NZ Horse Racing Requirements

**There may be errors or ambiguities in this information – please seek clarification from the agency if necessary. Also, please note that this information may change later.**

**NZ Horse Racing**

NZ Horse Racing agency specializes in setting up horse racing meetings and requires an application to administer its meetings and tbookings. The agency wants a system that is easy for non-technical people to use and able to handle and recover any data lost in the event of a failure.

The administrative team at NZ Horse Racing is headed by the senior administrator, Tane Johnson, and consists of sales clerks, a race manager, an equine administrator, a race course administrator, and a jockey coordinator.

The sales clerks are responsible for adding customers’ details, updating and/or deleting customers’ details and producing the report on customers. The sales clerks are also responsible for adding, updating and deleting bookings.

The race course administrator is responsible for adding meetings’ details and producing the meetings report. The race course administrator is also responsible for adding, updating and deleting race courses’ details, updating existing meetings’ details, deleting meetings’ details, and producing the race courses report.

The race manager is responsible for adding races and producing the races report. Also, the race manager is responsible for updating existing races, deleting races, adding race entries, updating race entries, and deleting race entries.

The equine administrator is responsible for adding, updating, and deleting owner details and horse details. The equine administrator is also responsible for producing the owners report.

The jockey coordinator is responsible for adding, updating, and deleting jockey details.

Tane has spoken to his staff and they all agree that they require the size of text boxes to reflect the maximum size of the corresponding data.

A race course can be associated with zero to many meetings and a meeting is associated with one race course. A meeting can be associated with zero to many races and a race is associated with one meeting.

A race can be associated with zero to many race entries and a race entry is associated with one race. A jockey can be associated with zero to many race entries and a race entry is associated with one jockey. A horse can be associated with zero to many race entries and a race entry is associated with one horse. An owner can be associated with zero to many horses and a horse is associated with one owner.

A meeting can be associated with zero to many bookings and a booking is associated with one meeting. A customer can be associated with zero to many bookings and a booking is associated with one customer.

1. Add Race Course

Use Case ID: 1

Description: This use case enables a race course administrator to add a race course’s details.

Priority: Low

Pre-conditions: The race course administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The race course administrator selects the “Add Race Course” function.
* The system displays the “Add Race Course” form with all fields blank.
* The race course administrator enters the race course’s id (1 to 999 inclusive), race course name, street address, suburb, city, and phone number).
* The system either confirms that all necessary fields are filled in correctly and saves the race course’s details or the system identifies any missing or incorrect fields and prompts for the completion of all fields.
* The system displays the “Race Course added successfully” message.
* The system displays the “Add another race course?” prompt.
* The race course administrator either elects to end the use case and the system closes the form, or elects to enter another race course and the system displays the “Add Race Course” form with all fields blank.

1. Update Race Course

Use Case ID: 2

Description: This use case enables a race course administrator to update a selected race course’s details.

Priority: Low

Pre-conditions: The race course administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The race course administrator selects the “Update Race Course” function.
* The system displays the “Update Race Course” form.
* The system displays a list of all of the race courses (race course id and race course name).
* The race course administrator either selects the race course whose details need updating or elects to cancel the operation and the system closes the form.
* The system displays the race course’s details (race course id, race course name, street address, suburb, city, and phone number).
* The race course administrator updates the relevant details (race course name, street address, suburb, city, and phone number only) and elects to update the race course.
* The system validates the entries in the fields and, either asks for confirmation to change the race course’s details, or identifies missing or incorrect fields and prompts for completion of the entry.
* The race course administrator either confirms the change of details or elects to cancel the changes and the system closes.
* The system saves the race course’s details.
* The system displays the “Race Course updated successfully” message.
* The system displays the “Update another race course?” prompt.
* The race course administrator either elects to end the use case and the system closes the form, or elects to update another race course and the system displays a list of all of the race courses (race course id and race course name).

1. Delete Race Course

Use Case ID: 3

Description: This use case enables a race course administrator to delete a selected race course’s details.

Priority: Medium

Pre-conditions: The race course administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The race course administrator selects the “Delete Race Course” function.
* The system displays the “Delete Race Course” form.
* The system displays a list of all the race courses (race course id and race course name) that do not have any meetings.
* The race course administrator selects the race course who requires deleting or the race course administrator elects to cancel the operation and the system closes the form.
* The system displays the race course’s details (race course ID, race course name, street address, and phone number).
* The race course administrator either elects to delete the race course or elects to cancel the operation and the system closes the form.
* The system displays the “Are you sure you want to delete this race course?” prompt.
* The race course administrator either confirms the deletion or elects to choose a different race course and the system displays a list of all the race courses (race course id and race course name) that do not have any meetings.
* The system deletes the race course’s details.
* The system displays the “Race Course deleted successfully” message.
* The system displays the “Delete another race course?” prompt.
* The race course administrator either elects to end the use case and the system closes the form, or the race course administrator elects to delete another race course and the system displays a list of all the race courses (race course id and race course name) that do not have any meetings.

1. Produce Race Courses Report

Use Case ID: 4

Description: This use case enables a race course administrator to produce the race courses report.

Priority: Medium

Pre-conditions: The race course administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The race course administrator selects the “Produce Race Courses Report” function
* The system displays the “Race Courses Report” form
* The race course administrator either selects the option to generate the report or elects to cancel, without generating the report, and the system closes the form.
* The system gets the details (race course id, race course name, street address, suburb, city, and phone number) of each race course.
* The system gets the details (meeting id, meeting name, and meeting date) of each meeting allocated to each race course.
* The system then generates the race courses report (race course id, race course name, street address, suburb, city, phone number of each race course plus the meeting id, meeting name, and meeting date of each meeting allocated to each race course) with the race courses sorted by race course name.
* The system closes the form to end the use case.

1. Add Meeting

Use Case ID: 5

Description: This use case enables a race course administrator to add a meeting’s details.

Priority: Medium

Pre-conditions: The race course administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The race course administrator selects the “Add Meeting” function.
* The system displays the “Add Meeting” form with all fields blank.
* The race course administrator enters the meeting’s details (meeting id (1-999999 inclusive), meeting name, meeting date, ticket price ($40.00 to $1,500.00 inclusive), and capacity (500 to 250,000 inclusive).
* The system either confirms that the details are filled in correctly or identifies that the details are missing or incorrect and prompts for the completion of the details.
* The system displays a list of race courses (race course id and race course name).
* The race course administrator selects a race course.
* The race course administrator either elects to add the meeting or elects to cancel the operation and the system closes the form.
* The system saves the meeting’s details (meeting id, meeting name, meeting date, ticket price, capacity, and race course id).
* The system displays the “Meeting added successfully” message.
* The system displays the “Add another meeting?” prompt.
* The race course administrator either elects to end the use case and the system closes the form, or elects to enter another meeting and the system displays the “Add Meeting” form with all fields blank.

1. Update Meeting

Use Case ID: 6

Description: This use case enables a race course administrator to update a selected meeting’s details.

Priority: Low

Pre-conditions: The race course administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The race course administrator selects the “Update Meeting” function.
* The system displays the “Update Meeting” form
* The system displays a list of all the meetings (meeting id and meeting name).
* The race course administrator either selects the meeting that has details that needs updating or elects to cancel the operation and the system closes the form.
* The system displays the meeting’s details (meeting id, meeting name, meeting date, ticket price, capacity, and race course name).
* The race course administrator updates the relevant details (meeting name, meeting date, ticket price, and capacity only).
* The system validates the entries in the fields and either prompts for confirmation to change the meeting’s details or identifies missing or incorrect fields and prompts for completion of the entry.
* The race course administrator either confirms the change of details or cancels the changes and the system closes the form.
* The system saves the meeting’s details.
* The system displays the “Meeting updated successfully” message.
* The system displays the “Update another meeting?” prompt.
* The race course administrator either elects to end the use case and the system closes the form, or elects to update another meeting and the system displays the “Update Meeting” form and displays a list of all the meetings (meeting id and meeting name).

1. Delete Meeting

Use Case ID: 7

Description: This use case enables a race course administrator to delete a selected meeting’s details.

Priority: Medium

Pre-conditions: The race course administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The race course administrator selects the “Delete Meeting” function.
* The system displays the “Delete Meeting” form.
* The system displays a list of all the meetings (meeting id and meeting name) that have no bookings.
* The race course administrator either selects the meeting that requires deleting or elects to cancel the operation and the system closes the form.
* The system displays the meeting’s details (meeting id, meeting name, meeting date, and race course name).
* The race course administrator either elects to delete the meeting or elects to cancel the operation and the system closes the form.
* The system displays the “Are you sure you want to delete this meeting?” prompt.
* The race course administrator either confirms the deletion or elects to choose a different meeting and the system displays a list of all the meetings (meeting id and meeting name).
* The system deletes all of the race details associated with the meeting.
* The system deletes the meeting’s details.
* The system displays the “Meeting deleted successfully” message.
* The system displays the “Delete another meeting?” prompt.
* The race course administrator either elects to end the use case and the system closes the form, or elects to delete another meeting and the system displays the “Delete Meeting” form and displays a list of all the meetings (meeting id and meeting name) that have no bookings.

1. Produce Meetings Report

Use Case ID: 8

Description: This use case enables a race course administrator to produce the meetings report.

Priority: Medium

Pre-conditions: The race course administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The race course administrator selects the “Meetings Report” function.
* The system displays the “Meetings Report” form.
* The race course administrator either selects the option to generate the report or elects to cancel without generating the report and the system closes the form.
* The system gets the details (meeting ID, meeting name, meeting date, ticket price, and capacity) of each meeting.
* The system gets the details (race course name, street address, suburb, and city) of the race course for each meeting.
* The system gets the details (race id, race name, and race type) for each race for each meeting.
* The system counts the number of bookings made for each meeting.
* The system then generates the meeting report (meeting ID, meeting name, meeting date, ticket price, capacity, race course name, street address, suburb, city, race id, race name, race type, and count of bookings for each meeting) with the meetings sorted by name.
* The system closes the form to end the use case.

1. Add Customer

Use Case ID: 9

Description: This use case enables a sales clerk to add a customer’s details.

Priority: Medium

Pre-conditions: The sales clerk has logged onto the system

Post Conditions: None

Assumptions: None

* The sales clerk selects the “Add Customer” function.
* The system displays the “Add Customer” form with all fields blank.
* The sales clerk enters the customer’s id (1-99999999 inclusive), last name, first name, street address, suburb, city, email address, phone number, and credit status (valid or invalid).
* The system either confirms that all necessary fields are filled in correctly and saves the customer’s details or the system identifies any missing or incorrect fields and prompts for the completion of all fields.
* The system displays the “Customer added successfully” message.
* The system displays the “Add another customer?” prompt.
* The sales clerk either elects to end the use case and the system closes the form, or elects to enter another customer and the system displays the “Add Customer” form with all fields blank.

1. Update Customer

Use Case ID: 10

Description: This use case enables a sales clerk to update a selected customer’s details.

Priority: Low

Pre-conditions: The sales clerk has logged onto the system

Post Conditions: None

Assumptions: None

* The sales clerk selects the “Update Customer” function.
* The system displays the “Update Customer” form
* The system displays a list of all the customers (customer id, last name and first name).
* The sales clerk either selects the customer whose details need updating or elects to cancel the operation and the system closes the form.
* The system displays the customer’s details (customer ID, last name, first name, street address, suburb, city, email address, phone number, and credit status).
* The sales clerk updates the relevant details (last name, first name, street address, suburb, city, email address, phone number, and credit status) and elects to update the customer.
* The system validates the entries in the fields and, either asks for confirmation to change the customer’s details, or identifies missing or incorrect fields and prompts for completion of the entry.
* The sales clerk either confirms the change of details or elects to cancel the changes and the system closes.
* The system saves the customer’s details.
* The system displays the “Customer updated successfully” message.
* The system displays the “Update another customer?” prompt.
* The sales clerk either elects to end the use case and the system closes the form, or elects to update another customer and the system displays the “Update customer” form and displays a list of all the customers (customer id, last name and first name).

1. Delete Customer

Use Case ID: 11

Description: This use case enables a sales clerk to delete a selected customer’s details.

Priority: Medium

Pre-conditions: The sales clerk has logged onto the system

Post Conditions: None

Assumptions: None

* The sales clerk selects the “Delete Customer” function.
* The system displays the “Delete Customer” form and displays a list of all the customers (customer id, last name and first name) who do not have any bookings.
* The sales clerk selects the customer who requires deleting or the sales clerk elects to cancel the operation and the system closes the form.
* The system displays the customer’s details (customer ID, last name, first name, street address, suburb, city, and credit status).
* The sales clerk either elects to delete the customer or elects to cancel the operation and the system closes the form.
* The system displays the “Are you sure you want to delete this customer?” prompt.
* The sales clerk either confirms the deletion or elects to choose a different customer and the system displays a list of all of the customers (customer id, last name and first name) who do not have any bookings.
* The system deletes the customer’s details.
* The system displays the “Customer deleted successfully” message.
* The system displays the “Delete another customer?” prompt.
* The sales clerk either elects to end the use case and the system closes the form, or the sales clerk elects to delete another customer and the system displays the delete customer form and displays a list of all the customers (customer id, last name and first name) who do not have any bookings.

1. Produce Customers Report

Use Case ID: 12

Description: This use case enables a sales clerk to produce the customers report.

Priority: Medium

Pre-conditions: The sales clerk has logged onto the system

Post Conditions: None

Assumptions: None

* The sales clerk selects the “Produce Customers Report” function
* The system displays the “Customers Report” form
* The sales clerk either selects the option to generate the report or elects to cancel, without generating the report, and the system closes the form.
* The system gets the details (customer’s id, last name, first name, street address, suburb, city, email address, phone number, and credit status) of each customer.
* The system counts the number of bookings each customer has made.
* The system then generates the customers report (customer id, last name, first name, street address, suburb, city, email address, phone number, credit status, and count of bookings) with the customers sorted by first name within last name.
* The system closes the form to end the use case.

1. Add Owner

Use Case ID: 13

Description: This use case enables the equine administrator to add an owner’s details.

Priority: Medium

Pre-conditions: The equine administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The equine administrator selects the “Add Owner” function.
* The system displays the “Add Owner” form with all fields blank.
* The equine administrator enters the owner’s id (1 to 9999 inclusive), last name, first name, street address, suburb, city, email address, phone number, and NZRA membership (yes or no).
* The system either confirms that all necessary fields are filled in correctly and saves the owner’s details or the system identifies any missing or incorrect fields and prompts for the completion of all fields.
* The system displays the “Owner added successfully” message.
* The system displays the “Add another owner?” prompt.
* The equine administrator either elects to end the use case and the system closes the form, or elects to enter another owner and the system displays the “Add Owner” form with all fields blank.

1. Update Owner

Use Case ID: 14

Description: This use case enables the equine administrator to update a selected owner’s details.

Priority: Low

Pre-conditions: The equine administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The equine administrator selects the “Update Owner” function.
* The system displays the “Update Owner” form
* The system displays a list of all the owners (owner id, last name and first name).
* The equine administrator either selects the owner whose details need updating or elects to cancel the operation and the system closes the form.
* The system displays the owner’s details (owner ID, last name, first name, street address, suburb, city, email address, phone number, and NZRA membership).
* The equine administrator updates the relevant details (last name, first name, street address, suburb, city, email address, phone number, and NZRA membership) and elects to update the owner.
* The system validates the entries in the fields and, either asks for confirmation to change the owner’s details, or identifies missing or incorrect fields and prompts for completion of the entry.
* The equine administrator either confirms the change of details or elects to cancel the changes and the system closes.
* The system saves the owner’s details.
* The system displays the “Owner updated successfully” message.
* The system displays the “Update another owner?” prompt.
* The equine administrator either elects to end the use case and the system closes the form, or elects to update another owner and the system displays the “Update owner” form and displays a list of all the owners (owner id, last name and first name).

1. Delete Owner

Use Case ID: 15

Description: This use case enables the equine administrator to delete a selected owner’s details.

Priority: Medium

Pre-conditions: The equine administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The equine administrator selects the “Delete Owner” function.
* The system displays the “Delete Owner” form and displays a list of all the owners (owner id, last name and first name) who do not have any horses.
* The equine administrator selects the owner who requires deleting or the equine administrator elects to cancel the operation and the system closes the form.
* The system displays the owner’s details (owner ID, last name, first name, street address, suburb, city, and NZRA membership).
* The equine administrator either elects to delete the owner or elects to cancel the operation and the system closes the form.
* The system displays the “Are you sure you want to delete this owner?” prompt.
* The equine administrator either confirms the deletion or elects to choose a different owner and the system displays a list of all of the owners (owner id, last name and first name) who do not have any bookings.
* The system deletes the owner’s details.
* The system displays the “Owner deleted successfully” message.
* The system displays the “Delete another owner?” prompt.
* The equine administrator either elects to end the use case and the system closes the form, or the equine administrator elects to delete another owner and the system displays the delete owner form and displays a list of all the owners (owner id, last name and first name) who do not have any horses.

1. Produce Owners Report

Use Case ID: 16

Description: This use case enables the equine administrator to produce the owners report.

Priority: Medium

Pre-conditions: The equine administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The equine administrator selects the “Produce Owners Report” function
* The system displays the “Owners Report” form
* The equine administrator either selects the option to generate the report or elects to cancel, without generating the report, and the system closes the form.
* The system gets the details (owner’s id, last name, first name, street address, suburb, city, email address, phone number, and NZRA membership) of each owner.
* The system gets the details (horse id and horse name) of each horse owned by each owner.
* The system then generates the owners report (owner id, last name, first name, street address, suburb, city, email address, phone number, credit status, and horse id, and horse name) with the owners sorted by first name within last name.
* The system closes the form to end the use case.

1. Add Booking

Use Case ID: 17

Description: This use case enables the sales clerk to add a booking for a customer for a selected meeting.

Priority: High

Pre-conditions: The sales clerk has logged onto the system

Post Conditions: None

Assumptions: None

* The sales clerk selects the “Add Booking” function.
* The system displays the “Add Booking” form with all fields blank.
* The system displays a list of customers (customer id, last name, and first name) with a valid credit status.
* The sales clerk selects a customer.
* The system displays a list of meetings (meeting id and meeting name).
* The sales clerk selects a meeting.
* The sales clerk enters the booking’s details (booking id (1 to 9999999999) quantity (1 to 4 inclusive), and status (pending or paid)).
* The system either confirms that the details are filled in correctly or identifies that the details are missing or incorrect and prompts for the completion of the details.
* The sales clerk either elects to add the booking or elects to cancel the operation and the system closes the form.
* The system saves the booking’s details (booking id, customer id, meeting id, booking date: today’s date, quantity, and status).
* The system displays the “Booking added successfully” message.
* The system displays the “Add another booking?” prompt.
* The sales clerk either elects to end the use case and the system closes the form, or elects to enter another booking and the system displays the “Add Booking” form with all fields blank.

1. Update Booking

Use Case ID: 18

Description: This use case enables the sales clerk to update a selected booking’s details.

Priority: High

Pre-conditions: The sales clerk has logged onto the system

Post Conditions: None

Assumptions: None

* The sales clerk selects the “Update Booking” function.
* The system displays the “Update Booking” form
* The system displays a list of all the pending bookings (booking id and booking date).
* The sales clerk either selects the booking that has details that needs updating or elects to cancel the operation and the system closes the form.
* The system displays the booking’s details (booking id, booking date, quantity, status, customer id, customer last & first name, and meeting name).
* The sales clerk updates the relevant details (quantity and status only).
* The system validates the entries in the fields and either prompts for confirmation to change the booking’s details or identifies missing or incorrect fields and prompts for completion of the entry.
* The sales clerk either confirms the change of details or cancels the changes and the system closes the form.
* The system saves the booking’s details.
* The system displays the “Booking updated successfully” message.
* The system displays the “Update another booking?” prompt.
* The sales clerk either elects to end the use case and the system closes the form, or elects to update another booking and the system displays the “Update Booking” form and displays a list of all the pending bookings (booking id and booking date).

1. Delete Booking

Use Case ID: 19

Description: This use case enables the sales clerk to delete a selected booking’s details.

Priority: High

Pre-conditions: The sales clerk has logged onto the system

Post Conditions: None

Assumptions: None

* The sales clerk selects the “Delete Booking” function.
* The system displays the “Delete Booking” form.
* The system displays a list of all the bookings (booking id and booking date).
* The sales clerk either selects the booking that requires deleting or elects to cancel the operation and the system closes the form.
* The system displays the booking’s details (booking id, booking date, quantity, status, customer id, customer last & first name, and meeting name).
* The sales clerk either elects to delete the booking or elects to cancel the operation and the system closes the form.
* The system displays the “Are you sure you want to delete this booking?” prompt.
* The sales clerk either confirms the deletion or elects to choose a different booking and the system displays a list of all the bookings (booking id and booking date).
* The system deletes the booking’s details.
* The system displays the “Booking deleted successfully” message.
* The system displays the “Delete another booking?” prompt.
* The sales clerk either elects to end the use case and the system closes the form, or elects to delete another booking and the system displays the “Delete Booking” form and displays a list of all the bookings (booking id and booking date).

1. Add Horse

Use Case ID: 5

Description: This use case enables the equine administrator to add a horse’s details.

Priority: High

Pre-conditions: The equine administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The equine administrator selects the “Add Horse” function.
* The system displays the “Add Horse” form with all fields blank.
* The equine administrator enters the horse’s details (horse id (1 to 999999 inclusive), horse name, gender (male, female or gelding), and date of birth).
* The system either confirms that the details are filled in correctly or identifies that the details are missing or incorrect and prompts for the completion of the details.
* The system displays a list of owners (owner id, last name, and first name).
* The equine administrator selects an owner.
* The equine administrator either elects to add the horse or elects to cancel the operation and the system closes the form.
* The system saves the horse’s details (horse id, horse name, gender, date of birth, and owner id).
* The system displays the “Horse added successfully” message.
* The system displays the “Add another horse?” prompt.
* The equine administrator either elects to end the use case and the system closes the form, or elects to enter another horse and the system displays the “Add Horse” form with all fields blank.

1. Update Horse

Use Case ID: 21

Description: This use case enables the equine administrator to update a selected horse’s details.

Priority: High

Pre-conditions: The equine administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The equine administrator selects the “Update Horse” function.
* The system displays the “Update Horse” form
* The system displays a list of all the horses (horse id and horse name).
* The equine administrator either selects the horse that has details that needs updating or elects to cancel the operation and the system closes the form.
* The system displays the horse’s details (horse id, horse name, gender, date of birth, and owner last name and first name).
* The equine administrator updates the relevant details (horse name, horse date, gender, and date of birth only).
* The system validates the entries in the fields and either prompts for confirmation to change the horse’s details or identifies missing or incorrect fields and prompts for completion of the entry.
* The equine administrator either confirms the change of details or cancels the changes and the system closes the form.
* The system saves the horse’s details.
* The system displays the “Horse updated successfully” message.
* The system displays the “Update another horse?” prompt.
* The equine administrator either elects to end the use case and the system closes the form, or elects to update another horse and the system displays the “Update Horse” form and displays a list of all the horses (horse id and horse name).

1. Delete Horse

Use Case ID: 22

Description: This use case enables the equine administrator to delete a selected horse’s details.

Priority: High

Pre-conditions: The equine administrator has logged onto the system

Post Conditions: None

Assumptions: None

* The equine administrator selects the “Delete Horse” function.
* The system displays the “Delete Horse” form.
* The system displays a list of all the horses (horse id and horse name) that have no race entries.
* The equine administrator either selects the horse that requires deleting or elects to cancel the operation and the system closes the form.
* The system displays the horse’s details (horse id, horse name, date of birth, and owner last name and first name).
* The equine administrator either elects to delete the horse or elects to cancel the operation and the system closes the form.
* The system displays the “Are you sure you want to delete this horse?” prompt.
* The equine administrator either confirms the deletion or elects to choose a different horse and the system displays a list of all the horses (horse id and horse name) that have no race entries.
* The system deletes all of the race details associated with the horse.
* The system deletes the horse’s details.
* The system displays the “Horse deleted successfully” message.
* The system displays the “Delete another horse?” prompt.
* The equine administrator either elects to end the use case and the system closes the form, or elects to delete another horse and the system displays the “Delete Horse” form and displays a list of all the horses (horse id and horse name) that have no race entries.

1. Add Jockey

Use Case ID: 23

Description: This use case enables the jockey coordinator to add a jockey’s details.

Priority: Medium

Pre-conditions: The jockey coordinator has logged onto the system

Post Conditions: None

Assumptions: None

* The jockey coordinator selects the “Add Jockey” function.
* The system displays the “Add Jockey” form with all fields blank.
* The jockey coordinator enters the jockey’s id (1 to 999999 inclusive), last name, first name, email address, weight (between 44.00 and 55.00 inclusive), and date of birth.
* The system either confirms that all necessary fields are filled in correctly and saves the jockey’s details or the system identifies any missing or incorrect fields and prompts for the completion of all fields.
* The system displays the “Jockey added successfully” message.
* The system displays the “Add another jockey?” prompt.
* The jockey coordinator either elects to end the use case and the system closes the form, or elects to enter another jockey and the system displays the “Add Jockey” form with all fields blank.

1. Update Jockey

Use Case ID: 24

Description: This use case enables the jockey coordinator to update a selected jockey’s details.

Priority: Low

Pre-conditions: The jockey coordinator has logged onto the system

Post Conditions: None

Assumptions: None

* The jockey coordinator selects the “Update Jockey” function.
* The system displays the “Update Jockey” form and displays a list of all of the jockeys (jockey id, last name, and first name).
* The jockey coordinator either selects the jockey whose details need updating or elects to cancel the operation and the system closes the form.
* The system displays the jockey’s details (jockey’s id, last name, first name, email address, date of birth, and weight).
* The jockey coordinator updates the relevant details (last name, first name, email address, date of birth, and weight only) and elects to update the jockey.
* The system validates the entries in the fields and, either asks for confirmation to change the jockey’s details, or identifies missing or incorrect fields and prompts for completion of the entry.
* The jockey coordinator either confirms the change of details or elects to cancel the changes and the system closes.
* The system saves the jockey’s details.
* The system displays the “Jockey updated successfully” message.
* The system displays the “Update another jockey?” prompt.
* The jockey coordinator either elects to end the use case and the system closes the form, or elects to update another jockey and the system displays a list of all of the jockeys (jockey id, last name, and first name).

1. Delete Jockey

Use Case ID: 25

Description: This use case enables the jockey coordinator to delete a selected jockey’s details.

Priority: Medium

Pre-conditions: The jockey coordinator has logged onto the system

Post Conditions: None

Assumptions: None

* The jockey coordinator selects the “Delete Jockey” function.
* The system displays the “Delete Jockey” form.
* The system displays a list of all the jockeys (jockey id, last name and first name) who are not allocated to any race entries.
* The jockey coordinator selects the jockey who requires deleting or the jockey coordinator elects to cancel the operation and the system closes the form.
* The system displays the jockey’s details (jockey ID, last name, first name, weight, and date of birth).
* The jockey coordinator either elects to delete the jockey or elects to cancel the operation and the system closes the form.
* The system displays the “Are you sure you want to delete this jockey?” prompt.
* The jockey coordinator either confirms the deletion or elects to choose a different jockey and the system displays a list of all the jockeys (jockey id, last name and first name) who are not allocated to any race entries.
* The system deletes the jockey’s details.
* The system displays the “Jockey deleted successfully” message.
* The system displays the “Delete another jockey?” prompt.
* The jockey coordinator either elects to end the use case and the system closes the form, or the jockey coordinator elects to delete another jockey and the system displays a list of all the jockeys (jockey id, last name and first name) who are not allocated to any race entries.

1. Add Race Entry

Use Case ID: 26

Description: This use case enables a race manager to add a race entry to a selected race.

Priority: High

Pre-conditions: The race manager has logged onto the system

Post Conditions: None

Assumptions: None

* The race manager selects the “Add Race Entry” function.
* The system displays the “Add Race Entry” form.
* The system displays a list of all the scheduled races (race id and race name).
* The race manager either selects the race to which the entry will be added or elects to cancel the operation and the system closes the form.
* The system displays the race’s details (race id, race name, and meeting name).
* The system displays the list of the race entries (horse name, jockey last name, and jockey first name) already added to the selected race.
* The system displays a list of the horses (horse id and horse name).
* The race manager selects a horse.
* The system displays the error message “The horse has already entered this race” if the selected horse is already entered into the selected race and then displays a list of the horses (horse id and horse name)
* The system displays a list of the jockeys (jockey id, last name, and first name)
* The race manager selects a jockey.
* The system displays the error message “The jockey has already entered this race on another horse” if the selected jockey is already entered into the selected race and then displays a list of the jockeys (jockey id, last name, and first name)
* The race manager enters the status (pending, disqualified, or confirmed) of the race entry for the race.
* The race manager elects to add the race entry to the race or elects to cancel the operation and the system closes the form.
* The system generated a unique race entry id (1-9999999999) for the race entry.
* The system saves the race entry details (race entry id, race id, horse id, jockey id, status, and race entry time (set to zero)).
* The system displays the “Race entry added to race successfully” message.
* The system displays the “Add another race entry?” prompt.
* The race manager either elects to end the use case and the system closes the form, or the race manager elects to add another race entry and the system displays the “Add Race Entry” form and displays a list of all the scheduled races (race id and race name).

1. Update Race Entry

Use Case ID: 27

Description: This use case enables the race manager to update a selected race entry.

Priority: Medium

Pre-conditions: The race manager has logged onto the system

Post Conditions: None

Assumptions: None

* The race manager selects the “Update Race Entry” function.
* The system displays the “Update Race Entry” form
* The system displays a list of all the scheduled races (race id and race name) that have race entries.
* The race manager either selects the race with the race entry which is being updated or elects to cancel the operation and the system closes the form.
* The system displays the race’s details (race id, race name, and status).
* The system displays the list of the race entries (race entry id, horse name, jockey last name, and jockey first name) associated with the selected race.
* The race manager selects a race entry.
* The system displays the details of the race entry (race entry id, horse name, jockey last name, jockey first name, status, and race entry time).
* The race manager updates the race entry’s status and race time (0 to 500 inclusive)
* The race manager elects to update the race entry or elects to cancel the operation and the system closes the form.
* The system updates race entry’s details.
* The system displays the “Race entry updated successfully” message.
* The system displays the “Update another race entry?” prompt.
* The race manager either elects to end the use case and the system closes the form, or the race manager elects to update another race entry and the system displays the “Update Race Entry” form and displays a list of all the scheduled races (race id and race name) that have race entries.

1. Delete Race Entry

Use Case ID: 28

Description: This use case enables the race manager to delete a selected race entry.

Priority: High

Pre-conditions: The race manager has logged onto the system

Post Conditions: None

Assumptions: None

* The race manager selects the “Delete Race Entry” function.
* The system displays the “Delete Race Entry” form
* The system displays a list of all the scheduled races (race id and race name) that have race entries.
* The race manager either selects the race with the race entry which is being deleted or elects to cancel the operation and the system closes the form.
* The system displays the race’s details (race id, race name, and status).
* The system displays the list of the race entries (race entry id, horse name, jockey last name, and jockey first name) associated with the selected race.
* The race manager selects a race entry.
* The system displays the details of the race entry (race entry id, horse name, jockey last name, jockey first name, status, and race entry time).
* The race manager elects to delete the race entry or elects to cancel the operation and the system closes the form.
* The system deletes race entry’s details.
* The system displays the “Race entry deleted successfully” message.
* The system displays the “Delete another race entry?” prompt.
* The race manager either elects to end the use case and the system closes the form, or the race manager elects to delete another race entry and the system displays the “Delete Race Entry” form and displays a list of all the scheduled races (race id and race name) that have race entries.

1. Add Race

Use Case ID: 29

Description: This use case enables a race manager to add a race’s details.

Priority: Medium

Pre-conditions: The race manager has logged onto the system

Post Conditions: None

Assumptions: None

* The race manager selects the “Add Race” function.
* The system displays the “Add Race” form with all fields blank.
* The race manager enters the race’s details (race id (1-99999999 inclusive), race name, race time (format: HH:MM, e.g. 15:30, range 09:00 to 19:00 inclusive), status (scheduled or finished), and race type (hurdles, chase, handicap, or flat)).
* The system either confirms that the details are filled in correctly or identifies that the details are missing or incorrect and prompts for the completion of the details.
* The system displays a list of meetings (meeting id and meeting name).
* The race manager selects a meeting.
* The race manager either elects to add the race or elects to cancel the operation and the system closes the form.
* The system saves the race’s details (race id, race name, race time, status, race type, and meeting id).
* The system displays the “Race added successfully” message.
* The system displays the “Add another race?” prompt.
* The race manager either elects to end the use case and the system closes the form, or elects to enter another race and the system displays the “Add Race” form with all fields blank.

1. Update Race

Use Case ID: 30

Description: This use case enables a race manager to update a selected race’s details.

Priority: Medium

Pre-conditions: The race manager has logged onto the system

Post Conditions: None

Assumptions: None

* The race manager selects the “Update Race” function.
* The system displays the “Update Race” form
* The system displays a list of all the scheduled races (race id and race name).
* The race manager either selects the race that has details that needs updating or elects to cancel the operation and the system closes the form.
* The system displays the race’s details (race id, race name, race time, status, race type, and meeting name).
* The race manager updates the relevant details (race name, race time, status, and race type only).
* The system validates the entries in the fields and either prompts for confirmation to change the race’s details or identifies missing or incorrect fields and prompts for completion of the entry.
* The race manager either confirms the change of details or cancels the changes and the system closes the form.
* The system saves the race’s details.
* The system displays the “Race updated successfully” message.
* The system displays the “Update another race?” prompt.
* The race manager either elects to end the use case and the system closes the form, or elects to update another race and the system displays the “Update Race” form and displays a list of all the schedule races (race id and race name).

1. Delete Race

Use Case ID: 31

Description: This use case enables a race manager to delete a selected race’s details.

Priority: Medium

Pre-conditions: The race manager has logged onto the system

Post Conditions: None

Assumptions: None

* The race manager selects the “Delete Race” function.
* The system displays the “Delete Race” form.
* The system displays a list of all the finished races (race id and race name).
* The race manager either selects the race that requires deleting or elects to cancel the operation and the system closes the form.
* The system displays the race’s details (race id, race name, and meeting name).
* The race manager either elects to delete the race or elects to cancel the operation and the system closes the form.
* The system displays the “Are you sure you want to delete this race?” prompt.
* The race manager either confirms the deletion or elects to choose a different race and the system displays a list of all the races (race id and race name) that have no artists allocated.
* The system deletes all of the details of race entries associated with the race.
* The system deletes the race’s details.
* The system displays the “Race deleted successfully” message.
* The system displays the “Delete another race?” prompt.
* The race manager either elects to end the use case and the system closes the form, or elects to delete another race and the system displays the “Delete Race” form and displays a list of all the finished races (race id and race name).

1. Produce Races Report

Use Case ID: 32

Description: This use case enables a race manager to produce the races report.

Priority: Medium

Pre-conditions: The race manager has logged onto the system

Post Conditions: None

Assumptions: None

* The race manager selects the “Races Report” function.
* The system displays the “Races Report” form.
* The race manager either selects the option to generate the report or elects to cancel without generating the report and the system closes the form.
* The system gets the details (race ID, race name, race time, status race type, and meeting name) of each race.
* The system gets the details (horse name, status, jockey last name and jockey first name) of each race entry for each race.
* The system then generates the race report (race ID, race name, race time, status, race type, meeting name, horse name, status, jockey last name and jockey first name) with the races sorted by race name.
* The system closes the form to end the use case.