

**Restaurant Manager User Guide**

**What is RManager?**

RManager is a tool for managing bookings in a restaurant environment. It allows staff members to add, edit, remove and view bookings for bookings made on specific times and dates.

Additionally, RManager will provide a tool for the automatic allocation of bookings to their respective tables based on multiple factors.

**Current Progress**

Currently, RManager provides a platform through which restaurant staff can:

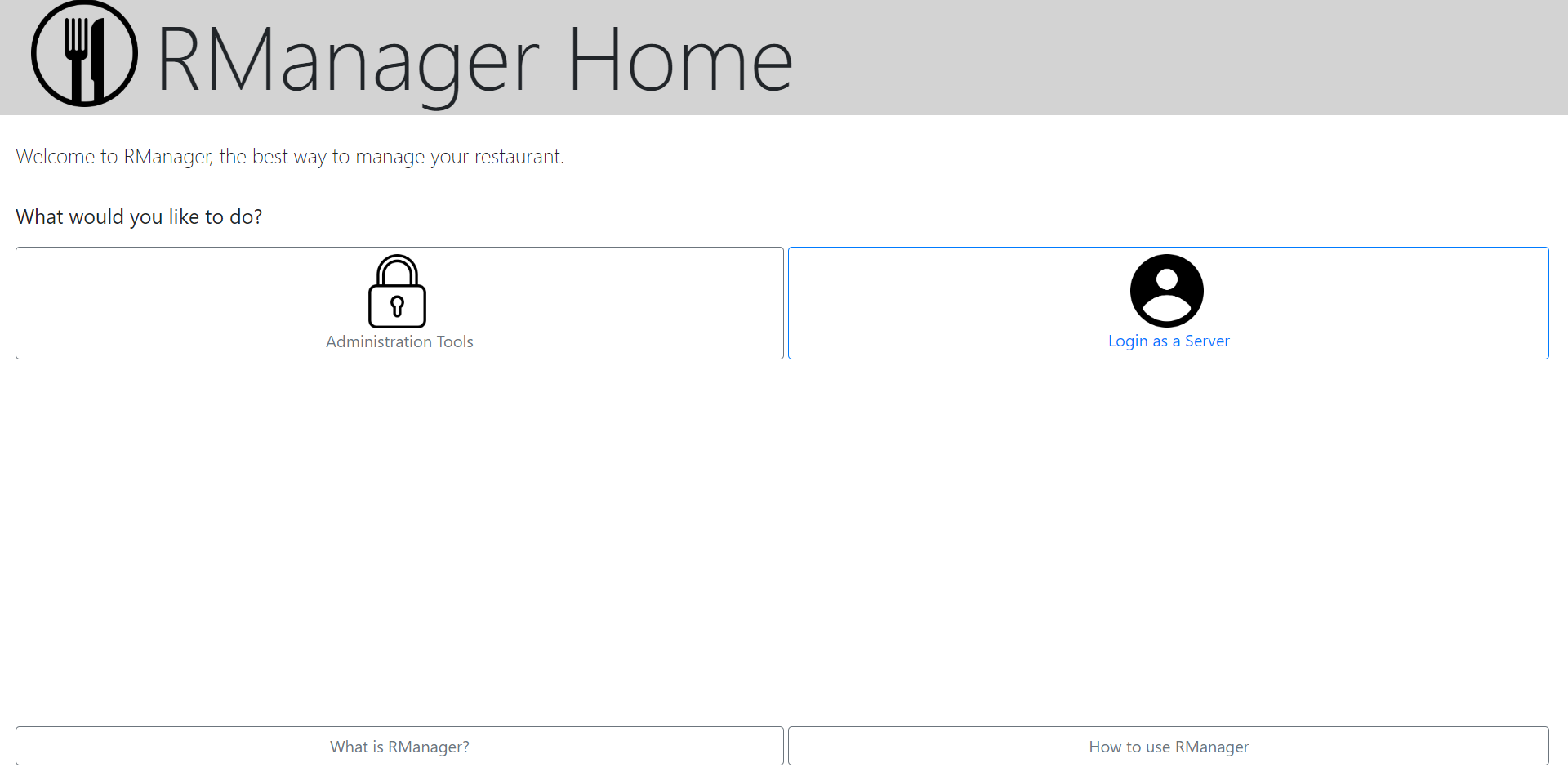
1. Create new bookings
2. View previously made bookings
3. Edit previously made bookings (only certain parameters supported)
4. Delete previously made bookings
5. Download a .txt file of previously made bookings based on time/date
6. Query bookings based on name
7. View all bookings in local storage (via admin tools)
8. Clear all local storage (via admin tools)

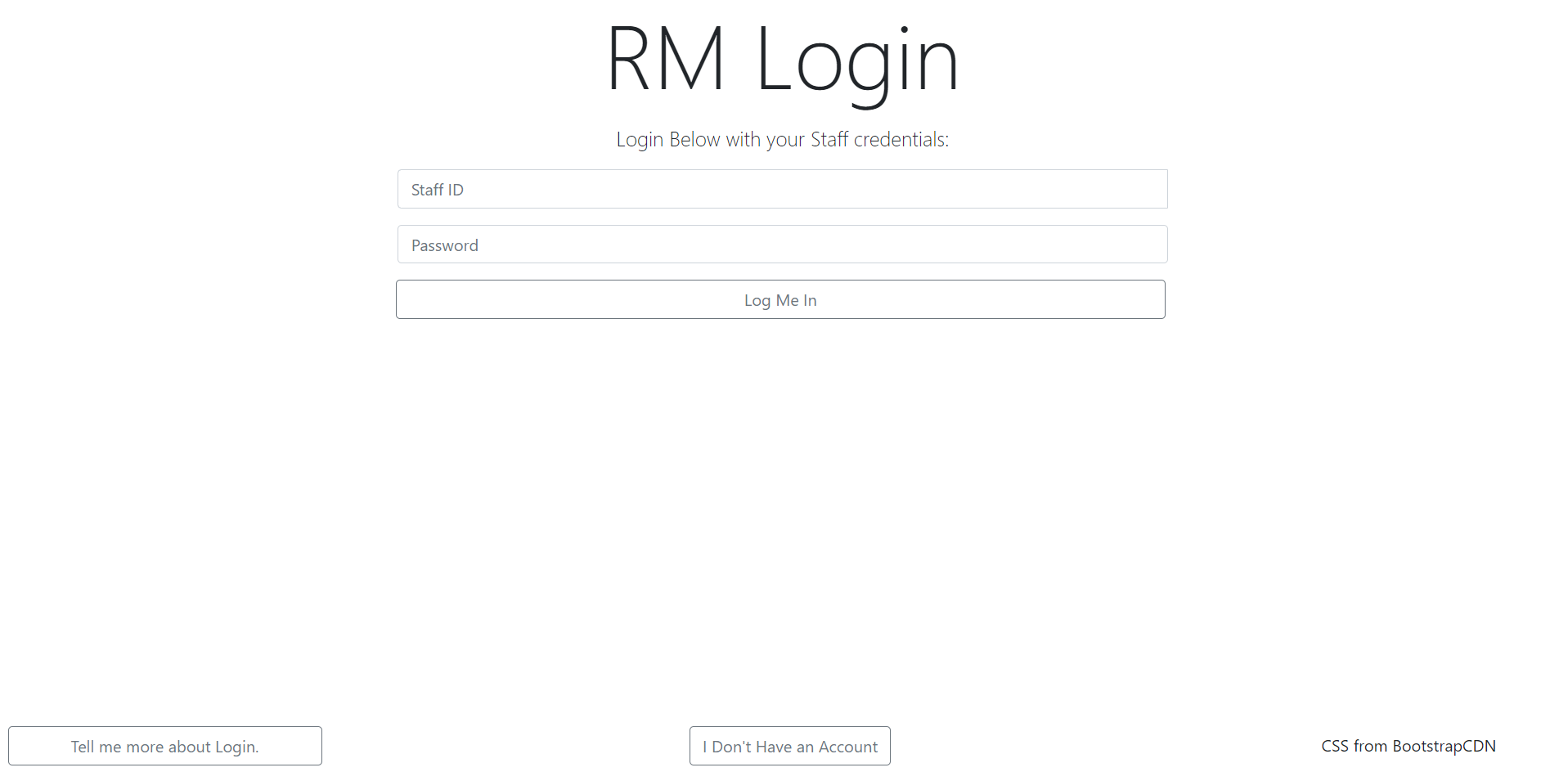
**Future Development Goals**

RManager is only partially through its development, and thus has several future development goals. Some of these include (in order):

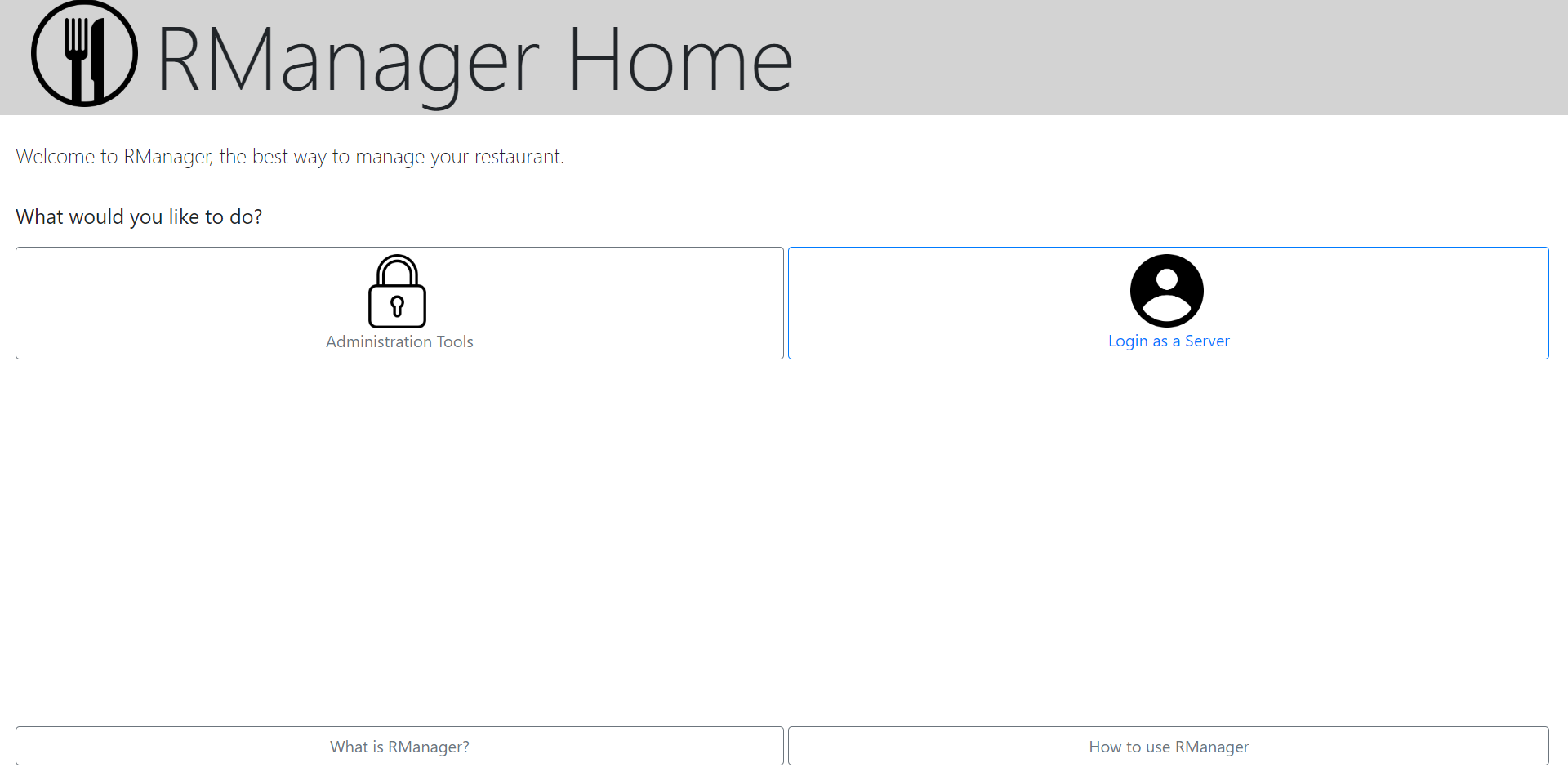
1. Completing modules for allocating tables
2. Allowing administrators more tools for managing staff logins
3. Implementing a more dynamic user interface
   1. **Logging in and Account Creation**

RManager allows for the creation and use of user/staff accounts. These accounts increase the security of the system by reducing the likelihood of an unauthorised person gaining access.

*To create an account*, select ‘log on as a server’

Then click ‘I don’t have an account’ at the bottom of the webpage

This will take you to a page where you can create a user account. The page will also let you know if the username is already taken, and will also ensure that data is entered in all fields.

*To log in*, press on ‘log on as a server’

Then, enter your login details in the fields provided.

* 1. **Make a New Booking**

To make a new booking, from the home screen (main.html):

1. Press ‘Create Booking’ (top right)
2. Enter the relevant details such as booking name, date, time and location.
3. Press ‘Submit Booking’
4. Assuming that all fields are filled, a confirmation dialogue will appear. If the data is correct, press ‘Ok’.
5. The booking will then have been created.
   1. **Edit an Existing Booking**

To edit an existing booking, from the home screen (main.html):

1. Press ‘Manage Bookings’ (top right)
2. *To view the details for a booking:*
   1. Type in the booking name at the top of the screen.
   2. Press ‘Query Records’
   3. The relevant details will then be available for viewing.
   4. *If you would then like to edit the booking:*
      1. Press ‘Edit this Booking’
      2. Enter the relevant information into the popup window.
      3. Press ‘Submit Booking’ and confirm the details with the dialogue box.

***Please note:*** *only certain details of the bookings are available for changing post-booking.  
To change any other details, please delete the booking and create a new one.*

**External Libraries**

Restaurant Manager uses multiple libraries to increase functionality and decrease the size of the program.   
All external libraries are accessed via a Content Delivery Network (CDN). This increases load time due to the fact that the libraries may already be cached on the machine. Additionally, the use of a CDN decreases the complexity of install compared to NPM or Yarn.

1. **JQuery**

RManager uses JQuery in specific situations to increase DOM manipulation and CSS animation functionality. JQuery allows for easy access of DOM components using the $ sign:

$('#successdiv').show()

1. **Bootstrap** (styles)

RManager uses Bootstrap for the vast majority of CSS styling and formatting. Bootstrap allows for easy, simplistic and clean styling to be applied to HTML components through the use of a class identifier.

1. **SweetAlert**

RManager uses SweetAlert to generate clean, interactive and responsive alert and prompt dialogues. These replace the native JS alerts which are not intuitive or aesthetically pleasing.

SweetAlert is called using the global swal variable and utilises an object-based approach in alert creation, with each ‘parameter’ of the alert contained within an object:

swal({

title: 'What is RManager?',

text: 'RManager is a tool for restaurants to create, manage and manipulate bookings. \n To find out more, visit the docs.',

icon: 'info',

})

Additionally, SweetAlert uses arrow functions for responsive alert functionality:

swal({

title: "Are you sure you want to do that?",

text: "Doing so will delete all usernames, passwords, bookings and other data.",

icon: "warning",

buttons: true,

dangermode: true}

)

.then((value) => {

// Code to be executed goes here…

});