| **Identifier** | | UC-1 | |
| --- | --- | --- | --- |
| **Name** | | Signup | |
| **Description** | | This use case allows a user to sign up to the system.. | |
| **Priority** | | High | |
| **Actors** | | Users (Passengers, Admin, Applicant) | |
| **Pre-condition(s)** | | The system is running and accessible.  The user is not already registered with the system. | |
| **Post-condition(s)** | | The user is successfully registered and directed to the login page. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User selects "Sign Up" on the home page. | | User is directed to the signup page. |
| 2 | User enters their Full Name | |  |
| 3 | User enters their Email | |  |
| 4 | User selects their user type | |  |
| 5 | User sets a password | |  |
| 6 | User writes the same password in the “Confirm password field”. | | Checks if password field and confirm password field matches. |
| 7 | User clicks on the ‘Sign Up’ button | | On successful signup, User is directed to the login page |
| **Alternate Course(s) of Action** | | | |
| 1 | User forgets to enter the full name | | System displays message  “Full Name cannot be empty”. |
| 2 | User forgets to enter email | | System displays “Email field cannot be empty”. |
| 3 | User enters invalid email address | | System displays “Invalid email address”. |
| 4 | User sets a short password | | System displays “weak password – password should be at least 8 characters long”. |
| 5 | User enters an email which is already in the system | | System displays “This email already has an account enter another Email”. |
| 6 | User enters different password in the “Confirm password field” | | System displays “Password doesn’t match with the password”. |
| 7 | User clicks the "Cancel" button on the signup page. | | User is redirected back to the login page without creating an account. |

| **Identifier** | | UC-2 | |
| --- | --- | --- | --- |
| Name | | Login | |
| **Description** | | This use case involves the user login process for Railway management system with three different user types: passenger, admin, and applicant. | |
| **Priority** | | High | |
| **Actors** | | Users (Passengers, Admin, Applicant) | |
| **Pre-condition(s)** | | The system is running and accessible.  The user is registered with the system. | |
| **Post-condition(s)** | | The user is successfully logged in and directed to their respective dashboard. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User selects "Login" on the home page. | | User is directed to the login page. |
| 2 | User enters their User Name | | None |
| 3 | User enters their Password | | None |
| 4 | User selects their user type (Passengers, Admin, Applicant) | | None |
| 7 | User clicks on the ‘Login button | | User is directed to the user dashboard.  Admin: Redirected to admin dashboard  Passenger: Redirected to passenger dashboard  Applicant; directed to applicant dashboard |
| **Alternate Course(s) of Action** | | | |
| 1 | User forgets to enter the user name | | System displays message  “User Name cannot be empty”. |
| 2 | User forgets to enter password | | System displays “Password cannot be empty”. |
| 3 | User Enters incorrect username | | System displays “Username or password is incorrect”. |
| 4 | User Enters incorrect password | | System displays “Username or password is incorrect”. |

| **Identifier** | | UC-3 | |
| --- | --- | --- | --- |
| **Name** | | Forget password and reset password | |
| **Description** | | This use case involves the process of users requesting a password reset and subsequently resetting | |
| **Priority** | | High | |
| **Actors** | | Users (Passengers, Admin, Applicant) | |
| **Pre-condition(s)** | | The system is running and accessible.  The user has an existing account.  The user is not logged in. | |
| **Post-condition(s)** | | The user successfully resets their password and can log in using the new password. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User selects "Forgot Password" on the login page. | | The user is directed to the password reset page. |
| 2 | User enters their registered email address. | | None |
| 3 | User enters their User clicks the "Submit" button. | | If the provided email address is associated with an existing account, the system sends a password reset link to the email address. |
| 4 | User checks their email for the password reset link. | |  |
| 7 | User clicks the password reset link. | | The user is directed to the password reset page. |
| 8 | User enters a new password and confirms it. | |  |
| 9 | User clicks the "Reset Password" button. | | The system validates the password reset and updates the user's password.  The system sends a confirmation email to the user. |
| 10 | User receives a confirmation email | |  |
| **Alternate Course(s) of Action** | | | |
| 1 | User enters an email address that is not associated with an existing account. | | The system displays an error message: "No account found with this email address." |
| 2 | User enters a new password that does not meet security requirements (e.g., too short). | | The system displays an error message: "Password must be at least 8 characters long." |
| 3 | User enters a new password, but it does not match the password confirmation. | | The system displays an error message: "Passwords do not match." |
| 4 | User clicks the "Cancel" button on the password reset page. | | The password reset process is canceled, and the user remains on the password reset page. |
| 5 | User does not receive the password reset email. | | The system provides an option to resend the password reset email. |

| **Identifier** | | UC-4 | |
| --- | --- | --- | --- |
| **Name** | | Book Ticket | |
| **Description** | | This use case allows users to search for and book tickets. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Train data is already present in the system. | |
| **Post-condition(s)** | | Ticket details will be updated in the system. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | Passenger selects "Book Tickets" on their dashboard. | | The passenger is directed to the ticket booking page with form. |
|  | User enters the route he wants to journey on. | |  |
| 2 | User enters the Departing Station. | |  |
| 3 | User enters Arrival Station | |  |
| 4 | User enters Date. | |  |
| 5 | User presser “search” button. | | A list of trains operating in given conditions appear. |
| 6 | User selects a train and time. | | Redirect to booking details page |
| 7 | Booking Details is triggered. | |  |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 2a | User searches for a date that does not exist | | System displays “Invalid date” |
| 3a | User enters data that does not follow the existing convention. | | System displays “Wrong format” |
| 4a | User clicks on cancel. | | User is returned to the search menu. |

| **Identifier** | | UC-5 | |
| --- | --- | --- | --- |
| **Name** | | Booking details | |
| **Description** | | This use case allows users to give book details. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Triggers** | | Triggered by use case Book ticket | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Route and train data should be in system. | |
| **Post-condition(s)** | | Ticket details will be updated in the system. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 |  | | Passenger is shown a form. |
| 2 | User enters CNIC. | |  |
| 3 | User enters Name | |  |
| 4 | User enters Age. | |  |
| 5 | User enters gender. | |  |
| 6 | User enters number of tickets | |  |
| 7 | User enters Phone number | |  |
| 8 | Passenger selects the class (e.g., First Class, Business Class, Economy class) | |  |
| 9 | User clicks on “confirm booking” | | System asks for confirmation |
| 10 | USer clicks on “yes” | | Ticket(s) is/are saved and printed. |
| 11 | User is prompted to select payment details | | Payment use case is triggered |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User enters wrong format or CNIC or number | | System displays “Wrong format” |
| 2a | User does not enter data | | System displays “This field can not be left empty” |
| 3a | User clicks on cancel. | | User is returned to the search menu. |

| **Identifier** | | UC-6 | |
| --- | --- | --- | --- |
| **Name** | | Passenger Filing a Complaint | |
| **Description** | | This use case involves the process of a passenger filing a complaint about a service or experience related to the Railway Management System. | |
| **Priority** | | Medium | |
| **Actors** | | Passengers | |
| **Pre-condition(s)** | | The system is running and accessible.  The passenger is logged into their account. | |
| **Post-condition(s)** | | The passenger successfully files a complaint, and the system records the complaint for further action. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | Passenger selects "File Complaint" on the passenger dashboard. | | The passenger is directed to the complaint submission page. |
| 2 | Passenger enters details of the complaint | | None |
| 3 | Nature of the complaint | | None |
| 4 | Description of the issue | | None |
| 5 | Relevant reference numbers (e.g., ticket number) | | None |
| 6 | Passenger attach Attachments (if applicable) | | None |
| 7 | Passenger reviews the complaint details. | | None |
| 8 | Passenger clicks the "Submit Complaint" button. | | The system records the complaint and assigns it a unique complaint reference number. |
| 9 | The system sends a confirmation email to the passenger, acknowledging the receipt of the complaint. | | None |
| **Alternate Course(s) of Action** | | | |
| 1 | Passenger submits incomplete or invalid complaint details. | | The system displays error messages, prompting the passenger to provide required information. |
| 2 | Passenger clicks the "Cancel" button during the complaint submission. | | The system cancels the complaint submission, and the passenger is redirected to their dashboard. |

| **Identifier** | | UC-7 | |
| --- | --- | --- | --- |
| **Name** | | Passenger Giving Feedback | |
| **Description** | | This use case involves the process of a passenger providing feedback on their experience with the Railway Management System. | |
| **Priority** | | Low | |
| **Actors** | | Passengers | |
| **Pre-condition(s)** | | The system is running and accessible.  The passenger is logged into their account. | |
| **Post-condition(s)** | | The passenger successfully provides feedback, which may be used to improve the system or services. | |
| **Typical Course of Action** | | | |
|  | **Actor Action** | | **System Response** |
|  | Passenger selects "Give Feedback" on the passenger dashboard. | | The passenger is directed to the feedback submission page. |
|  | Passenger provides feedback in the form of comments, suggestions, or ratings, as per the available options. | |  |
|  | Passenger reviews the feedback provided. | |  |
|  | Passenger clicks the "Submit Feedback" button. | |  |
|  | The system records the feedback and stores it for review and analysis. | |  |
| **Alternate Course(s) of Action** | | | |
|  | Passenger submits incomplete or invalid feedback. | | The system provides feedback, prompting the passenger to provide more specific or complete information. |
|  | Passenger clicks the "Cancel" button during the feedback submission. | | The system cancels the feedback submission, and the passenger is redirected to their dashboard. |

| **Identifier** | | UC-8 | |
| --- | --- | --- | --- |
| **Name** | | Bank Transfer Payment | |
| **Description** | | This use case involves the process of users making payments for train tickets using a bank transfer as the payment method. | |
| **Priority** | | High | |
| **Actors** | | Admin, Passengers, Applicants | |
| **Pre-condition(s)** | | The system is running and accessible.  The user is logged into their account (for logged-in users).  The user has selected and booked train tickets. | |
| **Post-condition(s)** | | The user successfully pays for the booked train tickets using a bank transfer. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User selects "Bank Transfer Payment" on the payment page. | |  |
| 2 | User enters Bank Name | |  |
| 3 | User enters Account number | |  |
| 4 | User enter Account holder's name | |  |
| 5 | User enters Transfer reference information. | |  |
| 6 | User initiates a bank transfer to the provided account. | | System processes the transaction through respective bank |
| 7 | User returns to the system and enters transaction details, including:  Transfer reference number  Date of the transaction | |  |
| 8 | User verifies the payment details and confirms the payment. | | The system verifies the payment with the provided transaction details. The system updates the ticket status to "Paid" upon successful payment verification. |
| 9 | The user receives a payment confirmation and e-tickets for the booked train tickets. | |  |
| **Alternate Course(s) of Action** | | | |
| 1 | User selects "Bank Transfer" but does not proceed with the payment. | |  |
| 2 | The user can click a "Cancel" button, and the payment process is canceled. | |  |
| 3 | User initiates a bank transfer but provides incorrect transaction details. | | The system displays an error message and prompts the user to provide correct transaction information. |
| 4 | The payment verification process fails due to incorrect transaction details. | | The system displays an error message and prompts the user to contact support. |

| **Identifier** | | UC-9 | |
| --- | --- | --- | --- |
| **Name** | | Online Transfer of Money Payment | |
| **Description** | | This use case involves the process of users making payments for train tickets or freight services using an online transfer of money as the payment method. | |
| **Priority** | | High | |
| **Actors** | | Admin, Passengers, Applicants | |
| **Pre-condition(s)** | | The system is running and accessible.  The user is logged into their account (for logged-in users).  The user has selected and booked train tickets or freight services. | |
| **Post-condition(s)** | | The user successfully makes an online transfer of money payment, and the system verifies and records the payment. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User selects "Pay Online" for the selected booking on the payment page. | | The user is redirected to an online payment gateway. |
| 2 | The user completes the payment process on the payment gateway, providing necessary information and confirming the transaction. | |  |
| 3 | The payment gateway verifies the transaction and returns a payment confirmation to the system. | |  |
| 4 | The system updates the booking status to "Paid." | |  |
| 5 | The user receives a payment confirmation and e-tickets or booking confirmation for the selected service. | |  |
| **Alternate Course(s) of Action** | | | |
| 1 | The user fails to complete the online payment process or the payment is declined. | | The payment gateway provides an error message, prompting the user to provide valid payment information or select an alternative payment method. |
| 2 | The system fails to receive a payment confirmation from the payment gateway. | | The system displays an error message and prompts the user to contact customer support for assistance. |

| **Identifier** | | UC-10 | |
| --- | --- | --- | --- |
| **Name** | | Card Payment | |
| **Description** | | This use case involves the process of users making payments for train tickets or freight services using a card payment as the method. | |
| **Priority** | | High | |
| **Actors** | | Admin, Passengers, Applicants | |
| **Pre-condition(s)** | | The system is running and accessible.  The user is logged into their account (for logged-in users).  The user has selected and booked train tickets or freight services. | |
| **Post-condition(s)** | | . The user successfully completes a card payment, and the system verifies and records the payment. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User selects "Pay by Card" for the selected booking on the payment page. | | The user is presented with a card payment form. |
| 2 | The user enters card details, including card number, expiry date, and CVV. | |  |
| 3 | The user clicks the "Pay" button to initiate the card payment. | | The payment gateway processes the card transaction and returns a payment confirmation to the system. |
| 4 | The system updates the booking status to "Paid." | |  |
| 5 | The user receives a payment confirmation and e-tickets or booking confirmation for the selected service. | |  |
| **Alternate Course(s) of Action** | | | |
| 1 | The user enters incorrect or incomplete card details. | | The system displays an error message, prompting the user to provide valid card information. |
| 2 | The card payment transaction is declined or fails. | | The payment gateway provides an error message, prompting the user to provide a valid card or select an alternative payment method. |
| 3 | The system fails to receive a payment confirmation from the payment gateway. | | The system displays an error message and prompts the user to contact customer support for assistance |

| **Identifier** | | UC-11 | |
| --- | --- | --- | --- |
| **Name** | | Checking Ticket Availability for Passengers | |
| **Description** | | This use case involves allowing passengers to check the availability of train tickets for different compartments (First Class, Business Class, Economy Class) and view the available seat numbers. | |
| **Priority** | | Medium | |
| **Actors** | | Passengers | |
| **Pre-condition(s)** | | The system is running and accessible. | |
| **Post-condition(s)** | | The passenger successfully checks the availability of train tickets for a specific compartment and views the available seat numbers. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | Passenger selects "Check Ticket Availability" on the passenger dashboard. | | The passenger is directed to the ticket availability page. |
| 2 | Passenger selects the journey details, including:  Departure station  Destination station  Departure date  Train selection | |  |
| 3 | Passenger selects the desired compartment (First Class, Business Class, Economy Class). | |  |
| 4 | Passenger clicks the "Check Availability" button. | | The system retrieves and displays the availability information, including:  Total available seats  List of available seat numbers |
| 5 | Passenger can review the available seat numbers and select the preferred seat for booking. | |  |
| **Alternate Course(s) of Action** | | | |
| 1 | Passenger enters incomplete or invalid journey details. | | The system displays error messages, prompting the passenger to provide the required information. |
| 2 | Passenger fails to select a compartment. | | The system displays an error message: "Please select a compartment." |
| 3 | The system does not find any available seats for the selected journey and compartment. | | The system displays a message: "No seats available for the selected journey and compartment. Please modify your search." |

| **Identifier** | | UC-12 | |
| --- | --- | --- | --- |
| **Name** | | Manage Trains | |
| **Description** | | This use case allows admin to manage running trains. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Train data is already present in the system. | |
| **Post-condition(s)** | | Trains details will be updated in the system. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the update train button on the dashboard. | | Search bar appears along with text saying “search for train” |
|  | User searches for the name of the train he wants to update. | | Related trains show up in a drop down menu. |
| 2 | User selects a train from the search result. | | Train info appears with interactable fields. |
| 3 | User clicks on the field they want to update.. | | Text in the field is highlighted. |
| 4 | User enters new information in the field. | |  |
| 5 | User clicks on OK. | | System prints “information updated successfully” and returns to the search menu. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 2a | User searches for a train that does not exist | | System displays “Invalid Name” |
| 3a | User enters data that does not follow the existing convention. | | System displays “Wrong format” |
| 4a | User clicks on cancel. | | User is returned to the search menu. |

| **Identifier** | | UC-13 | |
| --- | --- | --- | --- |
| **Name** | | Schedule Train Maintenance | |
| **Description** | | This use case allows the admin to schedule a maintenance session for a train. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Train data is already present in the system. | |
| **Post-condition(s)** | | System will record the scheduled appointment. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “schedule maintenance” button on the dashboard. | | Search bar appears along with text saying “search for train” |
|  | User searches for the name of the train he wants to send to maintenance . | | Related trains show up in a drop down menu. |
| 2 | User selects a train from the search result. | | User is redirected to the scheduling page.  Page includes a list of available times and mechanics. |
| 3 | User selects a date and time for the appointment. | |  |
| 4 | User enters a brief description of the reason for the maintenance appointment. | |  |
| 5 | User selects a preferred train mechanic. | |  |
| 6 | User clicks on the “confirm appointment” button. | | System displays “appointment booked successfully” |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User searches for a train that does not exist | | System displays “Invalid Name” |
| 2a | User forgets to select date and time. | | System displays “Date and Time field empty!” |
| 3a | User forgets to give a description of the maintenance issue. | | System displays “Cannot schedule an appointment without description.!” |
| 4a | User does not select a mechanic | | System displays “Cannot schedule an appointment without a mechanic!” |
| 5a | User clicks on the “cancel” button. | | Redirected to the previous page. |

| **Identifier** | | UC-14 | |
| --- | --- | --- | --- |
| **Name** | | Cancel Train Maintenance Appointment | |
| **Description** | | This use case allows the admin to delete a maintenance session for a train. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Train data is already present in the system.  Appointment has already been made prior to this. | |
| **Post-condition(s)** | | System will delete the scheduled appointment. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “cancel previous appointment” button on the dashboard. | | Search bar appears along with text saying “search for train” |
| 2 | User searches for the name of the train he wants to cancel for or the appointment number. | | User is redirected to the appointment list page.  Page includes a list of scheduled appointments. |
| 3 | User selects the appointment from the list that he wants to cancel. | | Maintenance information appears. |
| 4 | User clicks on the “confirm cancel” button. | | System prompts user “Are you sure” |
| 5 | User clicks on the “yes” button. | | Appointment is canceled.  Redirected to the previous page. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User searches for a train that does not exist | | System displays “Invalid Name” |
| 2a | User clicks on the “no” button. | | Appointment is not canceled.  Stay on that page. |

| **Identifier** | | UC-15 | |
| --- | --- | --- | --- |
| **Name** | | Manage Schedule | |
| **Description** | | This use case allows the admin to alter the schedule for trains. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Train data is already present in the system. | |
| **Post-condition(s)** | | System will update schedule | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “Change Schedule” button on the dashboard. | | Search bar appears along with text saying “search for date” |
| 2 | User searches for the date of the schedule he wants to update. | | User is redirected to a page containing all trips for that date.. |
| 3 | User clicks on the schedule from the list that he wants to change. | | Trip info appears |
| 4 | User clicks on the field he wants to update. | | Field becomes interactable. |
| 5 | User enters new data. | |  |
| 6 | User clicks on the “confirm changes” button. | | New data is stored and displayed. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User searches for a date that does not exist | | System displays “Invalid Name” |
| 2a | User enters data that does not follow the existing convention. | | System displays “Wrong format” |
| 3a | User clicks on cancel. | | User is returned to the search menu. |

| **Identifier** | | UC-16 | |
| --- | --- | --- | --- |
| **Name** | | Manage Passengers | |
| **Description** | | This use case allows the admin to manage passengers of a train. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Train data is already present in the system.  Passenger Data is already present in the system. | |
| **Post-condition(s)** | | System will record changes. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “Manage Passengers ” button on the dashboard. | | Search bar appears along with text saying “search for date or Passenger” |
| 2 | User searches for the date or passenger id. | | User is redirected to a page containing a list of all passengers on that date or those who match the passenger info provided in search.. |
| 3 | User clicks on the person from the list that he wants to check. | | Passenger info appears |
| 4 | User clicks on the “cancel ticket” option. | | System asks for confirmation. |
| 5 | User clicks on “yes”. | | System deletes that passenger and his ticket. |
| 6 | User clicks on the “confirm changes” button. | | New data is stored and displayed. Concerned passengers are notified. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User searches for a date or passenger id that does not exist | | System displays “Invalid Information” |
| 2a | User clicks on the “no” option.. | | User is redirected to the previous page. |
| 3a | User clicks on “close”. | | User is returned to the search menu. |

| **Identifier** | | UC-17 | |
| --- | --- | --- | --- |
| **Name** | | Unverified User | |
| **Description** | | This use case tells administration about invalid CNIC | |
| **Priority** | | Medium | |
| **Actors** | | Admin, Notification System | |
| **Trigger** | | NADRA sends message saying CNIC of user is invalid | |
| **Pre-condition(s)** | | Train data is already present in the system.  Passenger Data is already present in the system. | |
| **Post-condition(s)** | | System will record changes. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | System receives a notification of invalid CNIC | | System informs admin of the issue |
| 2 | Admin may check user info. | |  |
| 3 | Admin may cancel ticket | | System records cancellation and notifies passengers. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | Admin may not cancel ticket | | Passenger is not notified |

| **Identifier** | | UC-18 | |
| --- | --- | --- | --- |
| **Name** | | Manage Complaints | |
| **Description** | | This use case allows the admin to manage complaints filed by passengers or applicants. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Train data is already present in the system.  Passenger Data is already present in the system. | |
| **Post-condition(s)** | | System will record changes. Complaint registrant will be notified. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “Complaints ” button on the dashboard. | | A list of all complaints that are unresolved is shown. |
| 2 | User clicks on the complaint from the list that he wants to check. | | Complaint info appears. |
| 3 | User enters a small description of the resolved issue. | | System records fix. |
| 4 | User clicks on the “Mark as Resolved” option. | | System asks for confirmation. |
| 5 | User clicks on “yes”. | | System updates status of the complaint. Users who registered complaints may be notified. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User does not enter Description. | | System displays “Field can not be left empty” |
| 2a | User clicks on the “no” option.. | | User is redirected to the previous page. |
| 3a | User clicks on “close”. | | User is returned to the search menu. |

| **Identifier** | | UC-19 | |
| --- | --- | --- | --- |
| **Name** | | Manage Applications | |
| **Description** | | This use case allows the admin to manage applications provided by applicants. | |
| **Priority** | | High | |
| **Actors** | | Admin, Applicant | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  Applicant Data is already present in the system. | |
| **Post-condition(s)** | | System will record changes. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “Applications ” button on the dashboard. | | A list of all applications is shown  Search bar appears along with text saying “search for position” |
| 2 | User searches for a position. | | User is redirected to a page containing a list of all applicants on that position. |
| 3 | User clicks on the person from the list that he wants to check. | | Applicant info appears |
| 4 | User clicks on the “Call for Interview” option. | | System asks for confirmation. |
| 5 | User clicks on “yes”. | | System updates status and notifies applicants. |
| 6 | Schedule Interview use case is triggered | |  |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User searches for a position that does not exist | | System displays “Invalid Information” |
| 2a | User clicks on the “no” option.. | | User is redirected to the previous page. |
| 3a | User clicks on “close”. | | User is returned to the search menu. |

| **Identifier** | | UC-20 | |
| --- | --- | --- | --- |
| **Name** | | Schedule Interview. | |
| **Description** | | This use case allows the admin to schedule an interview with the applicant. | |
| **Priority** | | Medium | |
| **Actors** | | Admin, Applicant | |
| **Triggers** | | Admin Selects call for interview options in Manage applicants use case. | |
| **Pre-condition(s)** | | The users should be logged-in to the system.  Applicant Data is already present in the system. | |
| **Post-condition(s)** | | System will record changes. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | Admin fills 4 time slots fields for interviews. | | Notifies the applicant. |
| 2 | Applicants receive notification of the interview and click on it.. | |  |
| 3 | Applicant selects desired timeslot. | |  |
| 4 | Applicant clicks on “confirm submission” | | System asks for confirmation. |
| 5 | Applicant clicks on “yes” | | Information is recorded and communicated back to the admin. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | Admin fails to provide 4 time slots. | | System Displays “Field can not be left empty” |
| 2a | Admin adds the same time slots. | | System displays “Fields can not contain identical values” |
| 3a | Applicant does not select time slot and clicks confirm | | System displays “Time slots must be chosen.” |
| 4a | Applicant presses “no” | | Redirected to the previous page. |

| **Identifier** | | UC-21 | |
| --- | --- | --- | --- |
| **Name** | | Create Job Listing. | |
| **Description** | | This use case allows the admin to create a job vacancy or listing on the system. | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-condition(s)** | | The users should be logged-in to the system. | |
| **Post-condition(s)** | | System will record changes. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | Admin clicks on the “Create new listing” button from the dashboard. | | Redirect user to a form |
| 2 | User enters Position Title | |  |
| 3 | User enters Minimum Experience | |  |
| 4 | User enters a Job description. | |  |
| 5 | User enters Minimum wage | |  |
| 6 | User clicks on “Done” | | System asks for confirmation. |
| 7 | User Clicks on “yes” | | Changes are recorded and updated. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User does not add Position Title | | System Displays “Field can not be left empty” |
| 2a | User does not add Minimum Experience | | System Displays “Field can not be left empty” |
| 3a | User does not add Job description. | | System Displays “Field can not be left empty” |
| 4a | User does not add Minimum wage | | System Displays “Field can not be left empty” |
| 5a | User clicks on “no” | | Returns to previous page |
| 6a | User clicks on “close” | | Returns to previous page |

| **Identifier** | | UC-22 | |
| --- | --- | --- | --- |
| **Name** | | Search Job | |
| **Description** | | This use case shows jobs matching the search criteria. | |
| **Priority** | | Medium | |
| **Actors** | | Applicant | |
| **Pre-condition(s)** | | The user should be logged in to the system. | |
| **Post-condition(s)** | | System will record changes. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “Apply” button from dashboard | | A list of all listings is shown  Search bar appears along with text saying “search for position” |
| 2 | User searches for a position. | | List of all applicable listings appear. |
| 3 | User clicks on preferred job | | Is redirected to a new apply page |
| 4 | User enters Name | |  |
| 5 | User enter Phone number | |  |
| 6 | User enters email | |  |
| 7 | User uploads Resume. | |  |
| 8 | User clicks “Confirm Submission” | | System records data and redirects to the previous page. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User does not add name | | System Displays “Field can not be left empty” |
| 2a | User does not add phone number | | System Displays “Field can not be left empty” |
| 3a | User does not add email | | System Displays “Field can not be left empty” |
| 4a | User does not add resume | | System Displays “Field can not be left empty” |
| 5a | User clicks on “no” | | Redirects back to the form page. |
| 6a | User clicks on close | | Redirected to previous page, data lost |

| **Identifier** | | UC-23 | |
| --- | --- | --- | --- |
| **Name** | | Check Updates | |
| **Description** | | This use case shows applicant his application history and status | |
| **Priority** | | Medium | |
| **Actors** | | Applicant | |
| **Pre-condition(s)** | | The user should be logged in to the system.  Users should have already applied to a listing. | |
| **Post-condition(s)** | | System will record changes. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “check application status” button from dashboard | | A list of all previous applications is shown |
| 3 | User clicks on preferred application | | application details appear |
| 4 | User clicks on “Reapply” | | System asks for confirmation |
| 5 | User clicks “yes” | | System resends application and records |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User clicks on “no” | | Redirects back to the form page. |
| 2a | User clicks on close | | Redirected to previous page, data lost |

| **Identifier** | | UC-24 | |
| --- | --- | --- | --- |
| **Name** | | Register Complaint | |
| **Description** | | This use case allows either passenger or applicant to register a complaint. | |
| **Priority** | | Medium | |
| **Actors** | | Applicant, Passenger | |
| **Pre-condition(s)** | | The user should be logged in to the system. | |
| **Post-condition(s)** | | System will record changes. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “Lodge Complaint” button from dashboard | | User is redirected to a Form. |
| 2 | User enters Name | |  |
| 3 | User enter Phone number | |  |
| 4 | User enters email | |  |
| 5 | User enters a brief description of the issue they faced. | |  |
| 6 | User clicks “Confirm Submission” | | System asks for confirmation. |
| 7 | User clicks on “yes”. | | System records data and redirects to the previous page. |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 1a | User does not add name | | System Displays “Field can not be left empty” |
| 2a | User does not add phone number | | System Displays “Field can not be left empty” |
| 3a | User does not add email | | System Displays “Field can not be left empty” |
| 4a | User does not add resume | | System Displays “Field can not be left empty” |
| 5a | User clicks on “no” | | Redirects back to the form page. |
| 6a | User clicks on close | | Redirected to previous page, data lost |

| **Identifier** | | UC-25 | |
| --- | --- | --- | --- |
| **Name** | | log out | |
| **Description** | | This use case allows any type of user to log out of the system. | |
| **Priority** | | High | |
| **Actors** | | User | |
| **Pre-condition(s)** | | The user should be logged in to the system. | |
| **Post-condition(s)** | | System will log out. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User clicks on the “LogOut” button. | | System asks for confirmation “Are you sure you want to log out” |
| 2 | User clicks on “yes”. | | System records data and logs the user out.. |
|  | | | |
| **Alternate Course(s) of Action** | | | |

| **Identifier** | | UC-26 | |
| --- | --- | --- | --- |
| **Name** | | Freight Booking Triggering Payment | |
| **Description** | | his use case involves the process of users booking freight services and triggering the payment process for the booked freight. | |
| **Priority** | | High | |
| **Actors** | | Admin, Passengers, Applicants | |
| **Pre-condition(s)** | | The user should be logged-in to the system.  The user has selected and provided details for booking freight services. | |
| **Post-condition(s)** | | The user successfully initiates the payment process for the booked freight services, which may include various payment methods. | |
|  | | | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User selects "Book Freight" on the dashboard and enters freight details | | Details appear |
|  | User reviews the details of the booked freight services, including the type of freight, weight, dimensions, origin, destination, and pickup and delivery dates. | | Related trains show up in a drop down menu. |
| 2 | User selects a payment method:  - Bank transfer  - Online transfer of money  - Card payment | |  |
| 3 | User verifies the payment details and confirms the payment for the booked freight. | | The system records the freight booking and payment details, including the unique booking reference and payment method. |
| 4 | The user receives a payment confirmation and booking confirmation for the freight service. | | The system verifies and processes the payment with the selected method and updates the freight booking status to "Paid. |
| 5 |  | |  |
|  | | | |
| **Alternate Course(s) of Action** | | | |
| 2a | User enters invalid data | | System displays “Invalid data” |
| 3a | User does not enter data. | | System displays “Field can not be left emptyt” |
| 4a | User clicks on cancel. | | User is returned to the search menu. |

| **Identifier** | | UC-27 | |
| --- | --- | --- | --- |
| **Name** | | Checking Freight Availability | |
| **Description** | | This use case involves allowing users to check the availability of freight services for the transportation of goods. | |
| **Priority** | | Medium | |
| **Actors** | | Admin, Passengers, Applicants | |
| **Pre-condition(s)** | | The system is running and accessible. | |
| **Post-condition(s)** | | The user successfully checks the availability of freight services and receives information about availability. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | Passenger selects "Check Ticket Availability" on the passenger dashboard. | | The passenger is directed to the ticket availability page. |
| 2 | User enters freight details, including: | |  |
| 3 | User clicks the "Check Availability" button. | |  |
| 4 | The system retrieves and displays the availability information, including:  Number of available freight services  Details of available services (e.g., type, capacity, routes) | |  |
| 5 | User reviews the available services and selects the preferred service for booking. | |  |
| **Alternate Course(s) of Action** | | | |
| 1 | User enters incomplete or invalid freight details. | | The system displays error messages, prompting the user to provide the required information. |
| 2 | The system does not find any available freight services matching the provided details | | The system displays a message: "No available freight services found for the given criteria. Please refine your search." |

| **Identifier** | | UC-28 | |
| --- | --- | --- | --- |
| **Name** | | Ticket Cancellation | |
| **Description** | | This use case involves passengers canceling their booked train tickets. | |
| **Priority** | | High | |
| **Actors** | | Passengers | |
| **Pre-condition(s)** | | The passenger is logged into their account.  The passenger has booked train tickets. | |
| **Post-condition(s)** | | The passenger successfully cancels the train tickets, and the system updates the booking status. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | Passenger selects "Cancel Tickets" on the dashboard. | |  |
| 2 | Passenger chooses the ticket(s) to cancel | |  |
| 3 | The system calculates the refund amount and processes the cancellation. | | The system updates the booking status to "Cancelled," and the passenger receives a cancellation confirmation. |
| **Alternate Course(s) of Action** | | | |
| 1 | User does not cancel the tickets | |  |
| 2 | User does not receive a confirmation email | | Message appears “Contact support” |

| **Identifier** | | UC-29 | |
| --- | --- | --- | --- |
| **Name** | | Freight Tracking | |
| **Description** | | This use case allows users to track the status and location of their booked freight services. | |
| **Priority** | | Medium | |
| **Actors** | | Admin, Passengers, Applicants | |
| **Pre-condition(s)** | | The system is running and accessible.  The user has booked freight services.. | |
| **Post-condition(s)** | | The user successfully tracks the freight and receives real-time information. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | User selects "Track Freight" on the dashboard. | | Takes to tracking page |
| 2 | User enters the booking reference number or freight details. | |  |
| 3 | The system provides real-time tracking information, including location, estimated arrival time, and status updates. | |  |
|  |  |  |  |

| **Identifier** | | UC-30 | |
| --- | --- | --- | --- |
| **Name** | | Freight Cancellation | |
| **Description** | | This use case involves passengers canceling their booked Freight. | |
| **Priority** | | High | |
| **Actors** | | Passengers | |
| **Pre-condition(s)** | | The passenger is logged into their account.  The passenger has booked Freight | |
| **Post-condition(s)** | | The passenger successfully cancels the Freight, and the system updates the booking status. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| 1 | Passenger selects "Cancel Freight " on the dashboard. | |  |
| 2 | Passenger chooses the Freight(s) to cancel | |  |
| 3 | The system calculates the refund amount (if applicable) and processes the cancellation. | | The system updates the booking status to "Cancelled," and the passenger receives a cancellation confirmation. |
| **Alternate Course(s) of Action** | | | |
| 1 | User does not cancel the Freight | |  |
| 2 | User does not receive a confirmation email | | Message appears “Contact support” |
|  |  |  |  |