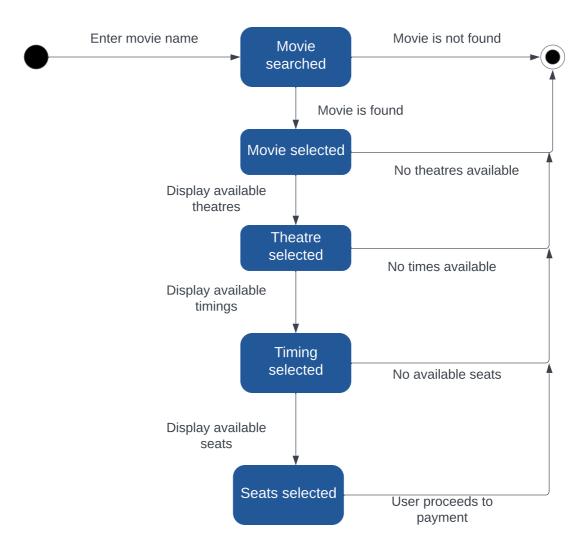


### **State Transition: Cancel Ticket**



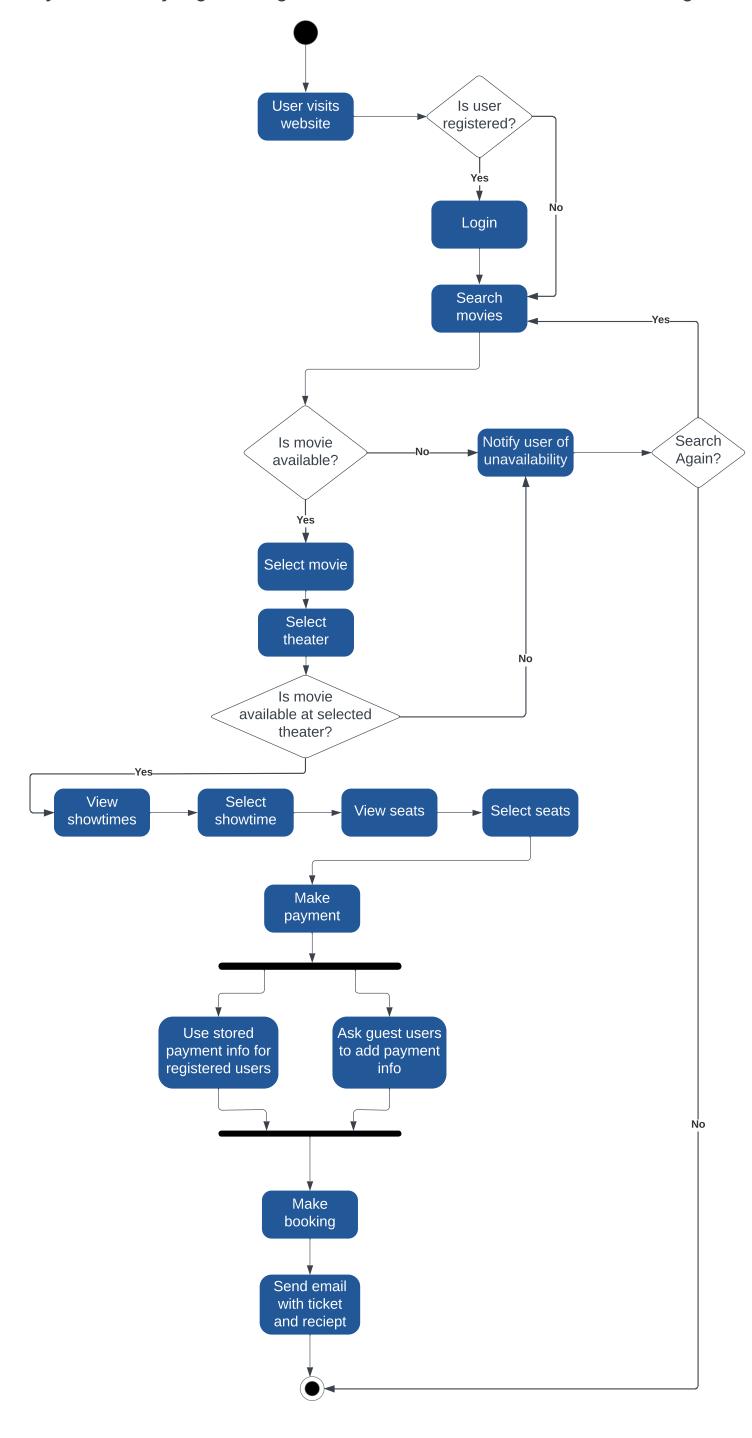
Author: Radia Jannat

## **State Transition: Select Movie, Theatre, Showtime**

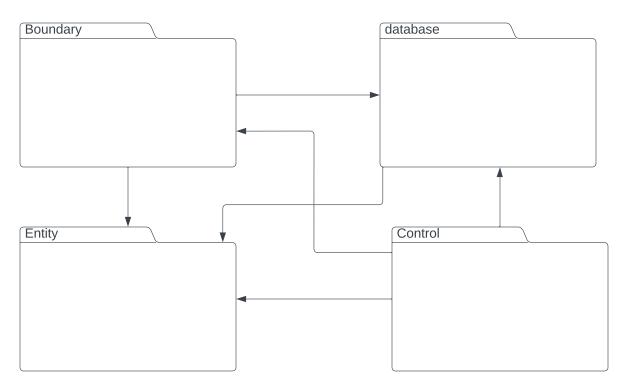


Author: Hridika Banik

System Activity Digram: Registered or Guest User makes a Movie Booking

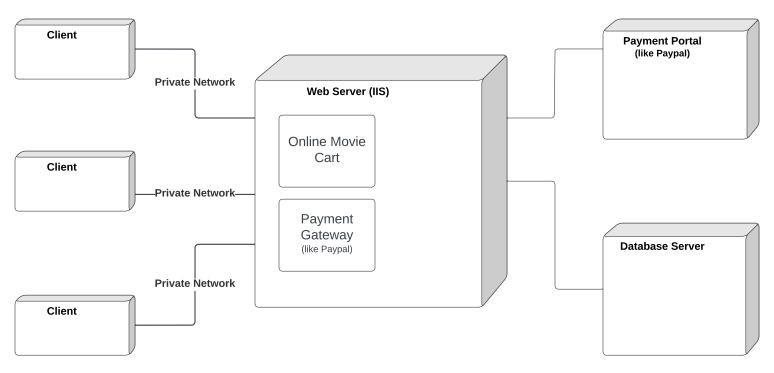


## Package Diagram



Author: Aarushi Roy Choudhury

## Deployemnt Diagram



Author: Aarushi Roychoudhury