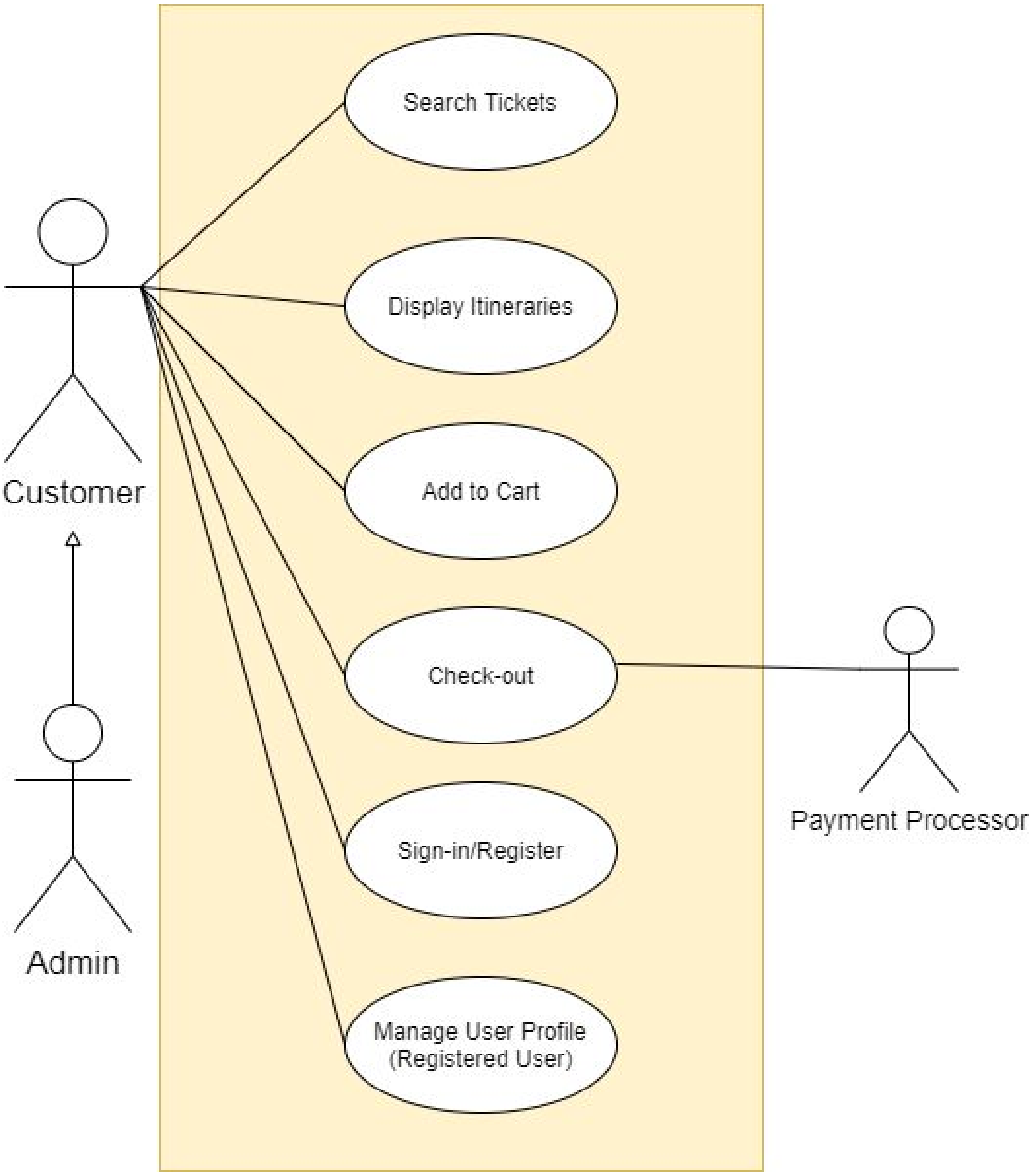
Train Ticket System – Group 4

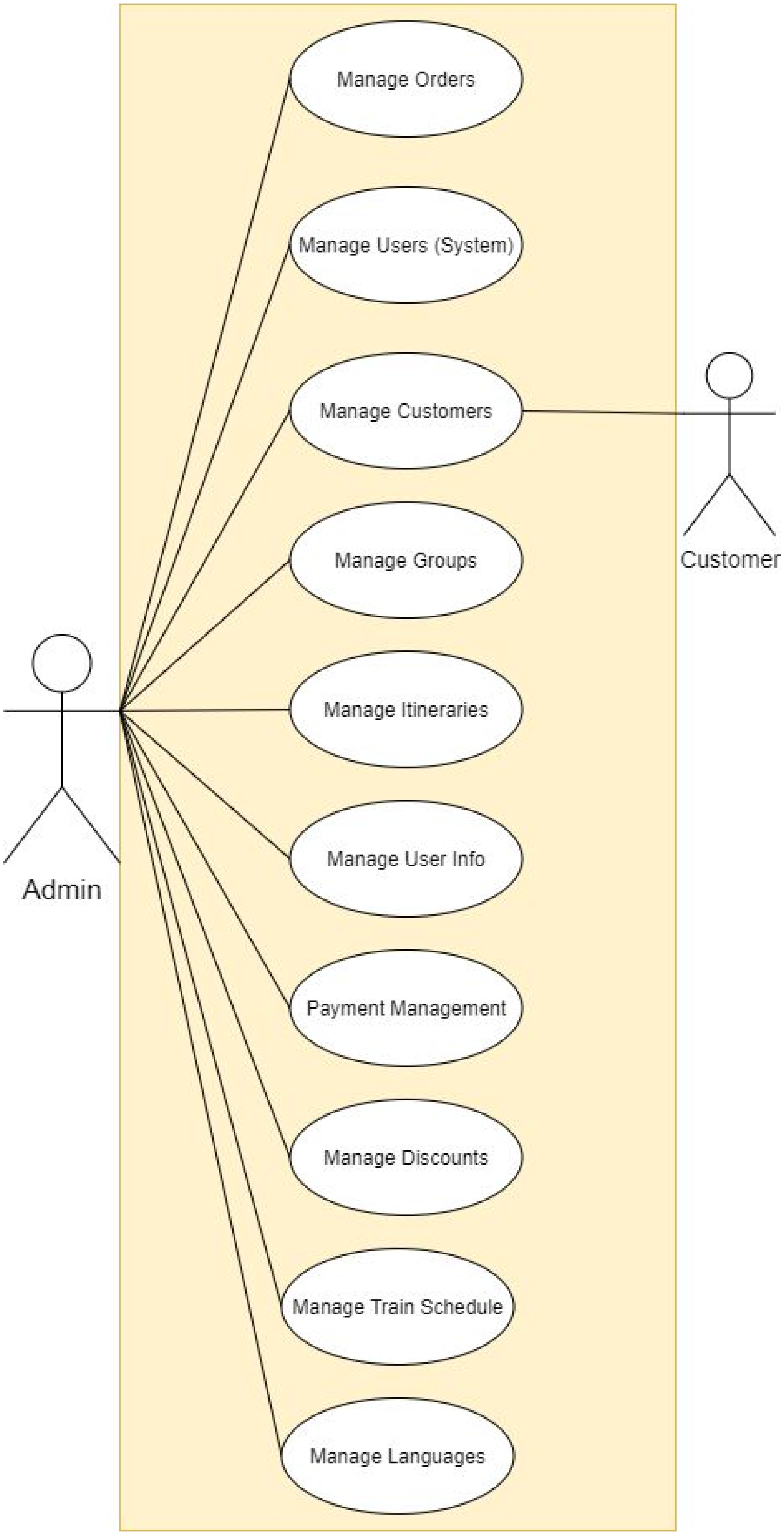
Jonathan McDaniel, Shadira Lamby, Alexandria Ostrowski,
Colton Shoenberger, Jordan Woolley, & Dylan Zeller

March 21, 2020

Table of Contents

Part One: Use Case Diagrams




Manage Orders
Manage Users (System)
Manage Customers
Manage Groups
Manage Itineraries
Manage User Info
Payment Management
Manage Discounts
Manage Train Schedule
Manage Languages
Admin
Customer

Flow of Events and User Stories

1.

Name: Search Tickets
Description: The user can browse a schedule of trips after selecting “from” and “to” location and see if tickets are available for those trips
Primary Actors: Customer, Admin
Flow of Events:

1. The user enters a location into the “from” field
2. The user enters a location into the “to” field
3. The user selects a departure date from the date-picker
4. The user selects the number of customers
5. The user clicks the “finds trains” button
6. A schedule of trips for the route requested by the user is displayed. For each route, the number of available tickets is displayed as well.
7. The user may click on a particular route to obtain more detailed information about what tickets are available (e.g. how many business class seats are available, how many coach seats, ticket prices, etc.)

Exceptional Flow of Events:

1. The user may provide an invalid entry in one of the form fields (e.g. the “from” or “to” location), in which case the system will display an error, prompting the user to correct the field
2. The user may provide valid entries in all form fields, but no trips are available, yielding no search results.

User Stories:

1. I want to view all trains heading from Philadelphia to New York this Friday.
2. I want to know how many tickets are available for a train headed back from New York to Philadelphia this afternoon.
3. I want to know the least expensive option for a train ticket from New York to Boston sometime this week.

2.

Name: Display Itineraries
Description: The user can browse, sort, and filter various itineraries
Primary Actors: Customer, Admin
Flow of Events:

1. The user clicks the “browse itineraries” button
2. The user enters a location into the “from” field
3. The user enters a location into the “to” field
4. A schedule of trips for the route is displayed in a table, with headers including date, departure time, arrival time, and train

5. The user may click on a particular table header to sort the table by data in that column
6. The user may further refine search results by filling in form fields to specify a desired date or date range, departure time or time range, etc.

Exceptional Flow of Events:

1. The user may provide an invalid entry in one of the form fields (e.g. the “from” or “to” location), in which case the system will display an error, prompting the user to correct the field
2. The user may provide valid entries in all form fields, but no trips are available, yielding no search results.

User Stories:

1. I am interested in planning a trip from Philadelphia to New York this Friday, and would like to know my options for departure times
2. I need a train ticket from New York back to Philadelphia today, and I need to make sure that I will arrive back in Philadelphia before 5pm.
3. I am planning a trip from Philadelphia to Boston and will be stopping briefly in New York on the way. I need to ensure I will have enough time between arriving in New York and boarding a train toward Boston to satisfy my needs.

3.

Name: Add to Cart

Description: The user can select tickets they are interested in purchasing and store them in a “shopping cart” to purchase later

Primary Actors: Customer, Admin

Flow of Events:

1. Use case begins when the customer views a ticket they are interested in purchasing
2. The user clicks on a radio button to choose their preferred seating option (e.g. business class or coach)
3. The user clicks a radio button to select an eligible discount, if applicable (e.g. military, student)
4. The user selects the quantity of tickets they would like to purchase
5. The user clicks the “Add to Cart” button
6. The user is directed to a view of their shopping cart. Here, they may choose to amend the amount of each item (ticket) in their shopping cart.
7. The use case ends when the user proceeds to checkout or exits the “shopping cart” page

Exceptional Flow of Events:

1. The user may select a number of tickets that exceeds the number available; the system will display an error and prompt the user to select a different amount

User Stories:

1. I am a student headed back to school from break. I need to purchase a ticket for a route from Philadelphia to New York this Friday.

2. I am a military service member who recently received a deployment letter, and need to report to base in two days. I need to purchase a ticket that will take me to the nearest city.
3. I am a father and would like to purchase tickets to take my family on vacation. I will need tickets for myself, my wife, and my two children from New York to Boston. We would like to ride in coach seats.

4.

Name: Check out

Description: The user can check out to complete a ticket purchase after adding an item to their cart.

Primary Actors: Customer, Admin

Secondary Actors: Payment Processor

Flow of Events:

1. This use case begins after the customer has successfully added a ticket to their cart.
2. The customer is prompted to input information such as name and phone number.
3. The customer completes necessary fields and selects “Continue”.
4. The customer is prompted to input payment information, such as credit card, voucher, and gift card.
5. The customer selects review purchase and confirms to end check out.

Exceptional Flow of Events:

1. *The user enters invalid information*
   The customer may provide invalid information in any of the edited fields on the "Information" or “Payment” page. The system alerts the customer of the invalid information and allows the customer to try again.
2. *The user does not edit any information on the page*
   The customer may not edit any information on the page before selecting "Continue". The system alerts the customer of missing information and allows the customer to try again.

User Stories:

1. As a customer, I want to purchase a ticket using my credit card.
2. As a customer, I want to purchase a ticket using a gift card.
3. As a customer, I want to purchase a ticket using a voucher.

5.

Name: Sign-in/Register

Description: The user can register for an account or sign in to an existing account.

Primary Actors: Customer, Admin

Flow of Events:

1. This use case begins after the customer has reached the homepage.

2. The customer then selects “Sign in” from the top of the page.
3. If the customer has an existing account, they will be prompted to enter their email address and password.
4. The customer will select “sign in” and be led to their “My account” page.
5. If the customer does not have an existing account, they will select “Join” and fill in necessary fields for first name, last name, password, email address, and country.
6. The customer's profile is created in the system.
7. The customer is directed to their “My account” page.
8. The use case ends.

Exceptional Flow of Events:

1. *The user enters invalid information*
   The customer may provide invalid information in any of the edited fields on the "Sign in" or “Join” page. The system alerts the customer of the invalid information and allows the customer to try again.
2. *The user does not enter any information on the page*
   The customer may not enter any information on the page before selecting "Sign in” or “Join”. The system alerts the customer of missing information and allows the customer to try again,

User Stories:

1. As a customer, I want to create a new profile.
2. As a customer, I want to sign in to my existing profile.
3. As a customer, I want to create a secondary profile.

6.

Name: Manage User Profile for Registered Users
Description: The user can manage the settings and information on their profile after signing into the system.
Primary Actors: Customer, Admin
Flow of Events:

1. This use case begins after the customer has successfully signed into the system.
2. The customer then hovers over the "My Account" dropdown in the header of the system and selects "My Profile".
3. The customer is then navigated to the "My Profile" page.
4. The customer will edit some information in one or many fields on the page.
5. The customer will then select "Save Profile" at the bottom of the page.
6. The customer's profile is updated in the system.
7. The customer is returned to the home page.
8. The use case ends.

Exceptional Flow of Events:

1. *The user enters invalid information*
   The customer may provide invalid information in any of the edited fields on the "My Profile" page. The system alerts the customer of the invalid information and allows the customer to try again.
2. *The user does not edit any information on the page*
   The customer may not edit any information on the page before selecting "Save Profile". This will lead the system to not update any fields and return the customer to the home page.

User Stories:

1. As a customer, I want to update my profile to use a new phone number. This will allow me to receive updates on my new phone.
2. As a customer, I want to update my profile to have a new billing address. This will allow me to receive mail at my new address.
3. As a customer, I want to remove a payment method that had been saved before because I no longer have access to that method of payment.
4. As a customer, I want to add a new method of payment because I currently have no payment methods saved in the system.

7.

Name: Manage Orders

Description: The admin can manage the orders that have been placed after signing into the system.

Primary Actors: Admin

Flow of Events:

1. This use case begins after the admin has signed into the system and has selected the "orders" link on the left pane.
2. The system displays a list of all orders placed in the system by the order ID, the customer ID, origin, destination, and order total in the table. A "details" and "remove" button are the two rightmost cells of each row.
3. The admin selects the "details" button for an order in the table.
4. The system displays an "edit order" page and a "more options" dropdown menu.
5. The admin remains on the default page and makes any changes requested necessary. This includes the total amount on the order, the origin, destination, customer ID, payment method, etc.
6. The admin selects "save order" at the bottom of the page.
7. The system will update the order details with the edited fields, and navigate to the orders page where it will display the table of orders.
8. The use case ends.

Exceptional Flow of Events:

1. *A field that was edited was invalid*
   The admin may provide invalid information in any of the edited fields on the "edit order"

page. The system alerts the admin of the invalid information and allows the admin to try again.

2. *The user clicks the remove button*
   When the table of orders is displayed, the admin selects the "remove" button in an order's row. A popup message asking the admin to confirm the removal is displayed. The admin confirms the removal, and the system removes the order.
3. *The user sorts the table by customer ID*
   When the table of orders is displayed, the admin selects the header for the customer ID column of the table. The system will sort the orders by the customer ID and display them in order.

User Stories:

1. As an admin, I want to edit the details of an order in the system. This allows me to apply discounts and edit other order details for the customers.
2. As an admin, I want to delete an order in the system because a customer wants to cancel their trip.
3. As an admin, I want to sort the orders by the customer ID. This will allow me to see all orders associated with each user.

8.

Name: Manage Users

Description: An admin of the system may add additional users to the administrative portal, modify existing users, or remove users.

Primary Actors: Admin

Flow of Events:

1. The use case begins when an admin navigates to the Manage Users section of the admin portal.
2. The admin is then presented with a list of current users and a button to add a new user.
3. To modify or delete an existing user the admin will click on a user to be taken to the details page for that user.
   a. The admin may then modify any field found on the user details page and then save changes by clicking Save at the bottom.
   b. The admin may also elect to remove a user by clicking the Delete User button found on the details page.
2. To add a new user, the admin will click the Add User button found on the main Manage Users screen

a. The admin will enter all required fields such as name, department, email address, phone, and groups that the user belongs to.

b. The admin will then save changes and be taken back to the main Manager Users page.

Exceptional Flow of Events:

1. If an admin attempts to edit a user or create a new user and fails to provide a required data field, the system will notify the admin of the user and allow a correction to be made. The admin can choose to correct the error and try to save again, or cancel.

User Stories:

1. As an admin, I want to add additional users to the admin portal so that they may complete administrative tasks on their own.
2. As an admin, I want to modify an existing user of the admin portal so that the user may have access to even more tightly restricted information as part of their promotion.
3. As an admin, I want to remove a user from the admin portal due to the user being terminated from employment.

9.

Name: Manage Customers

Description: An admin of the system may need to modify customer accounts directly to assist with customer service requests such as password resets, security question resets, or to have their account deleted.

Primary Actors: Admin

Secondary Actors: Customer

Flow of Events:

1. The use case begins when an admin navigates to the Manage Customers section of the admin portal.
2. The admin is presented with a list of all customers within a table. Each row represents a customer and each row contains buttons to reset a password, reset security questions, or delete customer.
3. To delete the customer, the admin needs only to click the “delete customer” button and then confirm the deletion by clicking confirm deletion in a popup.
4. To reset a customer’s password, the admin needs to click “reset password” and the customer will receive an email with a time sensitive link to reset their password.
5. To reset a user’s security questions, the admin needs to click “reset security questions” and the customer will receive an email with a time sensitive link to reset their security questions. The customer will also need to provide a one time code that is provided to the admin after clicking “reset security questions”.

Exceptional Flow of Events:

1. When resetting security questions, if the customer provides an incorrect one time code, the security questions will be set to their values as they were before the request was made.

User Stories:

1. As an admin I want to modify a customer’s security questions based on the outcome of a customer service call.
2. As an admin I want to be able to delete customer accounts that are deemed to be fraudulent.
3. As an admin I want to be able to reset a customer’s password to assist with customer service calls.

10.

Name: Manage Groups

Description: The *Manage Groups* use case describes how an admin user can add, edit, or remove groups .

Primary Actors: Admin

Flow of Events:

1. The use case begins when the admin user has successfully logged into their account.
2. The user clicks on *Groups*.
3. The application then displays a list of all groups including the following information: Group ID number, name, and type. The page also includes a *detail* and a *remove* button.
4. The user clicks on the *detail* button.
5. The application then displays the *Edit Group* page.
6. The admin user can edit the group's name and type.
7. The admin user then selects *Submit*.
8. The application then updates the group with the new information provided by the admin user.

Exceptional Flow of Events:

1. The admin user may enter no information into the *Group Name* field. The system returns a message alerting the user of the error.
2. The admin user may remove the *Group Type* without selecting a new one. The system returns a message alerting the user of the error.
3. The admin user makes no changes to the group. Upon exiting, the application does not update the group.

User Stories:

1. An admin user wishes to sort all active groups by number of users to better understand their user base.
2. An admin user wishes to remove a group that is no longer needed by their organization.
3. An admin user wishes to sort all active groups by name to more easily find the specific group they are searching for.

11.

Name: Manage Itineraries

Description: The *Manage Itineraries* use case describes how an admin user can add, edit, or remove itineraries.

Primary Actors: Admin

Flow of Events:

1. The use case begins when the admin user has successfully logged into their account
2. The user clicks on Itinerary
3. The application loads an Itinerary page. The page lists all itineraries with the following fields: Customer, Departure Location, Arrival Location, Date, Time, Train, Class, and Price. The page also includes an *Edit* and *Remove* button for each item.

4. The user clicks the *Edit* button for the desired itinerary they wish to manage.
5. The application loads and *Edit Itinerary* page.
6. The user then updates the information they wish to change.
7. The user then selects *Submit.*
8. The application then updates the itinerary with new information provided by the admin user.

Exceptional Flow of Events:

1. The admin user enters no information into one of the *Edit Itinerary* fields while updating an itinerary. The system returns an error alerting the user of the issue.
2. The admin user enters an *Arrival Location* that is not available for the selected *Departure Location*. When the user submits the updates, the application returns an error to the user.
3. The admin user selects the *Remove* option for an active itinerary. The application then confirms the users choice and then removes the itinerary from the system.

User Stories:

1. An admin user wishes to sort all active itineraries by departure location to better visualize the schedule of an individual train station.
2. An admin user wishes to view all of the upcoming itineraries by arrival time to see when a specific train is expected to arrive at their train station.
3. An admin user wishes to view the active itineraries by a date in the future to forecast the upcoming schedules of a single train station.

12.

Name: Manage User Information

Description: This use case describes how the admin user can add, edit, or remove information from the customer's profile.

Primary Actors: Admin

Flow of Events:

1. The use case begins when the admin user accesses the admin page and clicks the "customer" link on the left menu pane.
2. The system displays a customer list of all users registered to the system by customer ID, first and last name, and email address. A "details" and "remove" button are included in the two right most cells of each row. Two tabs also appear above the customer list--List of Customers and Customer Options.
3. The user clicks the "details" button.
4. The system displays the "edit customer" page and a "more options" dropdown menu.
5. The user remains on the default page and makes any changes requested by the customer. This includes the customer's billing address, birth date, phone number, email address, etc.
6. The user clicks the "save" button.
7. The system updates the customer's profile and displays "request successfully complete."
8. The use case ends.

Exceptional Flow of Events:

1. If the admin user clicks the “save” button before completing all required fields, the system will display an error message and prompt the user to complete all fields.
2. If the admin user mistakenly enters information (i.e. billing address, birth date, phone number, email address) in an unacceptable format, the system will display an error message and prompt the enter each field in the acceptable format.
3. If the admin user accidently clicked the “remove” button, the system would display a pop-up message asking the user to confirm or cancel the command. The user would click the “cancel” button and be returned to the customer list.
4. If the admin user wants to display all customers in chronological order, the user can hover their mouse over one of the following fields--customer’s first name, last name, or email address, and click the dropdown arrow. The user can then select to sort in ascending or descending order.

User Stories:

1. As an admin user, I want to view all customers in the customer’s list and sort them by customer name. This allows me to easily locate customers as they call in about their accounts.
2. As an admin user, I want the ability to edit customer information and for updates to be made in real-time.
3. As an admin user, I want to easily remove customers who request to no longer use our ticket system.

13.

Name: Payment Management

Description: This use case describes how the admin user is able to set all acceptable payment methods.

Primary Actors: Admin

Flow of Events:

1. The use case begins when the admin user accesses the admin page and clicks the “payment” link on the left menu pane.
2. The system displays all payment methods accepted--credit card, voucher, and gift card.
3. The user clicks one of the three payment method links. In this case, he/she clicks “credit card.”
4. The system displays the credit card page, which allows the user to enable/disable the credit card method and set it as default. The user must also enter the secret and publishable key and set the transaction type.
5. Once complete, the user clicks the “save” button.
6. The system processes the request and displays “request successfully completed.”
7. The use case ends.

Exceptional Flow of Events:

1. If the system fails to reflect the acceptable payment methods on the company webpage, then the admin user will have to repeat steps 1 through 6 of the basic flow.

2. If the admin user fails to input all fields, the system will generate an error message and prompt the user to complete all required fields.
3. If the admin user accidentally disables a payment method, then the user must repeat steps 1 through 6 of the basic flow.
4. If the admin user wants to enable vouchers, they must click the vouchers link under the payment methods page and click the enable box. They will also need to set the expiration date, parameters for acceptable voucher numbers, and the terms of the voucher. Once complete they will resume at step 5 of the basic flow.
5. If the admin user wants to enable gift cards, they must click the gift card link under the payment methods page and click the enable box. They will also need to set the parameters for acceptable gift card numbers. Once complete they will resume at step 5 of the basic flow.

User Stories:

1. As an admin user, I want to enable/disable certain payment methods.
2. As an admin user, I want to add different payment methods.
3. As an admin user, I want to remove payment methods no longer accepted by the company.

14.

Name: Manage Discounts

Description: An admin of the system may need to create a discount applying to everyone for a time period such as a holiday sale, or apply discounts to certain classes of users such as students or military.

Primary Actors: Admin

Flow of Events:

1. The use case begins when an admin navigates to the Manage Discounts section of the admin portal.
2. The admin is then given the option to modify existing discounts or create a new one
3. To create a new discount the admin clicks “Create new discount”

a. The admin then selects if this is a time, customer, or location based discount

b. For customer based, the admin selects the customer group the discount will apply to, and the discount amount as a percentage then clicks Save.

c. For time based, the admin enters the dates for which the discount will be in effect and the discount amount as a percentage, then clicks Save.

d. For location based, the admin selects the station that the discount will apply to and the discount amount as a percentage, then clicks Save.

2. To modify an existing discount, the user will click on the discount that they wish to modify.

a. The admin can then change the time, customer class, location, or discount amount depending on the type of discount being modified, then click Save.

Exceptional Flow of Events:

1. If the admin attempts to create a new discount for a location, customer class, or time period that is partially or fully covered by another discount, an error will occur. The admin may then revise their discount details or cancel altogether.
2. If the admin attempts to create a discount that is 100% or greater an error will occur. The admin may then modify the discount amount and try to save again, or cancel.

User Stories:

1. As an admin I want to be able to set a discount to all fares for a specified time period.
2. As an admin I want to be able to set a discount to a fare departing or going to specified destinations.
3. As an admin I want to be able to apply a discount to all fares purchased by certain classes of customers such as students or military.

15.

Name: Manage Train Schedule

Description: This use case describes how the admin user can add, edit, and view all train details.

Primary Actors: Admin

Flow of Events:

1. The use case begins when the admin user accesses the admin page and clicks the “Train Schedule” link on the left menu pane.
2. The system displays a train list of all trains by train number, train name, departure time, arrival time, destination, number of passengers, and available seats. Above the train list are 2 tabs--options and categories. A “details” button is included in the rightmost cell of every row.
3. The user clicks the details button.
4. The system displays the “edit train” page; allowing the user to edit departure times, add intermediate stops, or cancel the trip.
5. The user clicks the “save” button.
6. The system processes changes and displays “request successfully completed.”
7. The use case ends.

Exceptional Flow of Events:

1. If the admin user wants to add a train, the user will click the options dropdown tab and click “create new train.” The user will input all requested information (i.e. train number, train name, destination, departure/arrival time, etc.). The use case resumes at step 5 of the basic flow.
2. If the admin user wants to add a new category, the user will click the categories dropdown tab click “add new category.” The user will input all requested information. The use case resumes at step 5 of the basic flow.

User Stories:

1. As an admin user, I want to update the train schedule due to forecasted delays.
2. As an admin user, I want to add passengers to a certain train due to available seating.
3. As an admin user, I want to cancel trains due to inclement weather, train malfunctions, or railway discrepancies.

16.

Name: Manage Languages
Description: The *Manage Languages* use case describes how an admin user can change the available languages that are available to their users.
Primary Actors: Admin
Flow of Events:

1. The use case begins when the admin user has successfully logged into their account
2. The user then selects the *Languages*.
3. The application loads a *Manage Languages* page. The page lists the available languages to implement into the system with the following fields: Name, Code, and Enable.
4. The user then finds the desired language they wish to implement and select the enable option for the item.
5. The user then selects the *Submit* button.
6. The application then updates and saves the language option for the application.

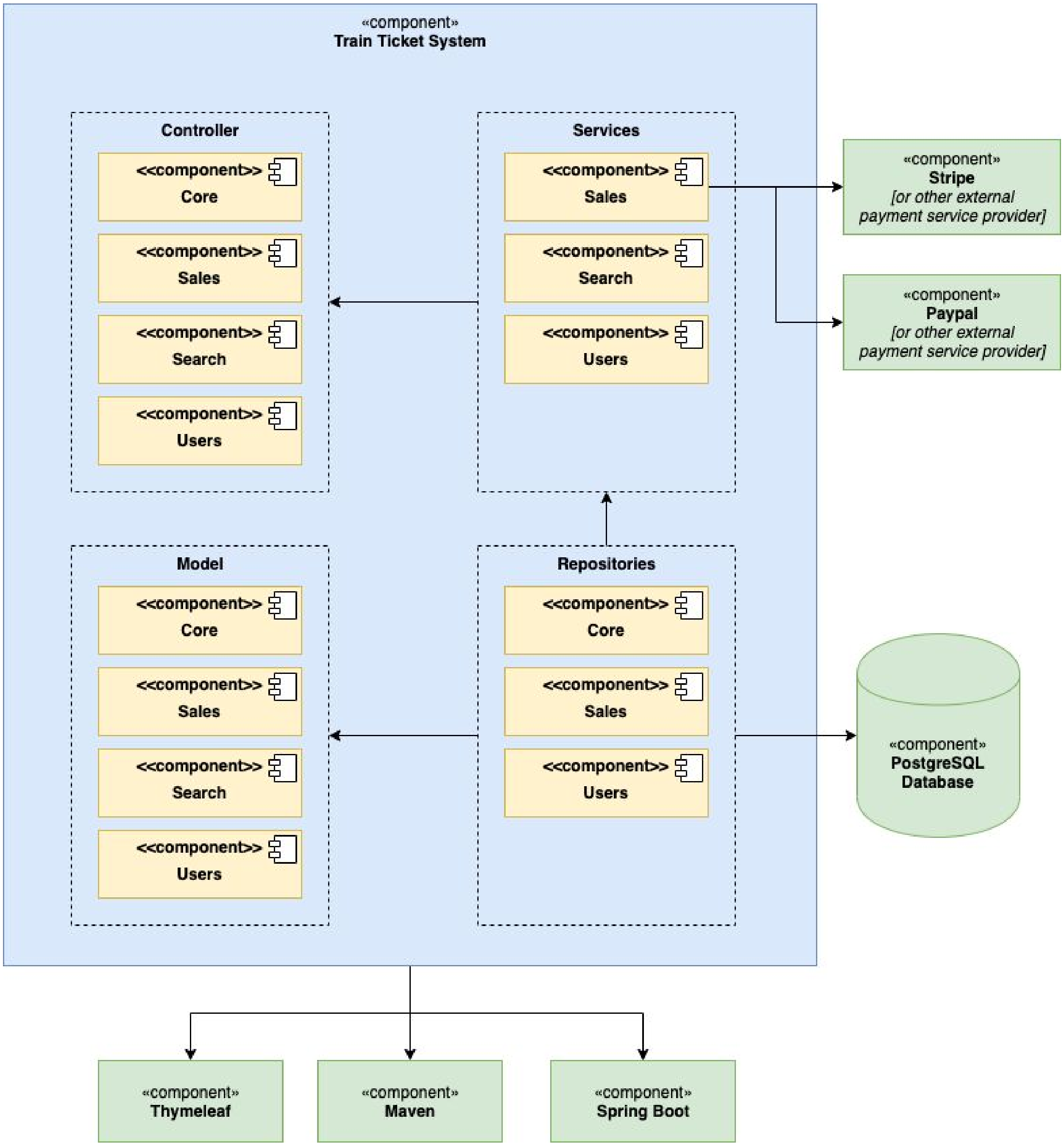
Exceptional Flow of Events:

1. The admin user has no languages selected for the application and submits the changes. The application reports an error informing the user that no language has been set.
2. The admin has made no changes to the language options of the application. When the user submits the page the application makes no changes to the application.

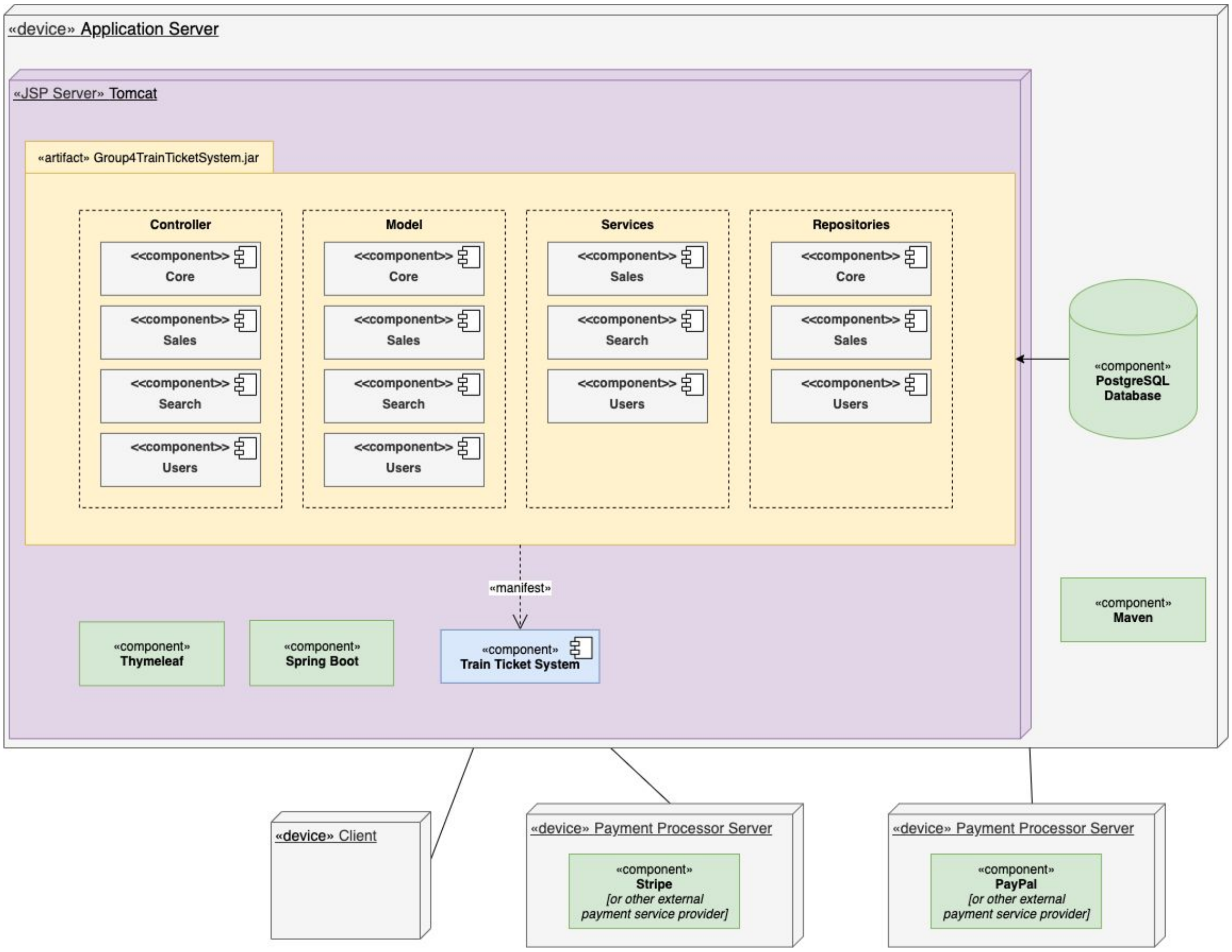
User Stories:

1. An admin user wishes to sort all languages by active and inactive to better see which languages are currently in use in their system.
2. An admin user wishes to view all of the available languages alphabetically to more easily find a language they are searching for.
3. An admin user wishes to sort the languages by activity to audit the system to see the usage of each active language.

Part Two: Component Diagram



Part Three: Deployment Diagram


«device» Application Server
«JSP Server» Tomcat
«artifact» Group4TrainTicketSystem.jar
Controller
<<component>> Core
<<component>> Sales
<<component>> Search
<<component>> Users
Model
<<component>> Core
<<component>> Sales
<<component>> Search
<<component>> Users
Services
<<component>> Sales
<<component>> Search
<<component>> Users
Repositories
<<component>> Core
<<component>> Sales
<<component>> Users
«component» PostgreSQL Database
«manifest»
«component» Thymeleaf
«component» Spring Boot
«component» Train Ticket System
«component» Maven
«device» Client
«device» Payment Processor Server
«component» Stripe [or other external payment service provider]
«device» Payment Processor Server
«component» PayPal [or other external payment service provider]