

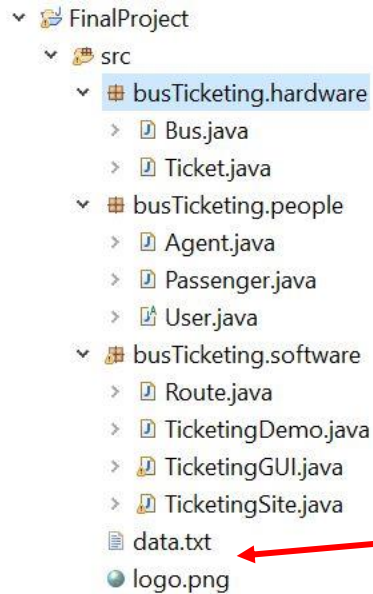
Greyhound Bus Ticketing System

For tutorial on how to run project and demonstration of functionality:

<https://www.youtube.com/watch?v=3wq-WJAndL0>

Tutorial: How to Open GUI

1. Open the project in Eclipse, ensure that the packages and corresponding classes are set up as shown below. Also make sure that the included files 'data.txt' and 'logo.png' are placed in the src folder.



2. To run the GUI right click the class 'TicketingDemo.java' located in the 'busTicketing.software' package and select 'run as Java Application'

Program Description

The Greyhound Bus Ticketing System is an online booking site with a primary goal of allowing customers to find and book bus tickets. The system provides a graphical interface that allows users to search for tickets and add them to their account. Users can also create a new account, as well as log into an old account to buy more tickets. The system contains a database of bus route information that reflects a bus's departure city, destination city, and date of travel. The Greyhound Bus Ticketing System allows its users to interact with this database in order to choose a bus route.

The main objects in this program include users, buses, tickets, and routes. Each user must have a name, email, password, and a credit card number. The system stores users based on their email and password, and users can log back into their account using these credentials. Each bus contains a unique bus number as well as a route. The bus number is unique to each bus and is what distinguishes each bus. Buses also have a manifest of passengers on board that is updated as passengers book tickets. A route contains a departure city and a destination city. Routes help organize where buses are scheduled to go in the system. Each route is further classified by a ticket. A ticket contains information such as the date of travel, the time of departure, and the time of arrival. Tickets are assigned to users after they have selected a route and purchased a ticket. Tickets help a user to keep track of their current bus schedule.

The system begins in the ticket search screen, where users are asked to provide their trip details including: departure city, destination city, and date of travel. The system performs error checking on these inputs to ensure that the user has entered valid information into the search box. If the user's information is valid, the system will display the bus routes that match the given information. The user will then be prompted to select a bus route. To do this, the user enters in the bus number corresponding to the ticket that they would like to purchase. Once again, error checking is performed on this bus number selection. The user is then prompted to create an account if they are not signed into the site. After this, the system displays the travel information corresponding to the selected ticket. If the user wants to confirm the purchase, they can click "confirm" and transfer the ticket to their account.

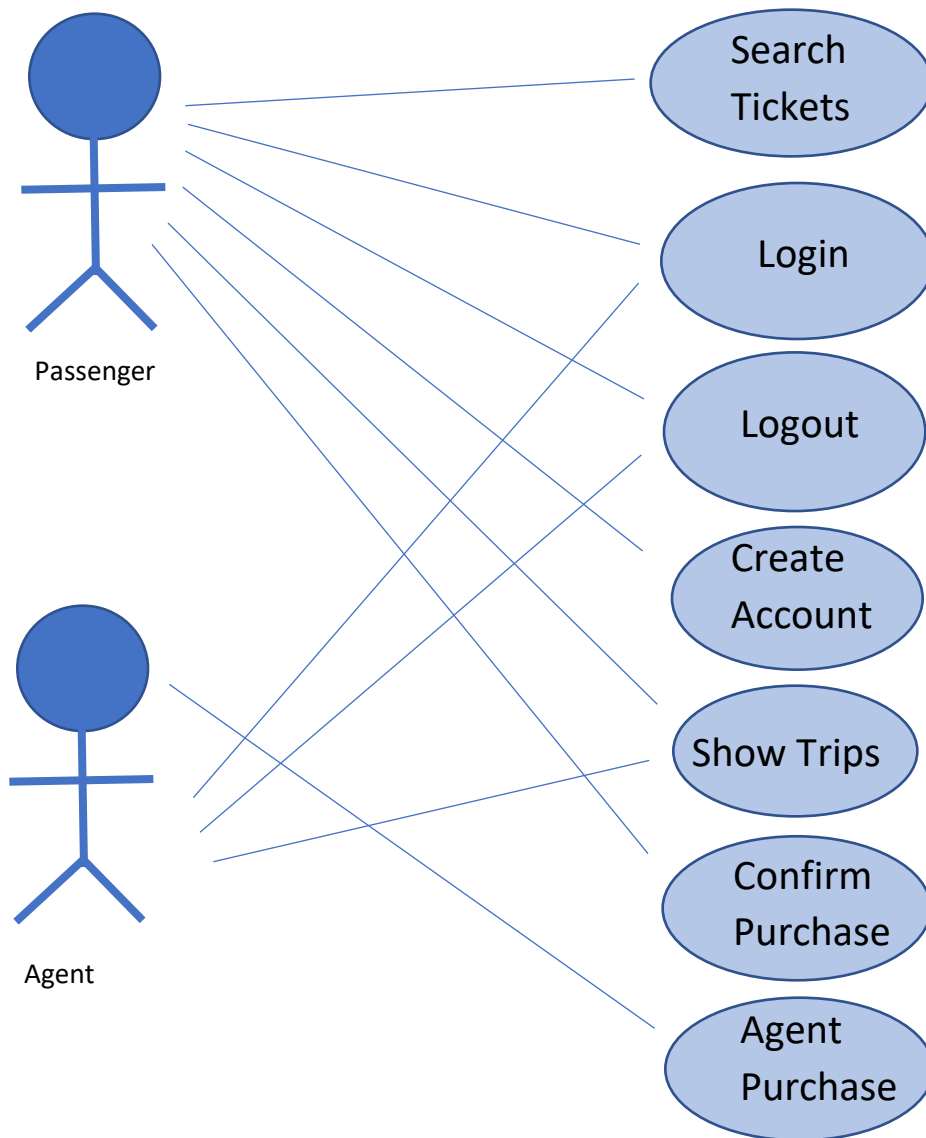
Also within the ticket search screen is a login menu that allows a returning user to login to their account. If a user enters a valid email and password, they will be successfully logged in. After a user has logged in they will be able to add additional tickets to their account by following the same steps described above. If the user has completed all of their purchases, they can logout of their account using the same menu.

Agents can also use the login menu to launch a special agent booking window. When agents enter the correct agent login for the site a new window will open that allows them to purchase a ticket for a customer. The agent must first create a passenger profile by entering the passenger's name, email, password and credit card number. Then the agent will be able to search for a bus ticket. Finally, the agent can add the ticket to the new passenger's account. The use of having this agent login is that it allows agents to purchase tickets for customers over the phone. It also gives the agents administrator access to the system to add tickets to customers.

Finally, within the search ticket window is also a trips menu tab. When this tab is selected it will display all of the trips that a logged in passenger is booked for. If no user is logged into the site, or the user does not have any booked trips, this screen will display that there are no booked trips for the customer. This screen will allow the user to see their current trips at any time they are browsing the system.

Overall, this system will help to streamline the bus ticket search and purchasing processes. Instead of having to call an agent, users will now be able to search and book tickets from their own computer. The approximate cost of this program will be based on the size of the database the customer wants to support. The more bus routes the customer wants to accommodate in the system, the larger the amount of hardware the system will need to handle the data.

Use Case Diagram



Use Case Descriptions

Use Case Name: Search Tickets
Actors: Passenger
Preconditions: GUI is in the initial search screen and is waiting for input by the user. At this point a user can either be logged in or not logged in to the system
Description: A user is prompted to enter the search criteria for their trip, including the departure city, the destination city, and the date. The system performs error checking to ensure that all inputs are valid. This error checking is described in detail in the error checking segment of the report. After this error checking is complete, and the system ensures that the user's inputs are valid, the system will display all possible trips that match the user's search criteria. At this point the user will be prompted to enter the bus number corresponding to the trip that they would like to purchase. Once again, the system will perform error checking to ensure that the user has entered a valid bus number into the GUI.
Exceptions: If the user enters an invalid departure city, invalid destination city, or an invalid date the system will throw an error and an error message will be displayed to the user. At this point the user will be prompted to resolve the error and continue their ticket search.
Postconditions: System now waits for user to either create an account if they are not logged in. Or, the system waits for the user to confirm their purchase if they already have an account

Use Case Name: Login
Actors: Passenger, Agent
Preconditions: The GUI is in the initial ticket search screen and is waiting for the user to either input search criteria into the search box or select a menu option.
Description: If a user selects the menu option of signing in, a pop-up will appear that requests that the user enters their email and password. After the user has entered this information in the GUI, the system checks to see if they have entered a valid profile. If a valid profile has not been entered, an error message will be displayed explaining this to the user and they will be prompted to try again. If the user has entered in a correct email and passcode a success message will be displayed to the user and the user can continue using the system as a logged-in customer.
Exceptions: If the user enters an invalid email or password that does not match one already found in the system, and error message will be displayed and they will be prompted to enter a new email or password.
Postconditions: System waits for user to enter in search criteria into the search box so that the system can display available bus tickets.

Use Case Name: Logout
Actors: Passenger, Agent
Preconditions: A user has already logged into the site by following the steps described in the use-case description above.
Description: If a user selects to logout of the site a message will be displayed to them that a successful logout has occurred. Additionally, the system will update its internal variables to reflect that the current user of the system is not logged in.
Exceptions: None
Postconditions: System waits for either a new user to log into the program, or a user to enter search criteria into the search box to begin a bus search.

Use Case Name: Create Account
Actors: Passenger
Preconditions: The create account screen is launched if a user has selected a ticket they would like to purchase, but are not logged in to the system. The program requires that all customers who purchase tickets log into the system first, therefore a first-time user will be prompted to create an account on the site.
Description: A pop-up window displays that prompts the user to enter information, including the user's first name, the user's email, a password, and a credit card number. All of this information will be stored on the site and will be used to help identify the current user if they would like to purchase another ticket on their account. A button will be displayed on the bottom of the window reading "create account". Once the user presses this button their information will be saved in the site, and an account will be created for them.
Exceptions: A user must enter a 16 digit credit card, because only 16 digit numbers are valid. If the user does not enter a valid credit card number they will be prompted to enter another number until this condition is satisfied.
Postconditions: After a first-time user has created an account they will be allowed to continue to the final step of purchasing their bus ticket, which is confirming their order details and clicking the purchase button.

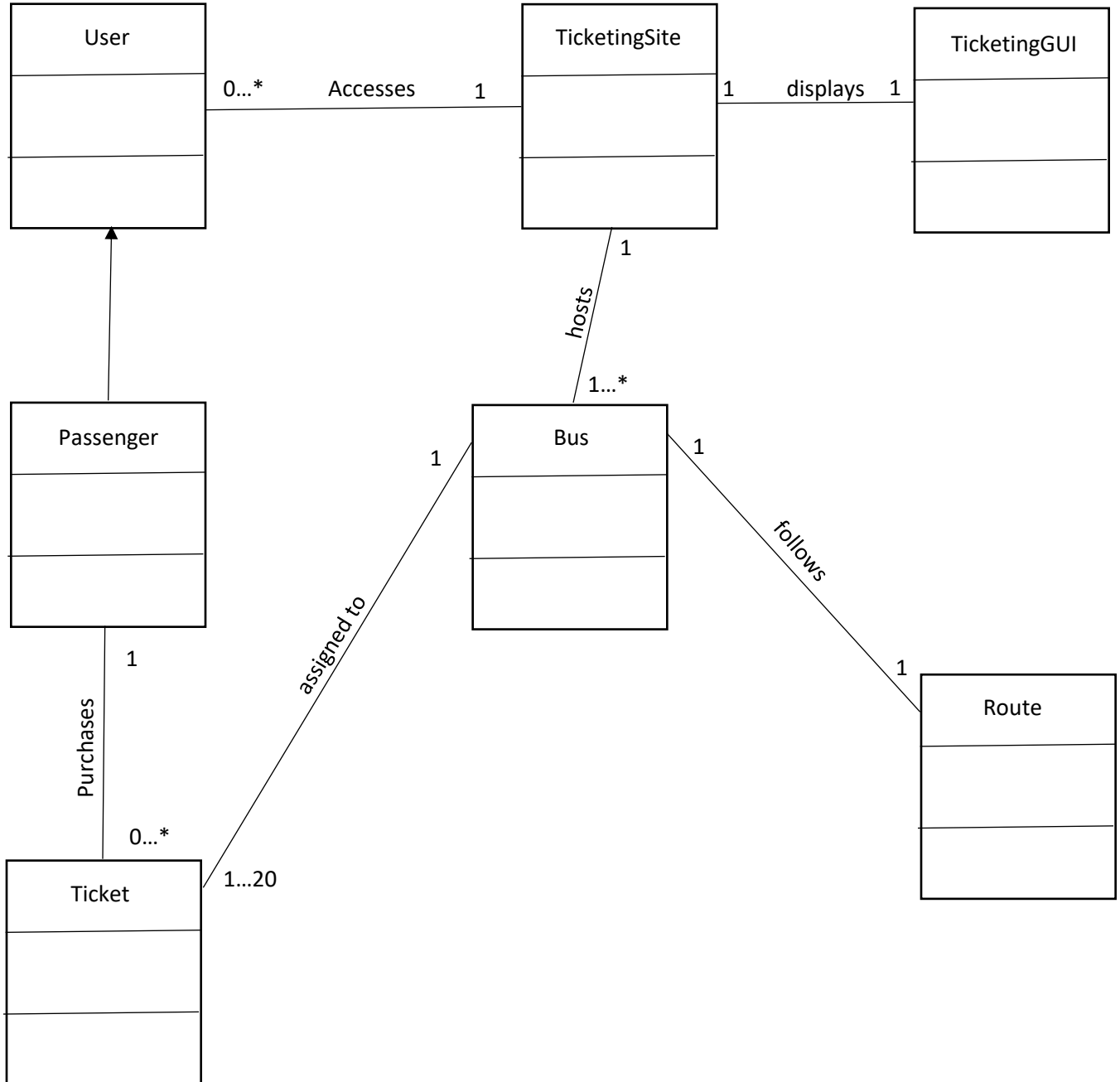
Use Case Name: Show Trips
Actors: Passenger, Agent
Preconditions: GUI is in the initial ticket search window and is waiting for the user to either select a menu option or search tickets.
Description: If the show trips menu option is selected a new pop up window will be displayed. If a passenger has selected this option the window will display all of the trips that the passenger is currently booked for. The window will show trip details such as departure city, destination city, arrival time, departure time, bus number, and date. If an agent selects this option the pop-up will display a master list of all tickets of all customers that have used the site. The pop-up will display the entirety of the system's trip data
Exceptions: If the system currently does not have any trips saved the window will display that there are no trips to show
Postconditions: The user is able to close the pop-up. After this the system will continue to wait in the initial screen. The system will wait for the user to either login, logout, or enter search criteria and begin a ticket search.

Use Case Name: Confirm Purchase
Actors: Passenger
Preconditions: A passenger has selected a ticket he/she would like to purchase and has either created an account or is signed into the site.
Description: A new window is displayed that shows the trip details for a passenger that has selected to purchase a given ticket. These trip details include departure and destination city, as well as the date of travel and the bus number. The user will then review this information and decide if they would like to go through with purchasing the ticket. If the user decides to purchase the ticket they will select the 'purchase' button and a message will be displayed stating that their purchase of the ticket was successful. If the user decides they would not like to purchase the ticket they can exit the window and return to the home screen.
Exceptions: None
Postconditions: After a user has successfully purchased a ticket, or if the user has decided to cancel the purchase, the system will return to its initial ticket search screen where it will wait for the user's next action.

Use Case Name: Agent Purchase
Actors: Agent
Preconditions: A user has successfully logged into the system as an agent, and the system has verified their status as an agent.
Description: If an agent is using the site a new pop-up window will appear with features that are only available to site agents. This pop-up will give the agent the opportunity to purchase a new ticket on a customer's behalf. The pop-up will prompt the agent to enter in a customer's name, email, password, and credit card number. At this point the system will create a new profile for this user and add their information to the system's database. Next the agent will be prompted to search for a ticket to add to this new customer's profile. The agent will be asked to enter in a departure city, a destination city, and a date. Finally, the agent will add this new ticket to the customer's newly created profile on the site.
Exceptions: Same exceptions exist as in the search ticket use case above
Postconditions: After an agent has added a ticket to the customer's profile they can either log out or add another ticket

Domain Class Model

Note: Class UMLs are shown in detail on following pages



User
-name: String -email: String -password: String
+User(name: String, email: String, password: String): void +getName(): String +getEmail(): String +getPassword(): String +setName(name: String): void +setEmail(email: String): void +setPassword(pass: String): void

Passenger
-creditCard1: int -creditCard2: int -tickets: Ticket[0...*]
+Passenger(name: String, email: String, cc: String, password: String): void +getTickets(): Ticket[0...*] +getCreditCard(): String +setCreditCard(cc: String): void +buyTicket(ticket: Ticket): void +printTickets(): void +lastFourDigits(): void

Bus
-busNumber: int -capacity: int -count: int -passengers: User[0...*]
+getBusNumber(): int +getCapacity(): int +getPassengerList(): User[0...*] +setBusNumber(busNumber: int): void +setCapacity(capacity: int): void +addPassenger(u: User): void +printPassengerList(): void

Ticket
-depart: String -dest: String -departTime: int -arriveTime: int -date: int -bus: Bus -price: int
+Ticket(depart: int, arrive: int, date: int, departure: String, dest: String, price: int): void +getDepartTime(): int +getArriveTime(): int +getDate(): int +getBus(): Bus +getDestination(): String +getDeparture(): String +printTripData(): void +dateToString(): String +timeToString(): String

Route
-destination: String -departure: String -schedule: Ticket[0...*]
+Route(depart: String, dest: String): void +getDestination(): String +getDeparture(): String +getSchedule(): Ticket[0...*] +setDestination(destination: String): void +setDeparture(departure: String): void +addTrip(a: Ticket): void +printRoute(): void

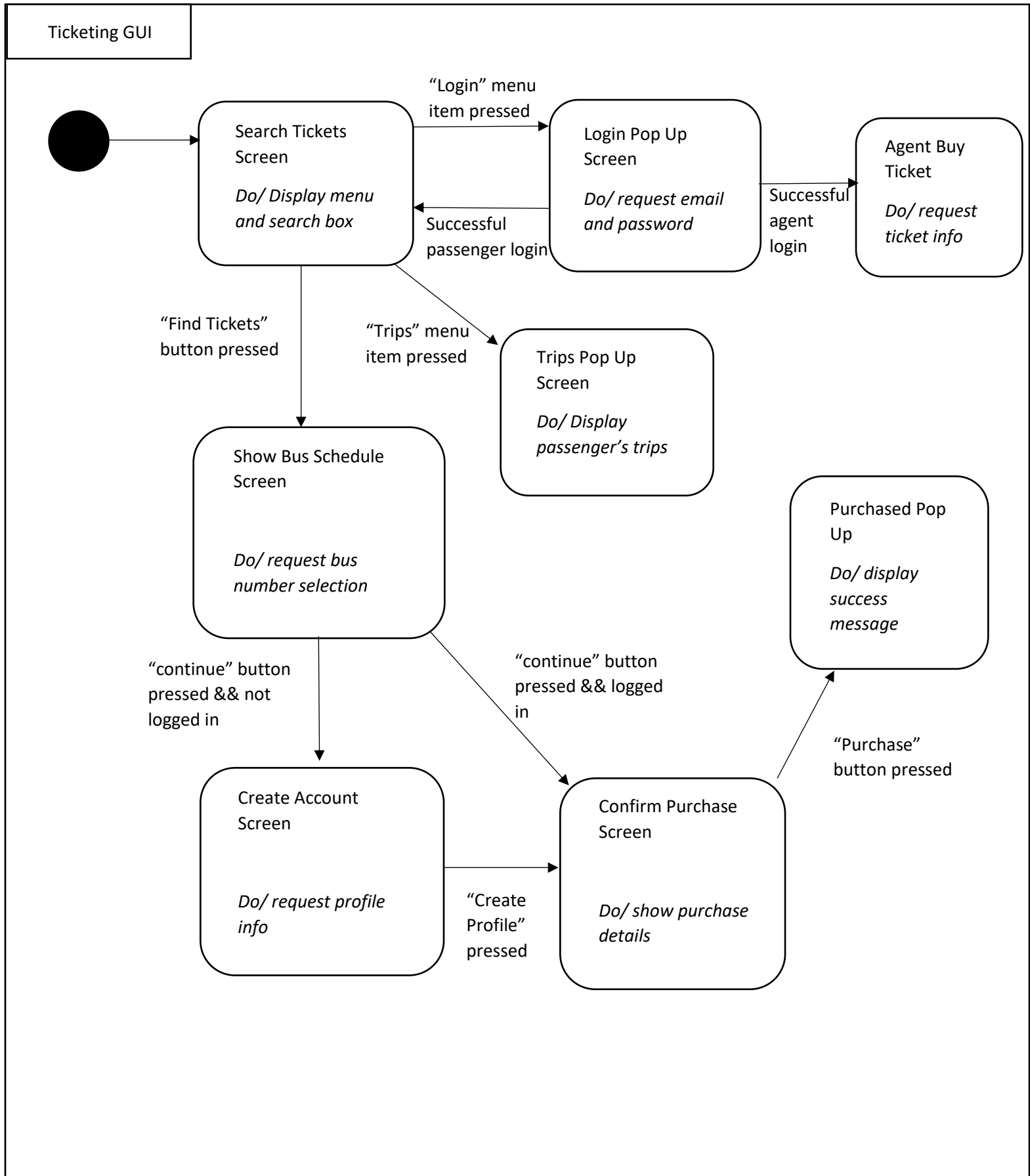
TicketingSite
-routes: Route[0...*] -potentialBus: Integer[0...*] -passengers: Passenger[0...*] -flag: boolean
+addData(): void +showTickets(depart: String, dest: String, date: String): Boolean +dateToInt(date: String): int +findTicket(busNumber: int): Ticket +addPassenger(p: Passenger): void +signIn(email: String, password: String): boolean +findPassenger(email: String, password: String): Passenger +getFlag(): Boolean +checkDepart(depart: String): Boolean +checkDest(dest: String): Boolean +checkDate(date: String): Boolean +checkDepartAndDest(depart: String, dest: String): Boolean +checkBus(busNum: String): Boolean +getPassengers(): Passenger[0...*]

TicketingGUI
-searchTickets: JButton -enterBusNumber: JButton -createAccount: JButton -confirmPurchase: JButton -menu: JMenuBar -login: JMenuItem -logout: JMenuItem -demo: TicketingSite -departCity: JTextField -destCity: JTextField -date: JTextField -busNumber: JTextField -name: JTextField -email: JTextField -password: JTextField -cc: JTextField -aDeaprtCity: JTextField -aDestCity: JTextField -aDate: JTextField -cDepartCity: JTextField -cDestCity: JTextField -cDate: JTextField

-newSchedule: JFrame
-newAccount: JFrame
-purchase: JFrame
-agentSearch: JFrame
-main: JFrame
-isLoggedIn: boolean
-isAgent: Boolean
-runningAgent: boolean
-p: Passenger
-t: Ticket

+TicketingGUI(windowName: String, demo: TicketingSite): void
+ButtonListener(): void
+ShowSchedule(): void
+BusNumButtonListener(): void
+CreateAccount(): void
+CreateAccountListener(): void
+purchaseTicket(): void
+purchaseListener(): void
+MenuListener(): void
+agentTicket(): void
+agentAccountListener(): void

State Diagram



State Descriptions

State Name: Search Tickets Screen
State Description: The search tickets screen is the main home screen of the bus ticketing GUI. On this screen the user has the option to either login/logout or display their trips. In addition, the user is prompted to enter in their search criteria information to begin a ticket search. This information includes a departure city, a destination city, as well as a date of departure.
Event Sequence That Produces State: GUI is started
Condition That Characterizes State: system waits for user to either select a menu item or enter search criteria
Events Accepted In This State: Login menu item pressed => system proceeds to login pop up window Logout menu item pressed => system automatically logs out the current user of the ticketing program Trips menu item pressed => system displays customer's trips 'Find Tickets' button pressed => system proceeds to a new screen which displays the results of user's search

State Name: Login Pop Up Screen
State Description: The login pop up screen is a pop up message that is displayed after a user has chosen to log into the system. The user is prompted to enter their email and password into the window. The system then automatically will verify if the user has entered a valid login.
Event Sequence That Produces State: User selects login menu item while in the ticket search screen.
Condition That Characterizes State: System waits for user to enter in a valid email and password into the pop up
Events Accepted In This State: Valid Login entered => user is returned to the ticket search screen Invalid Login entered => user is prompted to try again

State Name: Trips Pop Up Screen
State Description: This pop up screen is displayed when a user selects the trips menu item when in the ticket search screen. This pop up is used to display a passenger's list of trips on their account. In addition, if an agent is logged in this pop-up will show all trips contained in the system.
Event Sequence That Produces State: User selects trips menu item while in the ticket search screen
Condition That Characterizes State: System waits for user to review the information and then close window
Events Accepted In This State: User closes window => user is returned to the main ticketing screen

State Name: Agent Buy Ticket
State Description: This screen is the form that is used by an agent that is purchasing a ticket on behalf of a customer. Our program allows a logged in agent to purchase a ticket for a customer by entering in the customer's information such as their name, credit card number, email, and password. The agent can then search for bus routes and add a ticket to a passenger's account.
Event Sequence That Produces State: While in the login menu, the user enters a valid agent login. The system automatically detects if a valid agent login has been entered and will direct to this screen.
Condition That Characterizes State: System waits for agent to enter in a passenger's details and search for a ticket
Events Accepted In This State: Agent completes transaction => user is returned to main ticketing screen

State Name: Show Bus Schedule Screen
State Description: This screen shows all bus routes that match a customer's entered criteria. This screen will list all important trip details for each bus route, including the bus number, departure and arrival time, departure and arrival cities, and the date. The system will then request that the user enters the bus number of the bus ticket that they would like to purchase. If the system detects that a valid bus number has in fact been entered, the system will automatically direct the customer to the next step in the bus ticket purchasing process in the site.
Event Sequence That Produces State: User enters search criteria into the ticket search screen and selects find tickets button. This causes the system to generate a list of bus routes that match this search criteria. The system then proceeds to show these bus routes in a new window.
Condition That Characterizes State: System waits for user to review options and enter a bus number.
Events Accepted In This State: Bus Number entered and logged in => go to confirm purchase screen. Bus Number entered and not logged in => go to create account screen

State Name: Create Account Screen
State Description: This screen is shown to a first time user of the ticketing site. Our program requires that all customers have an account before purchasing a ticket. Therefore this site requests information from the user such as the user's name, email, password, and credit card number. Error checking is performed on this information to ensure that it is accurate. After this error checking is complete the system will automatically create a user profile and add it to the site's database
Event Sequence That Produces State: User has selected a ticket to purchase but is not logged in
Condition That Characterizes State: System waits for user to enter in accurate information into the form and then creates account
Events Accepted In This State: Create Account button pressed => profile is created for the user and system moves forward to confirm purchase screen

State Name: Confirm Purchase Screen
State Description: This screen is used to show the final confirmation details to the user before he/she decides to purchase the ticket. These details include the departure and destination cities, the departure and arrival times, the date, the bus number, and the price. This is the users last chance to cancel the purchase. Once the purchase button is pressed the system will automatically add the ticket to the passenger's account, as well as add the passenger to the bus's manifest.
Event Sequence That Produces State: A non-logged in user has created an account and is ready to purchase a ticket or a returning user has selected a ticket and is about to purchase the ticket.
Condition That Characterizes State: The system waits for the user to press the 'confirm' button so that the system can process the purchase
Events Accepted In This State: 'Confirm' button is pressed => system processes purchase and success message is shown Screen is closed => user has cancelled purchase, purchase does not go through the system and is not processed

State Name: Purchased Pop Up
State Description: This is a pop up screen that notifies a user that their purchase was successful as well as giving the user one final reminder of their trip details included in their ticket
Event Sequence That Produces State: User has just purchased a ticket by clicking the purchase button on the previous screen
Condition That Characterizes State: System waits for user to review information and then close the screen
Events Accepted In This State: Screen is closed => user is returned to the main ticket search screen

Milestone Review Fixes

As a result of the feedback we obtained from our milestone review, our group made several revisions to our project. These revisions are listed below

1. Refined overall class model to better reflect OO programming

One of the critiques we received was to better use OO programming principles by adding additional methods to our classes, and further dividing these methods among our classes. Our group implemented this change by revising our domain class model, adding additional methods to these classes, and refining our overall program design

2. Updated final report to reflect project functionality more accurately

One of the critiques we received was that our report did not accurately reflect our project. To fix this, our group revised our entire final report. We went through our project description and added comments about additional functionality that we had added. We revised our domain class model to reflect our updated classes and methods. We created a new state diagram and state descriptions to match the user experience in our GUI. Now our final report more accurately reflects our project and its use cases.

3. Adding in additional error checking to streamline user experience

One of the critiques we received was that there were several spots in our code where the user could enter an erroneous value, causing the program to crash when it tried to use this value. To address these concerns we added in additional error checking to our final project. Now our system will check and verify the validity of user inputs throughout our project. For example, in the main search tickets screen error checking is done to ensure that a user has entered a valid departure city, destination city, and date. If any of these fields are incorrect an error message is displayed to the user and the program is prevented from moving forward until the error has been resolved. This has helped to remove many of the bugs that were present in the initial project.

4. Adding additional functionality to increase the overall complexity of our project

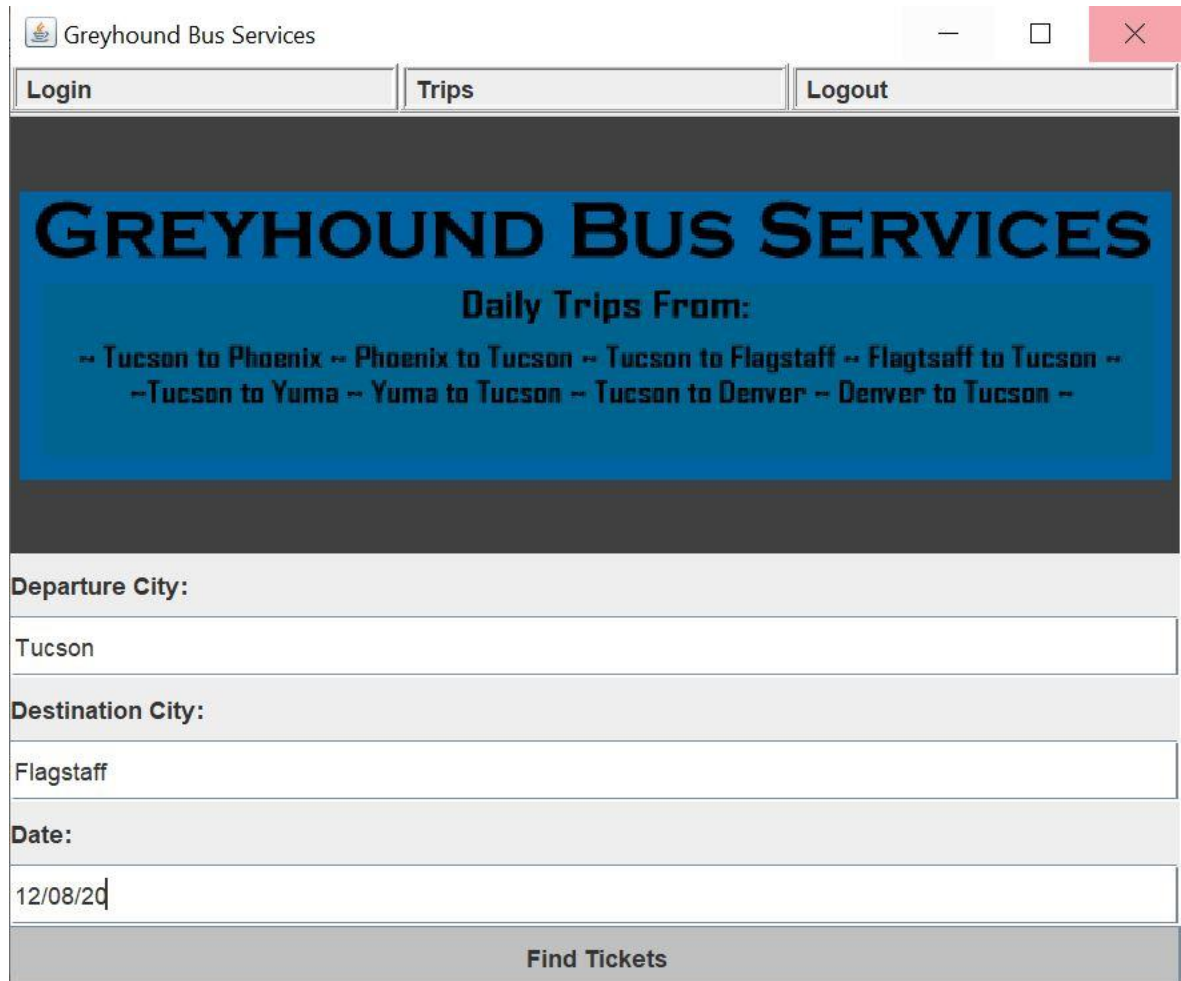
A final critique that we received was that our project was not complex enough to meet the project guidelines. To fix this issue our group added additional functionality to our program to increase the overall complexity of the project. One major example of this was adding in an agent feature to the system. Now our system has the ability to accept an agent login, which launches a new agent search screen. This took a significant amount of time for us to implement in the GUI, and it certainly increases the overall difficulty of our program.

GUI Testcases

The following section contains examples of testcases that can be replicated in the GUI.

New Customer Purchases Ticket


1. User is directed to the main ticketing search screen after starting the GUI. The user is then prompted to enter their trip details including the departure city, destination city, and date of travel.



The screenshot displays the Greyhound Bus Services web application. At the top, there is a navigation bar with a logo, the text "Greyhound Bus Services", and three buttons: "Login", "Trips", and "Logout". Below the navigation bar is a large blue banner with the text "GREYHOUND BUS SERVICES" and "Daily Trips From:". The banner lists several routes: "Tucson to Phoenix", "Phoenix to Tucson", "Tucson to Flagstaff", "Flagstaff to Tucson", "Tucson to Yuma", "Yuma to Tucson", "Tucson to Denver", and "Denver to Tucson". Below the banner, there are three input fields for "Departure City:", "Destination City:", and "Date:". The "Departure City" field contains "Tucson", the "Destination City" field contains "Flagstaff", and the "Date" field contains "12/08/20". At the bottom of the form is a "Find Tickets" button.

Greyhound Bus Services	Login	Trips	Logout
GREYHOUND BUS SERVICES Daily Trips From: ~ Tucson to Phoenix ~ Phoenix to Tucson ~ Tucson to Flagstaff ~ Flagstaff to Tucson ~ ~ Tucson to Yuma ~ Yuma to Tucson ~ Tucson to Denver ~ Denver to Tucson ~			
Departure City:			
Tucson			
Destination City:			
Flagstaff			
Date:			
12/08/20			
Find Tickets			

2. After user hits 'Find Tickets' all matching routes for the given search criteria are displayed in a new window. The user is prompted to enter the bus number of the ticket that they would like to purchase.

 Schedule for Tucson to Flagstaff on 12/08/20—□×

Date: 12/08/20

Price: \$115

Bus Number: 53

Departure Time: 10:30 am

Arrival Time: 2:30 pm

Date: 12/08/20

Price: \$100

Bus Number: 54

Departure Time: 12:25 pm


Arrival Time: 4:25 pm

Enter Bus Number:

53

Continue

3. After the user selects 'continue' they are prompted to create a new account. Our program requires that all customers be signed into an account before making a purchase. The user is prompted to enter their profile details.

 Create Account—□×

Enter your first name:

Thomas

Enter your email:

tg@email.com

Enter your password:


12345

Enter your credit card number (16 digits):

1234123412341234

Create Account

4. After selecting 'Create Account' the system automatically generates a new account for the user. The user is then taken to a confirmation screen which displays the trip details of the ticket they are about to purchase.

 Confirm Purchase—□×

Here are Thomas's trip details:

Depart from: Tucson, Arrive in: Flagstaff

Bus Number: 53
Departure Time: 10:30 am
Arrival Time: 2:30 pm
Date: 12/08/20

Confirm Purchase

5. After the user clicks 'Confirm Purchase' the purchase is complete and the system adds the ticket to the user's account. A success message is also shown to the user.

Greyhound Bus Services


Login Trips Logout

GREYHOUND BUS SERVICES

~ Tucson to Phoenix ~
~ Tucson to Yuma ~

Flagstaff to Tucson ~
over to Tucson ~

Purchase Complete!

 Travel Plans for Thomas
Tucson to Flagstaff
Bus Number: 53
Departure Time: 10:30 am
Arrival Time: 2:30 pm
Date: 12/08/20
Price: \$100

OK

Departure City:

Tucson

Destination City:

Flagstaff

Date:

12/08/20

Find Tickets

6. The user then logs out by selecting the 'logout' menu option. A success message is displayed to the user.

The screenshot shows a web browser window titled "Greyhound Bus Services". The navigation bar has three tabs: "Login", "Trips", and "Logout". The "Logout" tab is selected. The main content area has a blue header with the text "GREYHOUND BUS SERVICES" and "Daily Trips From:". Below this, a list of routes is displayed: "Tucson to Phoenix ~ Phoenix to Tucson ~ Tucson to Flagstaff ~ Flagstaff to Tucson ~ Tucson to Yuma ~ Yuma to Tucson ~ Tucson to Denver ~ Denver to Tucson ~". A modal dialog box is open in the center, titled "Logged Out", with a close button (X). The dialog contains an information icon (i) and the text "Successfully Logged Out", with an "OK" button at the bottom. The background form is partially obscured by the dialog. The form fields are: "Departure City:" with the value "Tucson", "Destination City:" with the value "Flagstaff", and "Date:" with the value "12/08/20". A "Find Tickets" button is at the bottom of the form.

Greyhound Bus Services

Login Trips Logout

GREYHOUND BUS SERVICES

Daily Trips From:

~ Tucson to Phoenix ~ Phoenix to Tucson ~ Tucson to Flagstaff ~ Flagstaff to Tucson ~
~ Tucson to Yuma ~ Yuma to Tucson ~ Tucson to Denver ~ Denver to Tucson ~

Logged Out

Successfully Logged Out

OK

Departure City:

Tucson

Destination City:

Flagstaff

Date:

12/08/20

Find Tickets

Purchasing a Ticket Using a Pre-Existing Account

1. If a user already has an account in the system, they can purchase additional tickets using the same account. To get started, select the login menu item from the ticket search screen. The user will be prompted to enter an email and password

Greyhound Bus Services

Login Trips Logout

GREYHOUND BUS SERVICES

Daily Trips From:

~ Tucson to Phoenix ~ Phoenix to Tucson ~ Tucson to Flagstaff ~ Flagstaff to Tucson ~
~ Tucson to Yuma ~ Yuma to Tucson ~ Tucson to Denver ~ Denver to Tucson ~


Depart

Destination City:

Date:

Find Tickets

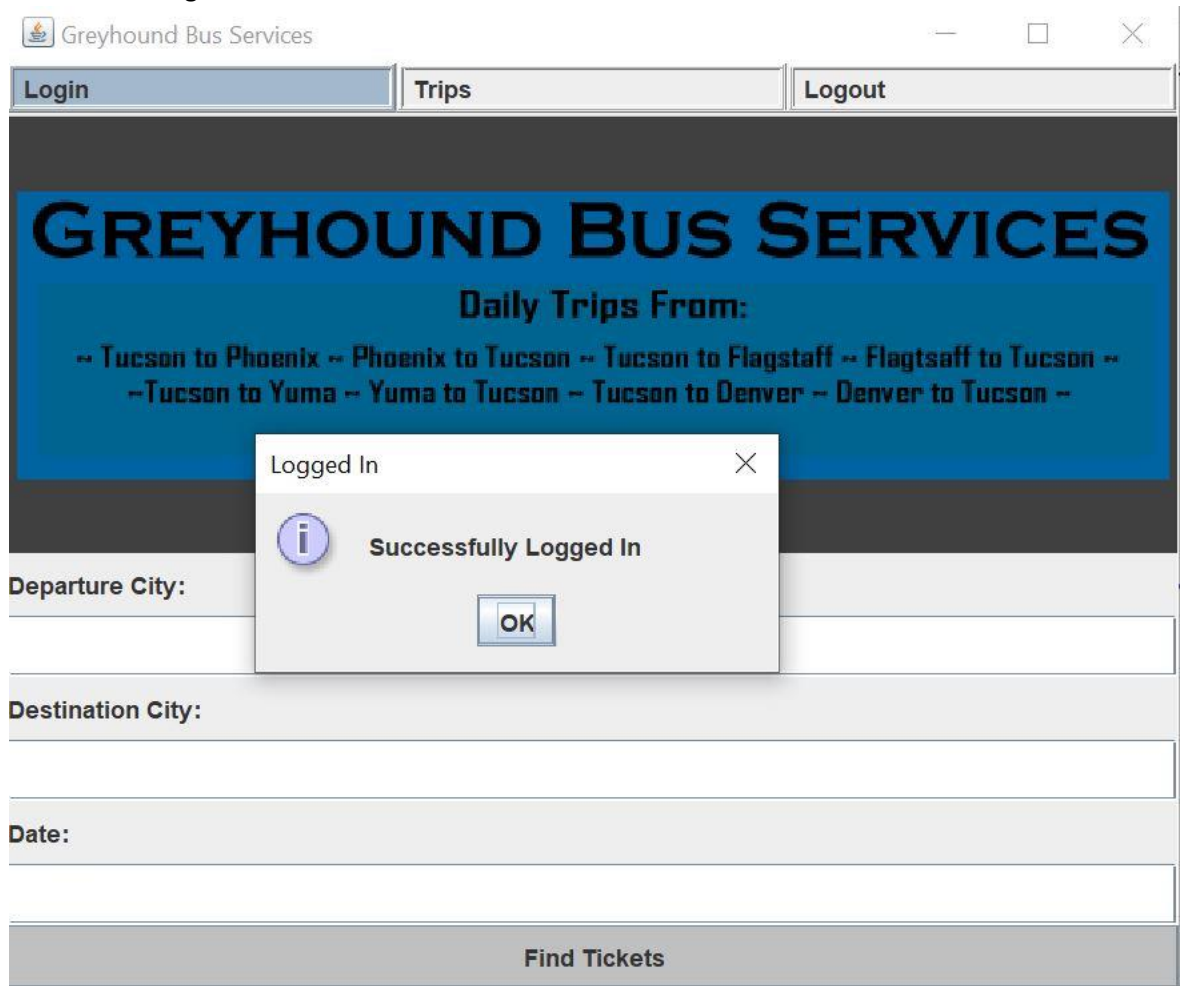
Login

 Email:

Password:

OK Cancel

2. After selecting 'ok' the user will be successfully logged into the site and can begin searching for tickets using the same method as described above.



The screenshot shows a web browser window titled "Greyhound Bus Services". The browser's address bar shows the URL "http://www.greyhound.com". The website has a navigation bar with "Login", "Trips", and "Logout" buttons. The main content area features a large blue banner with the text "GREYHOUND BUS SERVICES" and "Daily Trips From:". Below this, a list of routes is displayed: "Tucson to Phoenix ~ Phoenix to Tucson ~ Tucson to Flagstaff ~ Flagstaff to Tucson ~ Tucson to Yuma ~ Yuma to Tucson ~ Tucson to Denver ~ Denver to Tucson ~". A modal dialog box titled "Logged In" is centered on the screen, displaying an information icon, the text "Successfully Logged In", and an "OK" button. Below the dialog box, the website's search form is visible, with labels for "Departure City:", "Destination City:", and "Date:". At the bottom of the form is a "Find Tickets" button.

Greyhound Bus Services

Login Trips Logout

GREYHOUND BUS SERVICES

Daily Trips From:

~ Tucson to Phoenix ~ Phoenix to Tucson ~ Tucson to Flagstaff ~ Flagstaff to Tucson ~
~ Tucson to Yuma ~ Yuma to Tucson ~ Tucson to Denver ~ Denver to Tucson ~

Logged In

Successfully Logged In

OK

Departure City:

Destination City:

Date:

Find Tickets

Login

Trips

Logout

GREYHOUND BUS SERVICES

Daily Trips From:

~ Tucson to Phoenix ~ Phoenix to Tucson ~ Tucson to Flagstaff ~ Flagstaff to Tucson ~
~ Tucson to Yuma ~ Yuma to Tucson ~ Tucson to Denver ~ Denver to Tucson ~

Departure City:

Denver

Destination City:


Tucson

Date:

12/09/20

Find Tickets

3. As in the previous example, after selecting 'Find Tickets' a new screen will be displayed which shows all available tickets matching the passenger's criteria

 Schedule for Denver to Tucson on 12/09/20

Bus Number: 37
Departure Time: 8:15 am
Arrival Time: 9:15 pm
Date: 12/09/20
Price: \$145


Bus Number: 38
Departure Time: 9:30 am
Arrival Time: 10:30 pm
Date: 12/09/20

Enter Bus Number:

38

Continue

4. After selecting a bus number the user will be taken directly to the confirm purchase screen. The user will not be prompted to create an account, as the user is already signed in to the system.

 Confirm Purchase—□×

Here are **Thomas's** trip details:

Depart from: Denver, Arrive in: Tucson

Bus Number: 38
Departure Time: 9:30 am
Arrival Time: 10:30 pm
Date: 12/09/20

Confirm Purchase

5. After selecting 'Confirm Purchase' a success message will be displayed, showing that the user now has 2 tickets posted to their account.

Greyhound Bus Services

Login Trips Logout

GREYHOUND BUS SERVICES

~ Tucson to Phoenix ~
~ Tucson to Flagstaff ~

~ Flagstaff to Tucson ~
~ Denver to Tucson ~

Departure City:
Denver

Destination City:
Tucson

Date:
12/09/20

Purchase Complete!

Travel Plans for Thomas

Tucson to Flagstaff
Bus Number: 53
Departure Time: 10:30 am
Arrival Time: 2:30 pm
Date: 12/08/20
Price: \$100

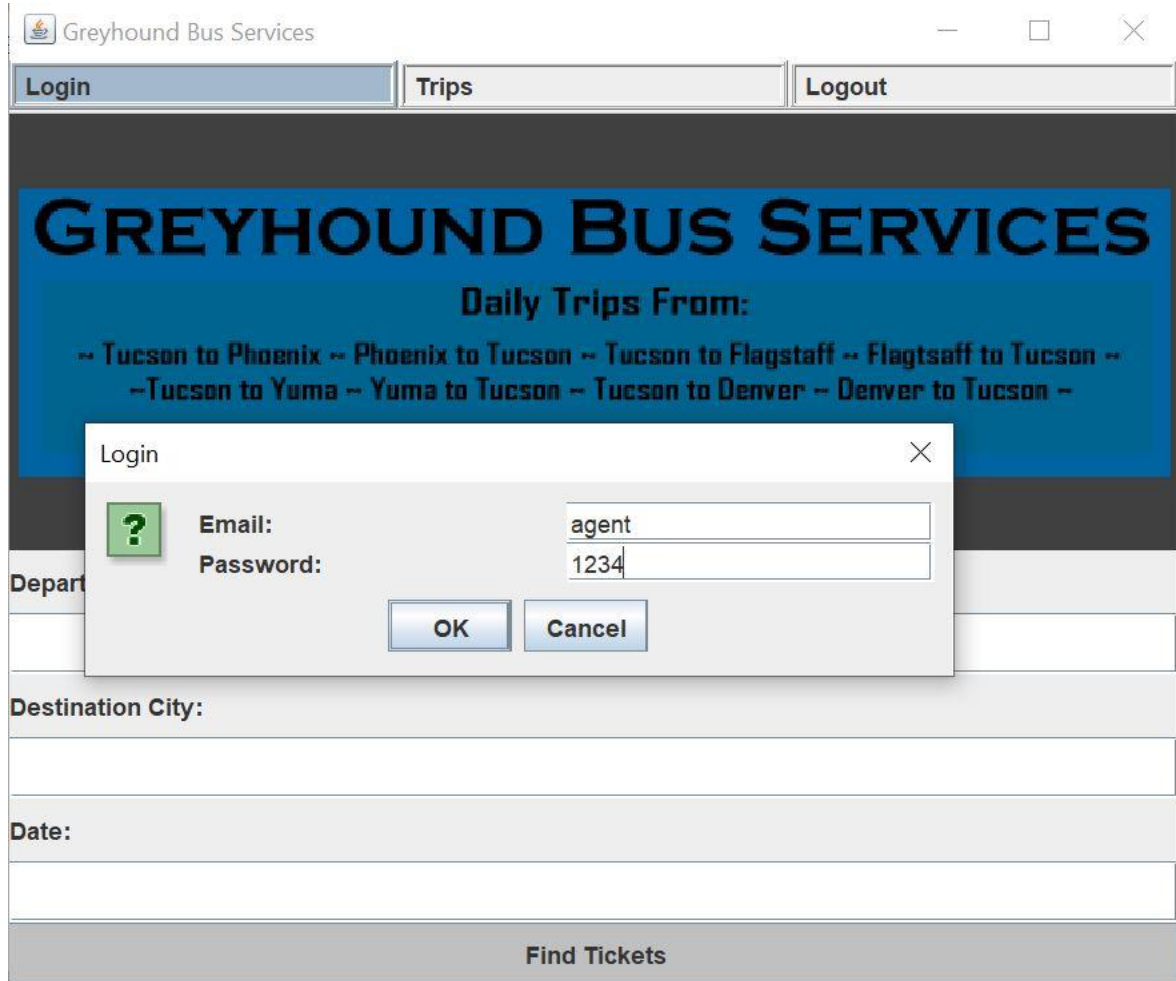
Denver to Tucson
Bus Number: 38
Departure Time: 9:30 am
Arrival Time: 10:30 pm
Date: 12/09/20
Price: \$135

OK

Find Tickets

Purchasing a Ticket as an Agent

1. The program also offers special functionality if a ticketing agent is purchasing a ticket for a customer. To begin, select the 'login' menu item from the main ticketing screen and enter the special agent email and password into the login form



The screenshot shows the 'Greyhound Bus Services' application window. The window has a title bar with the application name and standard window controls. Below the title bar is a menu bar with three items: 'Login' (highlighted), 'Trips', and 'Logout'. The main content area features a large blue banner with the text 'GREYHOUND BUS SERVICES' and 'Daily Trips From:'. Below the banner, there is a list of routes: '-- Tucson to Phoenix -- Phoenix to Tucson -- Tucson to Flagstaff -- Flagstaff to Tucson --' and '-- Tucson to Yuma -- Yuma to Tucson -- Tucson to Denver -- Denver to Tucson --'. A 'Login' dialog box is open in the foreground, containing a green question mark icon, 'Email:' and 'Password:' labels, and input fields with the values 'agent' and '1234'. The dialog box has 'OK' and 'Cancel' buttons. Below the dialog box, the main window shows a 'Depart' field, a 'Destination City:' label, and a 'Date:' label, all followed by empty input fields. At the bottom of the window is a 'Find Tickets' button.

Greyhound Bus Services

Login Trips Logout

GREYHOUND BUS SERVICES

Daily Trips From:

-- Tucson to Phoenix -- Phoenix to Tucson -- Tucson to Flagstaff -- Flagstaff to Tucson --
-- Tucson to Yuma -- Yuma to Tucson -- Tucson to Denver -- Denver to Tucson --

Login

? Email: agent
Password: 1234

OK Cancel


Depart

Destination City:

Date:

Find Tickets

- After pressing 'ok' you will be directed to a new screen accessible only to ticketing agents. On this screen the user will be prompted to enter a new customer's information, as well as the trip details for this customer.

 Agent Search — □ ×

Client Name

Connor

Client Email

cb@email.com

Client Password

12345

Client Credit Card Number

1234123412341234

Departure City:

Tucson

Destination City:


Denver

Date:

12/07/20

Find Tickets

3. As in previous examples, the system will then display all matching bus routes for the search criteria and request a bus number to be selected.

 Schedule for Tucson to Denver on 12/07/20

Bus Number: 39

Departure Time: 9:30 am

Arrival Time: 10:30 pm

Date: 12/07/20

Price: \$100

Bus Number: 40

Departure Time: 8:45 am

Arrival Time: 9:45 pm


Date: 12/07/20

Enter Bus Number:

40

Continue

4. The agent can then confirm the purchase in the confirm purchase screen. A success message will appear showing that a ticket has been added to the new passenger's account.

 Confirm Purchase—□×

Here are Connor's trip details:

Depart from: Tucson, Arrive in: Denver

Bus Number: 40
Departure Time: 8:45 am
Arrival Time: 9:45 pm
Date: 12/07/20

Confirm Purchase

Login

Trips

Logout

GREYHOUND BUS SERVICES

Daily Trips From:

~ Tucson to Phoenix ~
~ Tucson to

staff ~ Flagstaff to Tucson ~
er ~ Denver to Tucson ~

Purchase Complete!



Travel Plans for Connor
Tucson to Denver
Bus Number: 40
Departure Time: 8:45 am
Arrival Time: 9:45 pm
Date: 12/07/20
Price: \$125

OK

Departure City:

Destination City:

Date:

Find Tickets

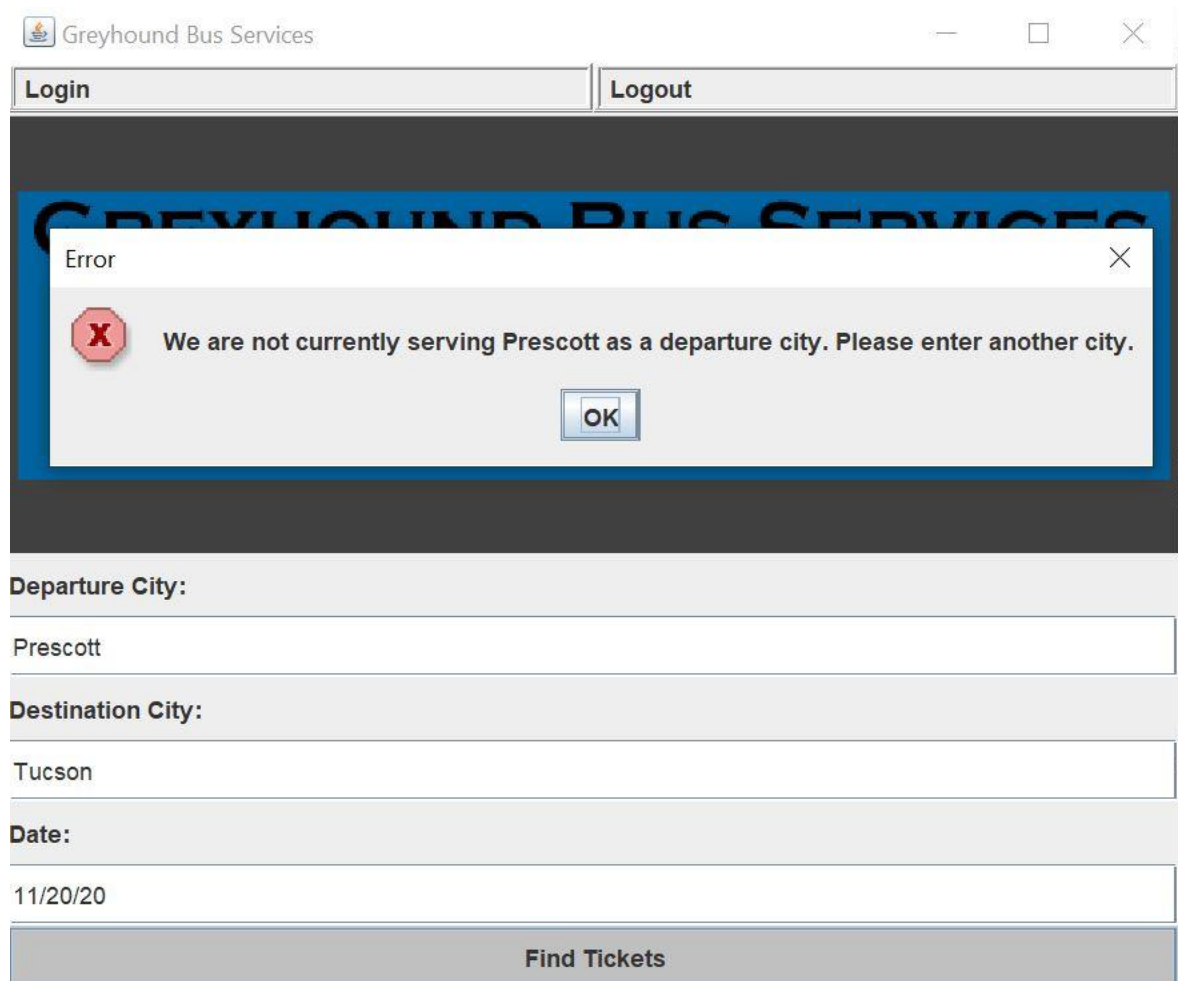
Error Checking

The following section contains all of the error checking procedures implemented by the program. Errors are organized based on where they occur in the program.

Ticket Search Errors

1. User enters an invalid departure city

If a user enters a departure city that is not listed in the database of bus routes, an error message will be displayed stating that the city is invalid. The user is able to close this error message and is prompted to enter a new departure city.

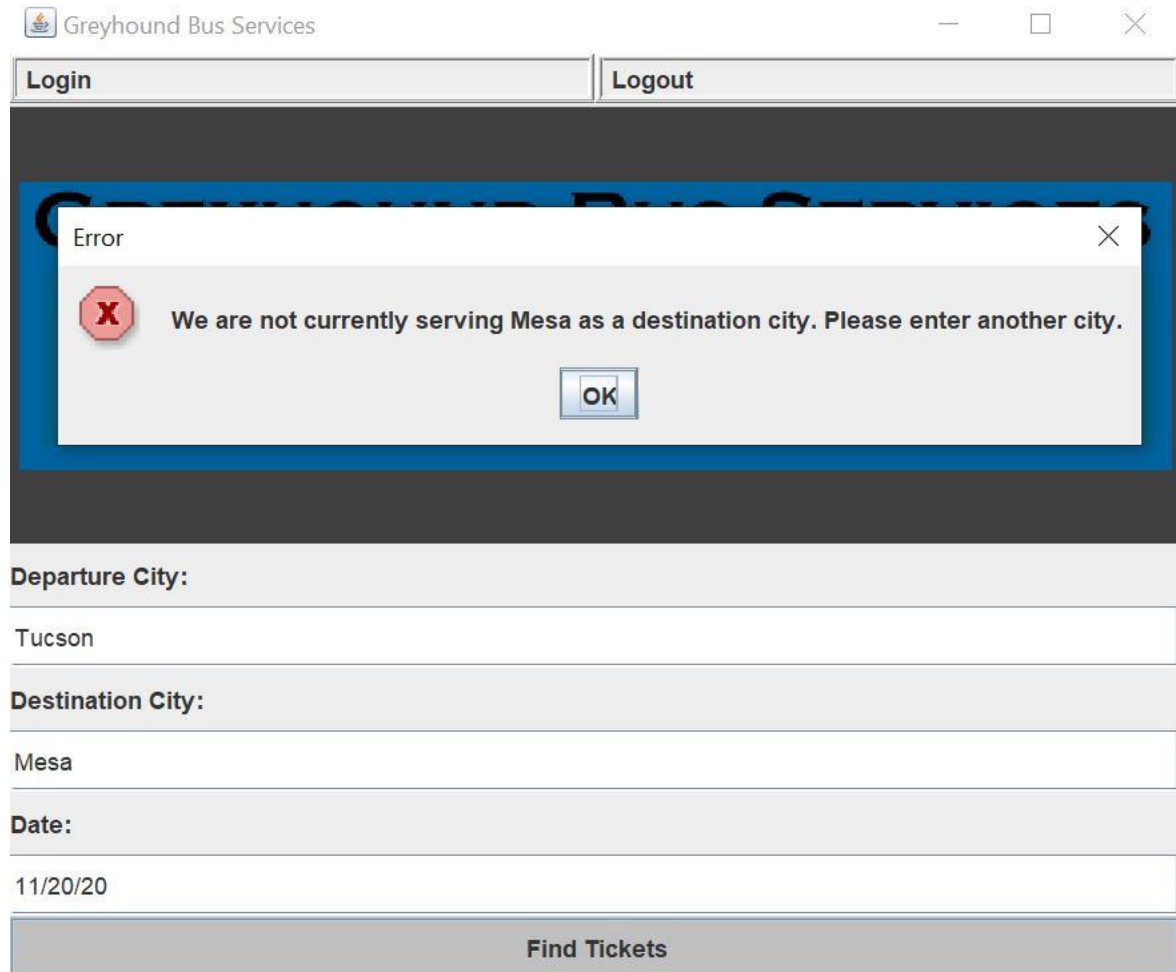


The screenshot shows a web application window titled "Greyhound Bus Services". At the top, there are "Login" and "Logout" buttons. Below these is a large blue banner with the text "GREYHOUND BUS SERVICES". An error dialog box is displayed in the center, with a red "X" icon and the message: "We are not currently serving Prescott as a departure city. Please enter another city." The dialog box has an "OK" button. Below the error message, the form fields are visible: "Departure City:" with the value "Prescott", "Destination City:" with the value "Tucson", and "Date:" with the value "11/20/20". At the bottom of the form is a "Find Tickets" button.


In the above example the user has entered Prescott as a departure city, which is not in the system's database of departure cities.

2. User enters a valid departure city, but invalid destination city

If a user enters a destination city that is not listed in the database of bus routes, an error message will be displayed alerting the user that the destination city is listed in any bus routes. The user will be able to close this error message and will be prompted to enter a new destination city.



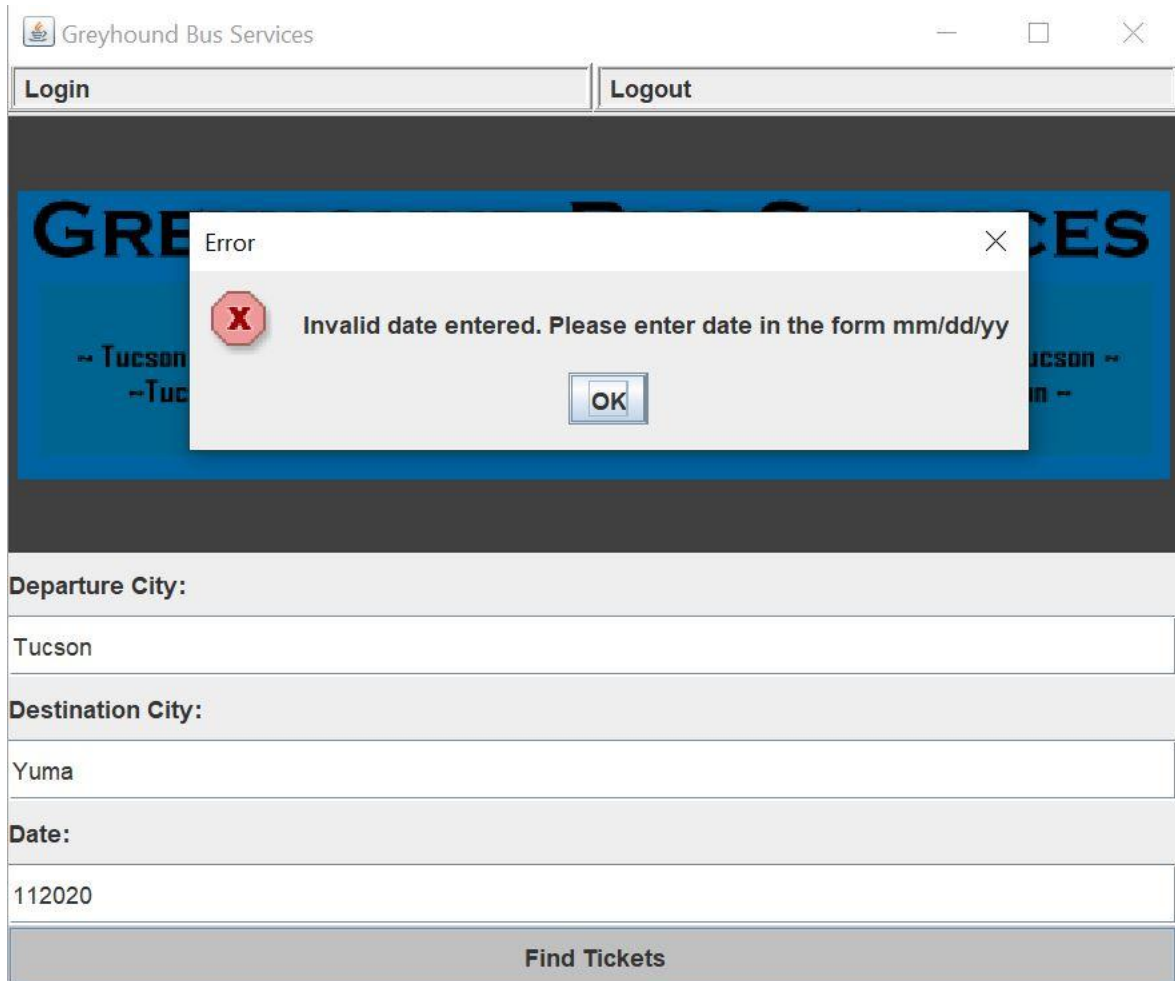
The screenshot shows a web browser window titled "Greyhound Bus Services". The browser has "Login" and "Logout" buttons in the top right. The main content area features a large "Error" dialog box with a red "X" icon. The message inside the dialog reads: "We are not currently serving Mesa as a destination city. Please enter another city." with an "OK" button below it. Below the dialog, the form fields are visible: "Departure City:" with "Tucson" entered, "Destination City:" with "Mesa" entered, and "Date:" with "11/20/20" entered. A "Find Tickets" button is at the bottom of the form.

Greyhound Bus Services	
Login	Logout
<div><div>Error</div><div> We are not currently serving Mesa as a destination city. Please enter another city.</div><div>OK</div></div>	
Departure City:	
Tucson	
Destination City:	
Mesa	
Date:	
11/20/20	
Find Tickets	

In the above example the user has entered a valid departure city, but has entered an invalid destination city. Therefore an error message is displayed stating that 'Mesa' is not a valid destination city.

3. User enters a valid departure and destination city, but enters a date in an invalid format

Our program requires that users enter dates in the format “mm/dd/yy”. If a user does not enter a date in this format an error message will be generated alerting the user that they have entered an invalid date. The user will be able to close this error message and will be prompted to enter a new date.



The screenshot shows a web application window titled "Greyhound Bus Services". It has a navigation bar with "Login" and "Logout" buttons. The main content area features a blue banner with the text "GREYHOUND BUS SERVICES" and "Tucson". Below the banner, there are three input fields: "Departure City:" with the value "Tucson", "Destination City:" with the value "Yuma", and "Date:" with the value "112020". A "Find Tickets" button is located at the bottom. An error message dialog box is displayed in the center, titled "Error", with a red 'X' icon and the text "Invalid date entered. Please enter date in the form mm/dd/yy". The dialog box has an "OK" button.

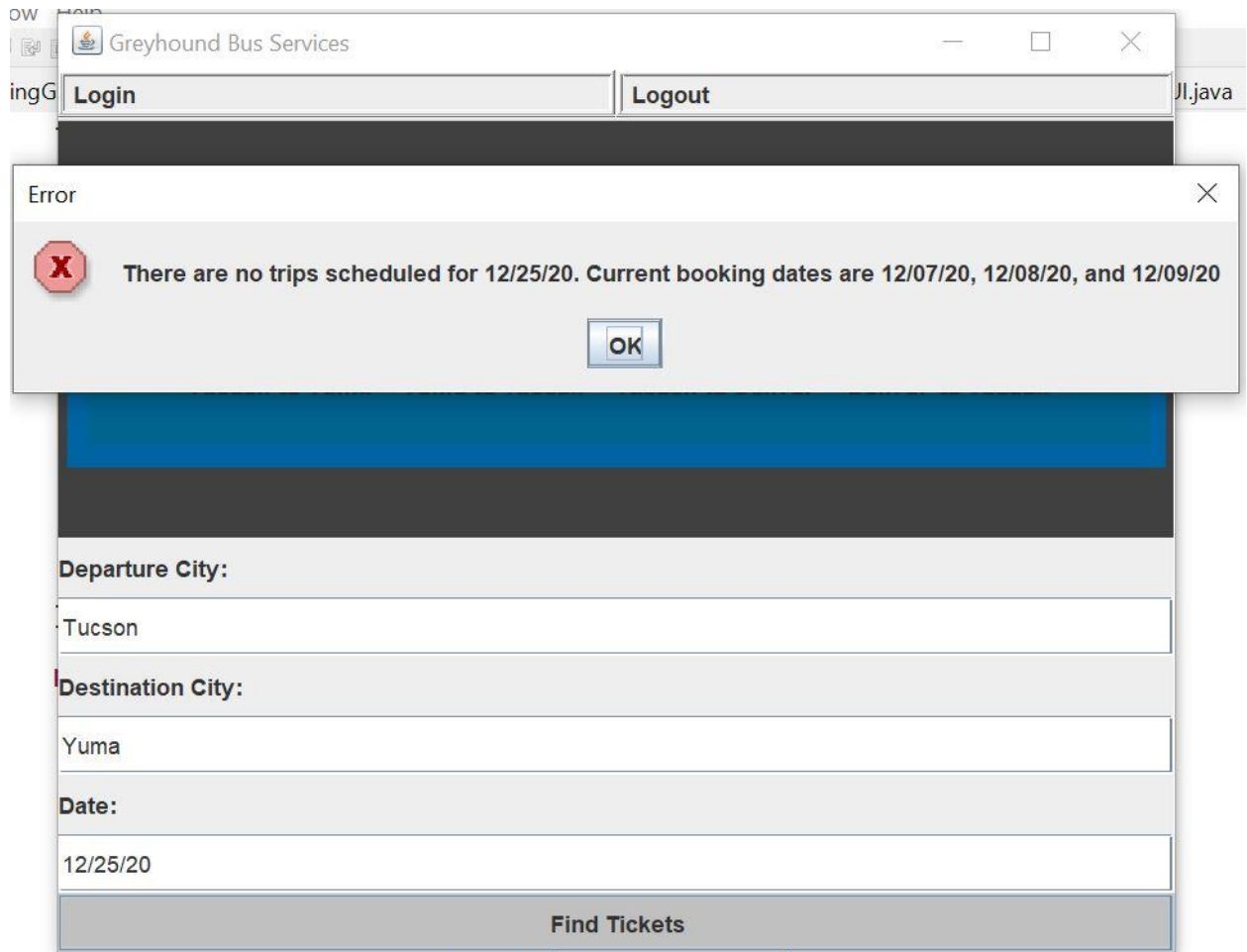
Field	Value
Departure City:	Tucson
Destination City:	Yuma
Date:	112020

Find Tickets

In the above example the user has entered a valid departure and destination city, but has entered a date that is in an invalid format. The user is alerted of this error and is notified that the valid date format is “mm/dd/yy”

4. **User enters a valid departure city, destination city, and date, but enters a date that has no matching bus routes**

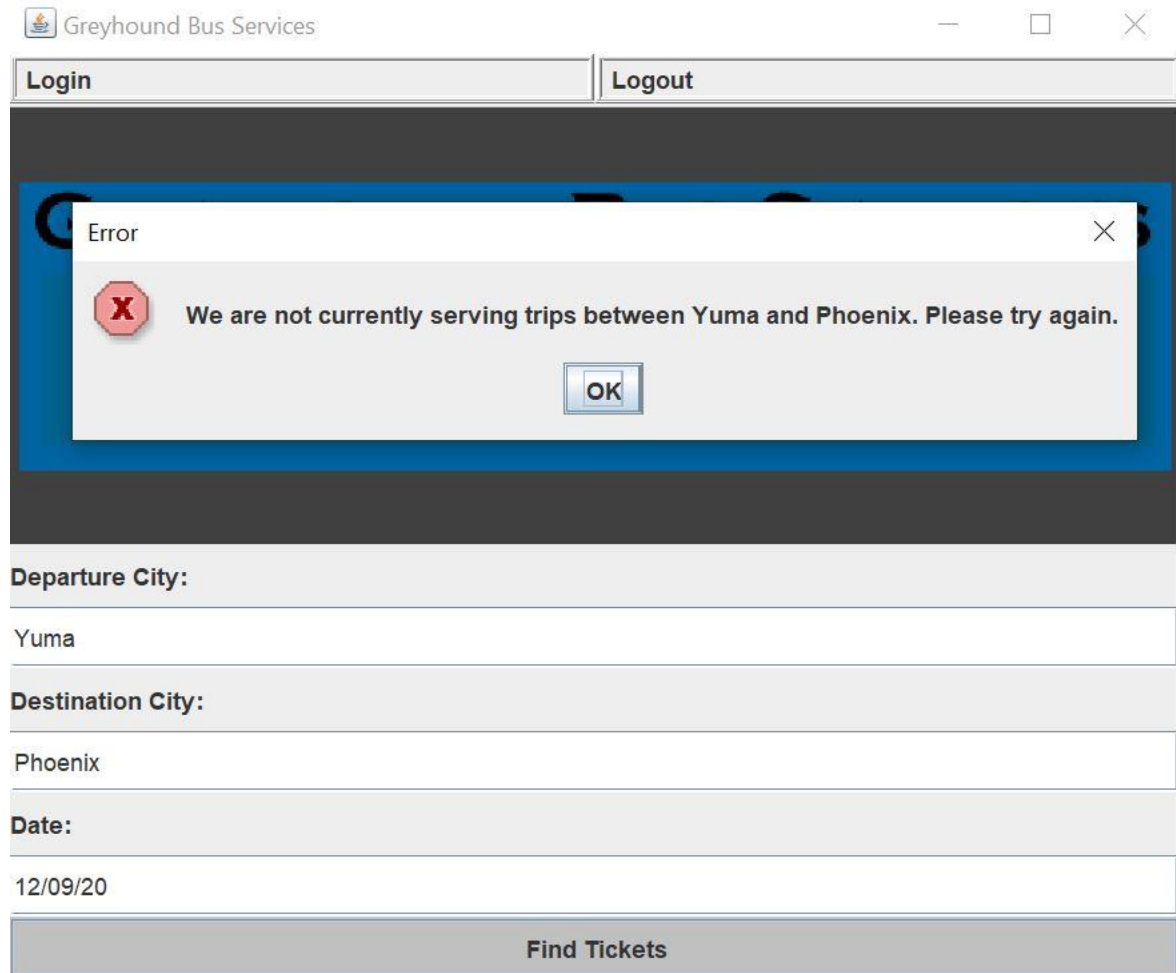
Our program will alert a user if they have entered a valid date that has no bus routes listed for that day. An error message will be generated and displayed that there are no routes for the given day, and will also display the current booking dates for our program. The user will be able to close this error message and will be prompted to enter a new date.



In the above example the user has entered a valid departure city, destination city, and date format. However, the system currently is not accepting reservations for the entered date of 12/25/20. Therefore, the user is notified of the error and told the current booking dates.

5. User enters a valid departure city, destination city, and booking date, but there are no matching routes for the two given cities

Our program alerts a user if they are about to search for a bus route that has no matching criteria (their search will return no results). If this is the case an error message will be generated alerting the user of the error. The user will be able to close the error message and will be prompted to enter new search criteria.



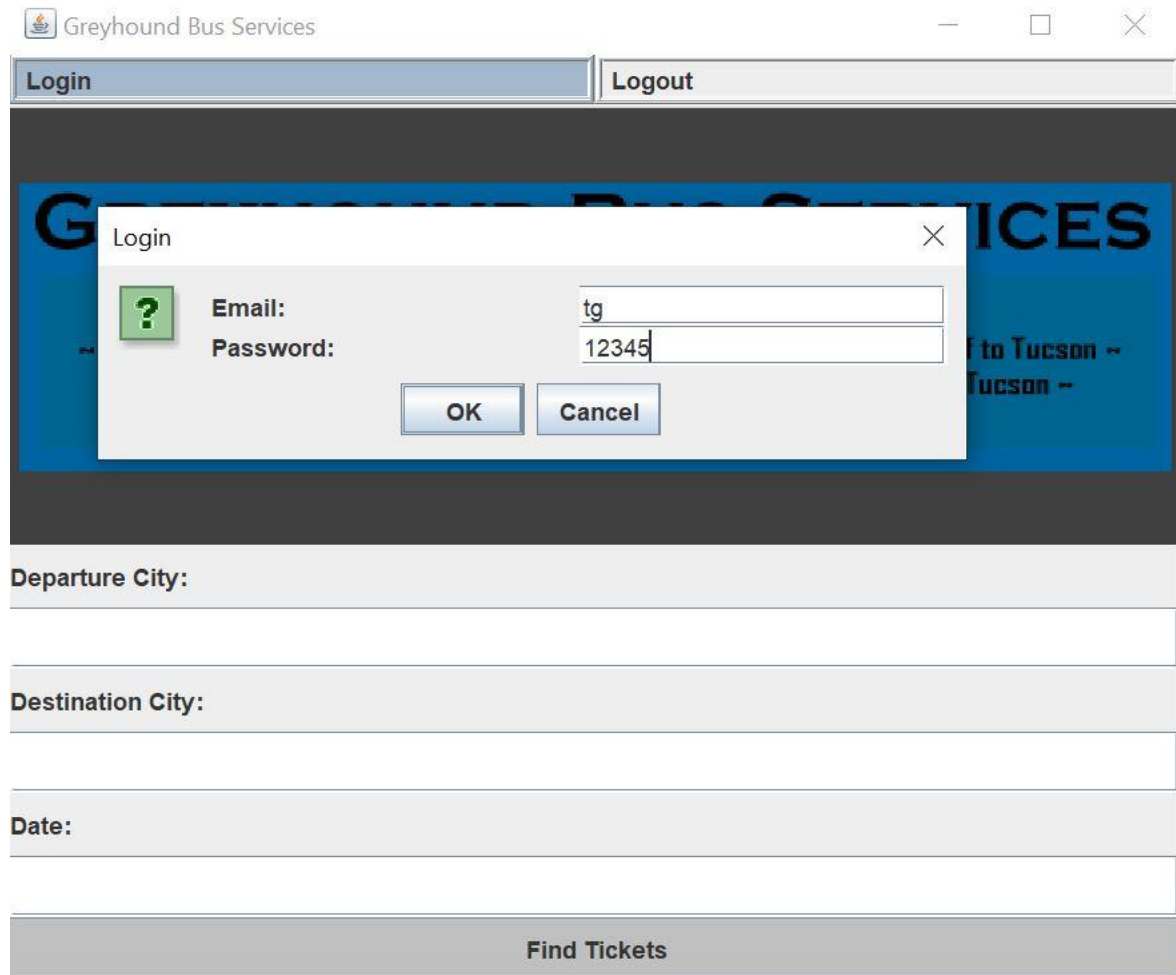
The screenshot shows a web application window titled "Greyhound Bus Services". It has a "Login" button on the left and a "Logout" button on the right. In the center, there is a blue banner with the text "Error" and a red "X" icon. Below the banner, a white error message box is displayed with the text: "We are not currently serving trips between Yuma and Phoenix. Please try again." and an "OK" button. Below the error message, there are three input fields: "Departure City:" with "Yuma" entered, "Destination City:" with "Phoenix" entered, and "Date:" with "12/09/20" entered. At the bottom, there is a "Find Tickets" button.

In the above example the user has entered valid departure and destination cities, as well as a valid booking date. However, there are no routes connecting Yuma and Phoenix for that given date. Therefore the user is notified of the error and asked to try again.

Login Errors

1. User enters an invalid email or password

If a user attempts to log into the system with a username and password that does not match a profile in the system, an error message will be generated stating that their login failed. The user will be able to close the error message and will be prompted to log in again.



The screenshot displays the Greyhound Bus Services login interface. At the top, there is a title bar with the text "Greyhound Bus Services" and standard window controls. Below the title bar, there are two tabs: "Login" (active) and "Logout". The main content area features a large blue banner with the text "Greyhound Bus Services" and "to Tucson ~ Tucson ~". Overlaid on this banner is a "Login" dialog box. The dialog box has a title bar with "Login" and a close button. It contains a green square icon with a white question mark. To the right of the icon are two input fields: "Email:" with the text "tg" and "Password:" with the text "12345". Below the input fields are two buttons: "OK" and "Cancel". Below the dialog box, there are three input fields labeled "Departure City:", "Destination City:", and "Date:". At the bottom of the interface is a button labeled "Find Tickets".

Greyhound Bus Services


Login Logout

GREYHOUND BUS SERVICES

~ Tucson to ~
~ Tucson to ~

Flagtsaff to Tucson ~
Denver to Tucson ~

Error

 Invalid email or password, please try again

OK

Departure City:

Destination City:

Date:

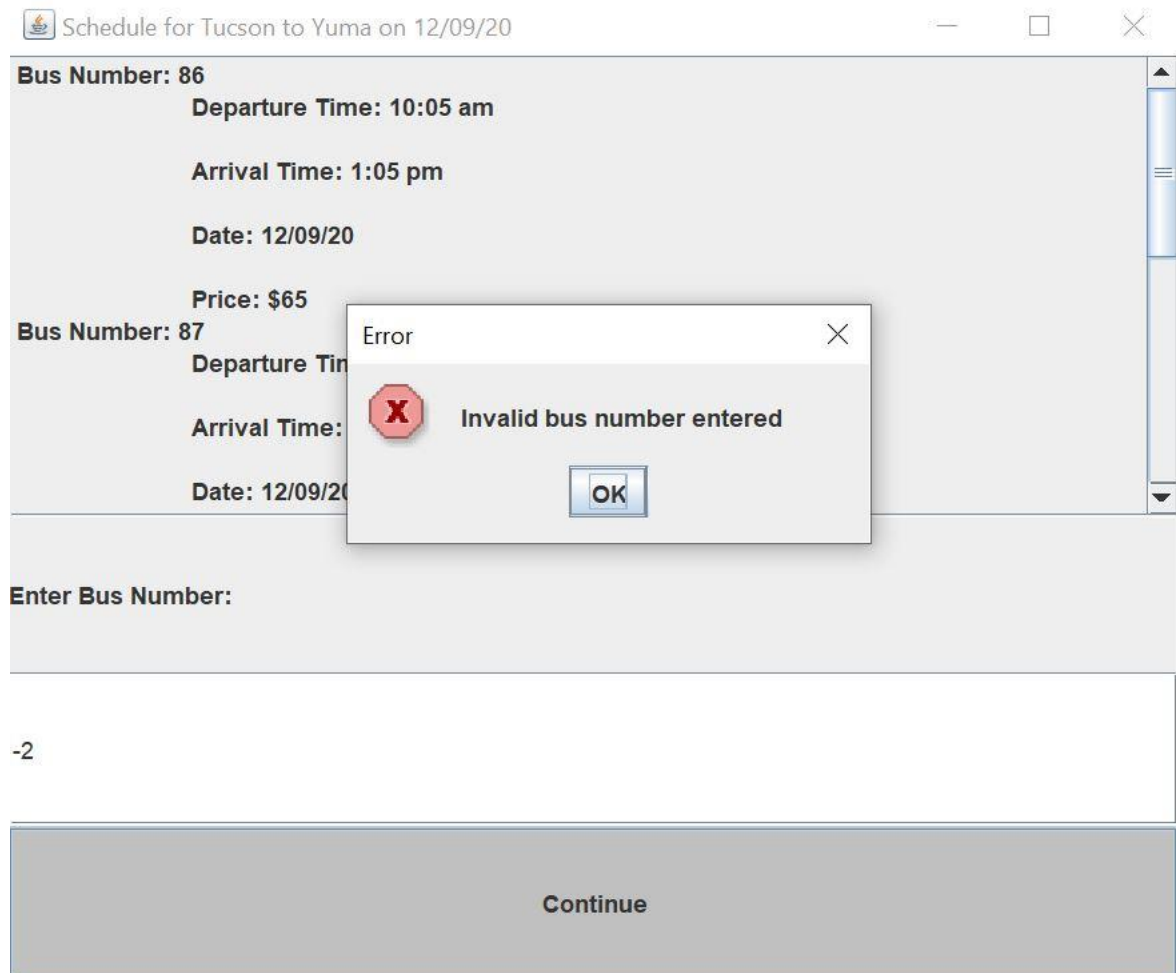
Find Tickets

In the above example the user enters in the following information for email and password: "tg" and "12345". These do not match any existing profiles in the system, so an error is generated that states that the email or password is incorrect, and the user is not logged in.

Bus Number Error

1. User enters an invalid bus number when attempting to purchase a ticket

If a user attempts to enter an invalid bus number when attempting to purchase a ticket, an error message will be displayed and the user will be prompted to enter another bus number. An example is shown below.



The screenshot shows a window titled "Schedule for Tucson to Yuma on 12/09/20". Inside the window, there is a list of bus schedules. The first entry is for Bus Number 86, with a departure time of 10:05 am, arrival time of 1:05 pm, and a price of \$65. The second entry is for Bus Number 87, with a departure time of 10:05 am, arrival time of 1:05 pm, and a price of \$65. Below the list, there is a label "Enter Bus Number:" followed by a text input field. An error message dialog box is displayed over the input field, with the title "Error" and the message "Invalid bus number entered". The dialog box has a red "X" icon and an "OK" button. Below the input field, there is a "Continue" button.

Bus Number: 86
Departure Time: 10:05 am
Arrival Time: 1:05 pm
Date: 12/09/20
Price: \$65

Bus Number: 87
Departure Time: 10:05 am
Arrival Time: 1:05 pm
Date: 12/09/20
Price: \$65

Error

Invalid bus number entered

OK

Enter Bus Number:

-2

Continue