

Table of Contents

RIGET ZOO ADVENTURES – TASK 1 – DESIGN DOCUMENT	2
DIAGRAMS	3
<i>Use Case Diagram</i>	3
<i>Hierarchy Diagram</i>	5
WIREFRAMES	8
<i>Style Guide</i>	8
Colour Scheme	8
Typography.....	10
Iconography	12
Components and States	12
Dimensions.....	14
<i>Base</i>	17
Logged Out	17
Logged In	18
Logged In - Interact.....	19
Explore Dropdown - Interact	20
CONSTANT	22
Home Page	22
Sign up Page.....	23
Login Page.....	24
<i>User</i>	25
Bookings – Room.....	25
Bookings – Tickets	27
<i>Administrator</i>	30
Bookings – Room.....	30
Bookings – Tickets	31
ALGORITHMS / FLOWCHARTS	34
<i>Register System</i>	34
<i>Login System</i>	36
<i>Shopping Cart - Ticket</i>	38
<i>Shopping Cart - Rooms</i>	40
<i>Booking System</i>	42
PROJECT DATA.....	43
<i>Users Table</i>	45
<i>Roles Table</i>	47
<i>UserRoles Table</i>	48
<i>Orders Table</i>	48
<i>OrdersDiscounts Table</i>	49
<i>Discounts Table</i>	49
<i>OrdersItems Table</i>	50
<i>TicketOrders Table</i>	51
<i>Tickets Table</i>	52
<i>RoomOrders Table</i>	52
<i>Rooms Table</i>	53
PROJECT TESTING.....	54

Riget Zoo Adventures – Task 1 – Design Document

This document contains the various diagrams that will be used for the implementation of the proposed application. These diagrams are to be used as visual aids for the development of the application.

At the top of the document contains the Use-Case Diagram and the hierarchy diagram, which provide an overview of the applications functions and architecture.

Afterwards are the various wireframes that have been designed for this application, within the Style Guide and its subheadings contains the relevant information about the designs colour scheme, typography, iconography, components and states. All to be used to make sure the application design stays consistent.

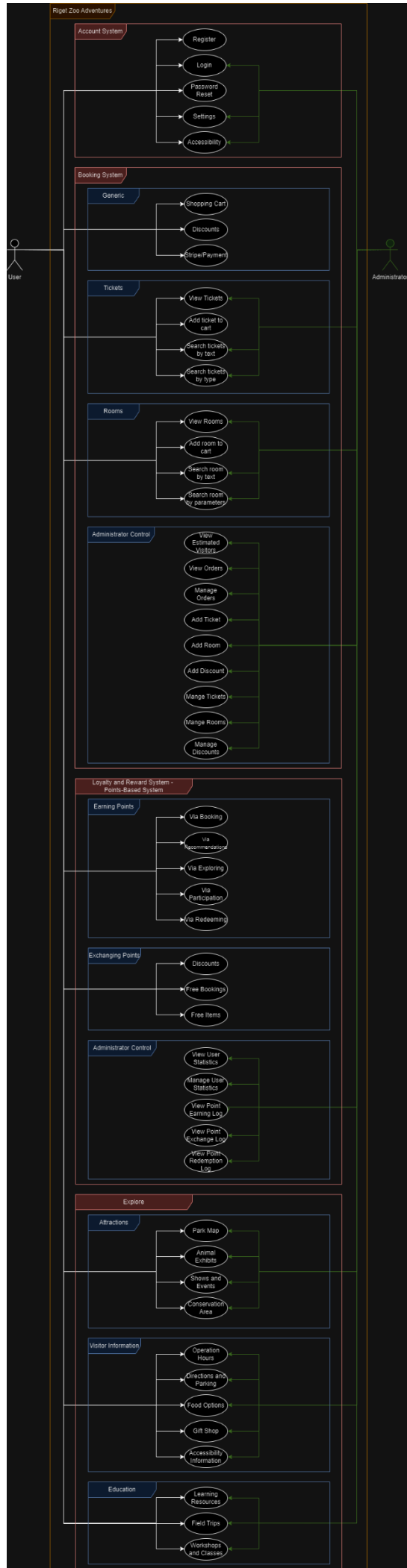
Within the Wireframes it contains several subheadings split between `Base`, `Constant`, `User` and `Administrator`. Each subheading shows web pages that are either consistent throughout no matter which user is logged in, or web pages that may have slight variations depending on the individuals' role and have as such been put into both the `Uses` and the `Administrator` headings.

Lastly are the algorithms that will be implemented within the application itself, to easily and visually show off these algorithms they have been represented as flowcharts for ease of understanding.

Diagrams

In the following sub sections are the diagrams related to how the application is to flow when implemented, it will showcase how the application is hierarchically structured along with how different roles use the application.

Use Case Diagram



Above is the use case diagram, it contains the visual representation of users and administrators interacting with the system and its various components and features that are implemented within.

In the diagram above the white individual is used as a representative of a standard user that one would be when signing up to/using the application. The user has full access to the account system allowing them to register, login and manage their account. Within the booking systems the user has restricted access to it, within the sub sections of, `Generic`, `Tickets` and `Rooms` the user has full access to these sections features allowing them to have access to adding these items to their shopping carts and paying for their cart as required, however they do not have access to any of the controls or features in the `Administrator Control` sub section, as within this section it contains various features and data a normal user should not have access to as it contains internal and sensitive information.

Within the loyalty and reward systems it has restricted access to it once more, within the sub sections of, `Earning Points` and `Exchanging Points` the user has full access to these sections features allowing them to have access to the ability to earn points via several different methods while also having access to exchanging the points they earn for free rewards/items. However, they do not have access to any of the controls or features in the `Administrator Control` sub section, as within this section it contains various features and data a normal user should not have access to as it contains internal and sensitive information.

Within the explore systems the user has full access to all the features within, allowing the user to explore all the content within the application, including the content with the `Attractions` sub section, `Visitor Information` sub section and the `Education` sub section, they have full access to this section.

In the diagram above the green individual is used as a representative of an administrator user. The administrator has restricted access to the account system, as they cannot register new administrator accounts nor can they reset their password as these are potentially security problems, as creating new administrators is risky while resetting a password should never occur unless somebody is attempting to reset the password to gain access to the account. Within the booking system the administrator has restricted access to it similarly to a standard user, but it is restricted in a different manor, as the administrator has no use of the shopping cart and for purchasing of items it as such does not have any access to the features in the `Generic` sub section as these are related to the purchasing of items, and within the `Tickets` and `Rooms` sub section it can do everything a standard user can, except for adding the item to the shopping cart as the administrator does not have one. However, the administrator has full access to the sub section `Administrator Control` and its features within, these features allow the administrator to view data relating to the booking system alongside having access to the creation and management of content related to this system such as tickets, rooms and discounts, the administrator can easily add and manage any of these existing data.

Within the loyalty and reward systems the administrator has restricted access to it once more, as the administrator has no use for earning points or redeeming them, as such the administrator does not have access to these features. However, the administrator has full access to the sub section `Administrator Control` and its features within, these features allow the administrator to view relevant information about the loyalty system such as user statistics or the logs of the system, it also gives the administrator access to managing individual user accounts as required to do so.

Within the explore systems the administrator has full access to all the features within, allowing the user to access all the content within such as the `Attractions` sub section, `Visitor Information` sub section and the `Education` sub section, they have full access to this section.

Hierarchy Diagram



Above is the hierarchy diagram, where it showcases how the entire application branches off from the singular login action from the home page when the user clicks to login. To simplify the diagram above any content that is accessible with no restrictions from both user roles has been expunged to make the diagram easier to read and understand, this means that the `Explore` section and its sub sections was not added as both the User and Administrator has absolute access to all the features implemented within.

Within the booking system, the User has access to all the generic options that setup the shopping cart and allow for payments, while the also has further access in the Tickets and Rooms sub sections that give it access to all its features, including the ability to add an item to the users shopping cart. As for the administrator, it has been restricted as it does not have any access to the generic section as it does not have any shopping cart, while it does have access to view the tickets and rooms it doesn't have access to add it to the shopping cart like the user does, however it does have access to features the user don't, this specifically being the `Administrator Control` section and its features, this section allows the administrator near full control over the booking system as it allows them to view orders and manage orders that have been made with the application, along with adding and managing additional tickets, rooms and discount codes as required.

Within the loyalty and reward system, the User has full access to the ability to earn points via several different methods such as booking, recommendations, exploring, etc. The user also has access to the features of exchanging their points for rewards such as discount codes, free items and even free

bookings. As for the administrator, it has been restricted as it does not have any access to the features of earning points via any method, nor exchanging points for any rewards. However, the administrator has access to the `Administrator Control` section and features, this allows the administrator to view and manage users statistics relating to the loyalty and reward system, while also having the ability to view the logs/reports about how points or being earned, exchanged and what codes are being redeemed.

Wireframes

Below are the wireframes that have been designed for this application, to allow for ease of understanding they have been split into relevant subsections.

Within the `Style Guide` subsection it contains various information related to how the application has been designed such as the Colour scheme used, the Typography used, Iconography used amongst other related information.

Within the `Base` subsection it contains the layout of the website's headers and footer elements, which remain unchanged throughout the design. It also contains the dropdown of the explore option to show all other available pages on the site.

Within the `Constant` subsection it contains information and designs that are constant throughout the application as these pages have no modifications depending on the user's role.

Within the `User` and `Administrator` sub sections it contains role specific content related to these roles, the designs between headings that line up are meant to look similar but for one role to have further features such as the administrator having access to buttons to control application content/settings.

Style Guide

Below are the various style guides that was used during the design process of this application, it contains various information related to how the application has been designed such as the Colour scheme used, the Typography used, Iconography along with how the Components look alongside their states.

Colour Scheme

The colour scheme below contains the various colours used to design the application, this is done so the colour throughout the design stays consistent and in style with it. Due to not receiving the clients current branding scheme colours we have temporarily chosen our own colours we believe are appropriate until the final colours can be received from the client, and then inputted into the design itself.

To choose these colours an automated tool was used to cycle between various different random colours until a good base colour scheme was chosen, from which it was further fine-tuned to get the colours shown below, the tool used is known as `Real Time Colours`.

Colours - Light Mode

Neutral



Branding

Primary



Primary,
#FF9514

Secondary



Secondary
#DEBB91

Accent

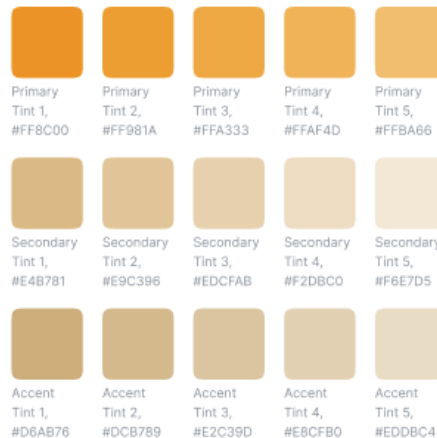


Accent
#E89F45

Color Shade



Color Tint



Action



Typography

The typography covers the how the text looks like within the application and its various locations such as headings, H1/H2/H3 text, body text, image captions, amongst other locations. The font chosen for the design is `Inter` due to the text looking good, clean and modern. Within the typography listed below it will specify the parameters for each type of text alongside where this content would be used i.e. Headers, Sub-Headers, Body, etc.

Typography		
Typography		
Headline/ super-large	Example:	Headline/super-large
Font: Inter Size: 96 Height: 128 Spacing: +1.5% Weight: Bold		
Headline/ extra-large	Example:	Headline/extra-large
Font: Inter Size: 60 Height: 84 Spacing: +0.5% Weight: Regular		
Headline/ large	Example:	Headline/large
Font: Inter Size: 48 Height: 64 Spacing: 0% Weight: Regular		
Headline/ medium	Example:	Headline/medium
Font: Inter Size: 34 Height: 48 Spacing: 0.25% Weight: Regular		
Headline/ small	Example:	Headline/small
Font: Inter Size: 24 Height: 36 Spacing: 0% Weight: Regular		
Headline/ extra-small	Example:	Headline/extra-small
Font: Inter Size: 20 Height: 30 Spacing: 0.15% Weight: Medium		
Subtitle/ base	Example:	Subtitle/base
Font: Inter Size: 16 Height: 24 Spacing: 0.15% Weight: Regular		
Subtitle/ secondary	Example:	Subtitle/secondary
Font: Inter Size: 14 Height: 20 Spacing: 0.1% Weight: Regular		
Body/ base	Example:	Body/base
Font: Inter Size: 16 Height: 24 Spacing: 0.5% Weight: Regular		
Body/ secondary	Example:	Body/secondary
Font: Inter Size: 14 Height: 20 Spacing: 0.1% Weight: Regular		
Other/ button	Example:	Other/button
Font: Inter Size: 14 Height: 20 Spacing: 1.25% Weight: Medium		
Other/ caption	Example:	Other/caption
Font: Inter Size: 12 Height: 18 Spacing: 0.4% Weight: Regular		
Other/ small	Example:	Other/small
Font: Inter Size: 10 Height: 14 Spacing: 1.5% Weight: Regular		
Other/ very-small	Example:	Other/very-small
Font: Inter Size: 8 Height: 14 Spacing: 1.5% Weight: Regular		

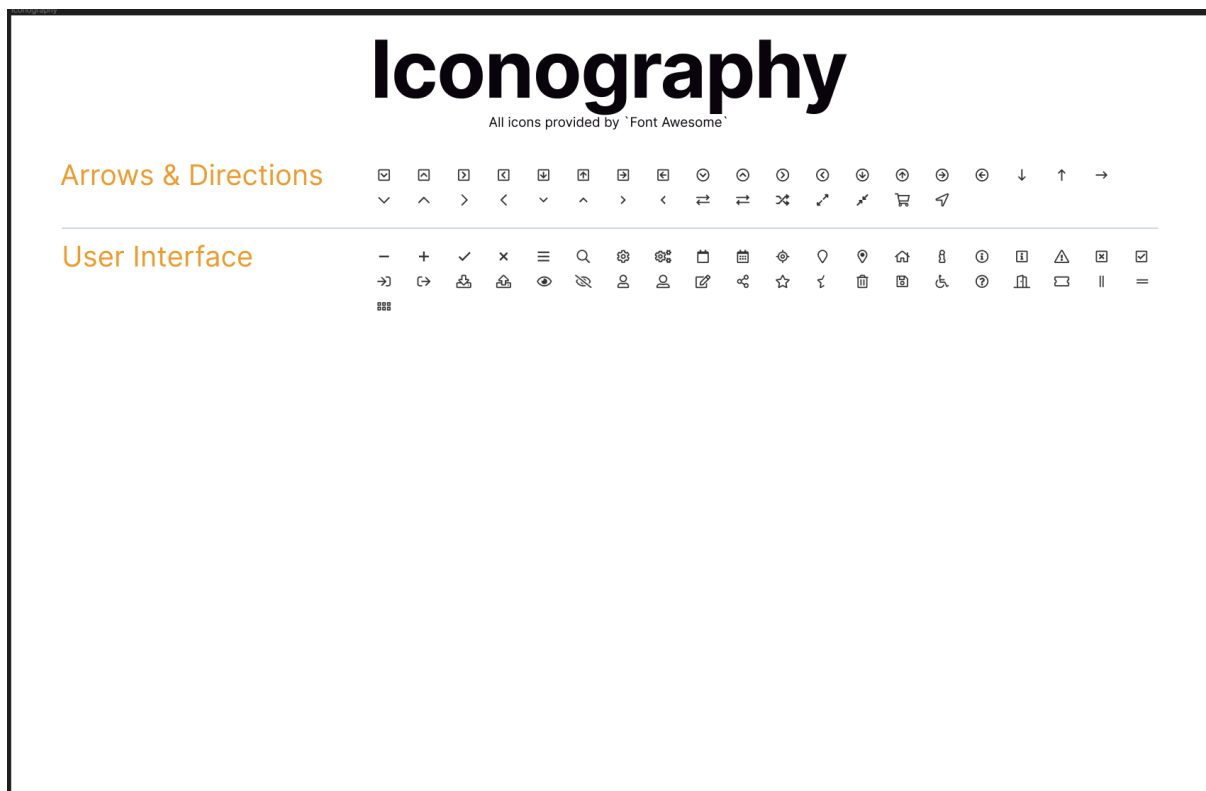
Iconography

The iconography contains the various icons that are to be used during the design process, this is done to keep the icons used throughout the design consistent and accurate. Within the iconography it contains two main sections them being `Arrows & Directions` and `

Within the arrows section it contains various pointing icons used to showcase how items are to work and is mainly used for subtle hints throughout the applications design.

Whereas within the user interface it contains various icons that are used throughout the application in various locations to give a visual representation to text to make it much easier for a user to find what they are looking for, as they need to simply search for an icon instead of reading text, making the user interface look much cleaner and the user experience much better.

In the iconography below, all the icons have been provided by `Font Awesome`, as such proper attribution must be given to them on the legality page or a page that is acceptable for such attribution within the application.



Components and States

The components and states showcase how the various components such as buttons and links look like within the application, this is done to ensure that the components stay consistent and clean throughout the application design.

Alongside this it also shows each component in various different interaction states, being the default state that the component normally looks like, the hover state when a cursor hovers over the component, the focused state when the component is focused, the click state when the user clicks the component and the disabled state when the components is disabled/not intractable.














































In the images below it shows 3 variations of the button and the links within the specified colour, there are two images the first image is for components that are shaded the primary colour while the second image is for components that are shaded the secondary colour. See the `Colour Scheme` for these colours.

Each variation of the button is slightly different, the first button is just plain text, the second button has an icon on the left whereas the third button shows an icon on the right.

Components And States - Primary

Components and States

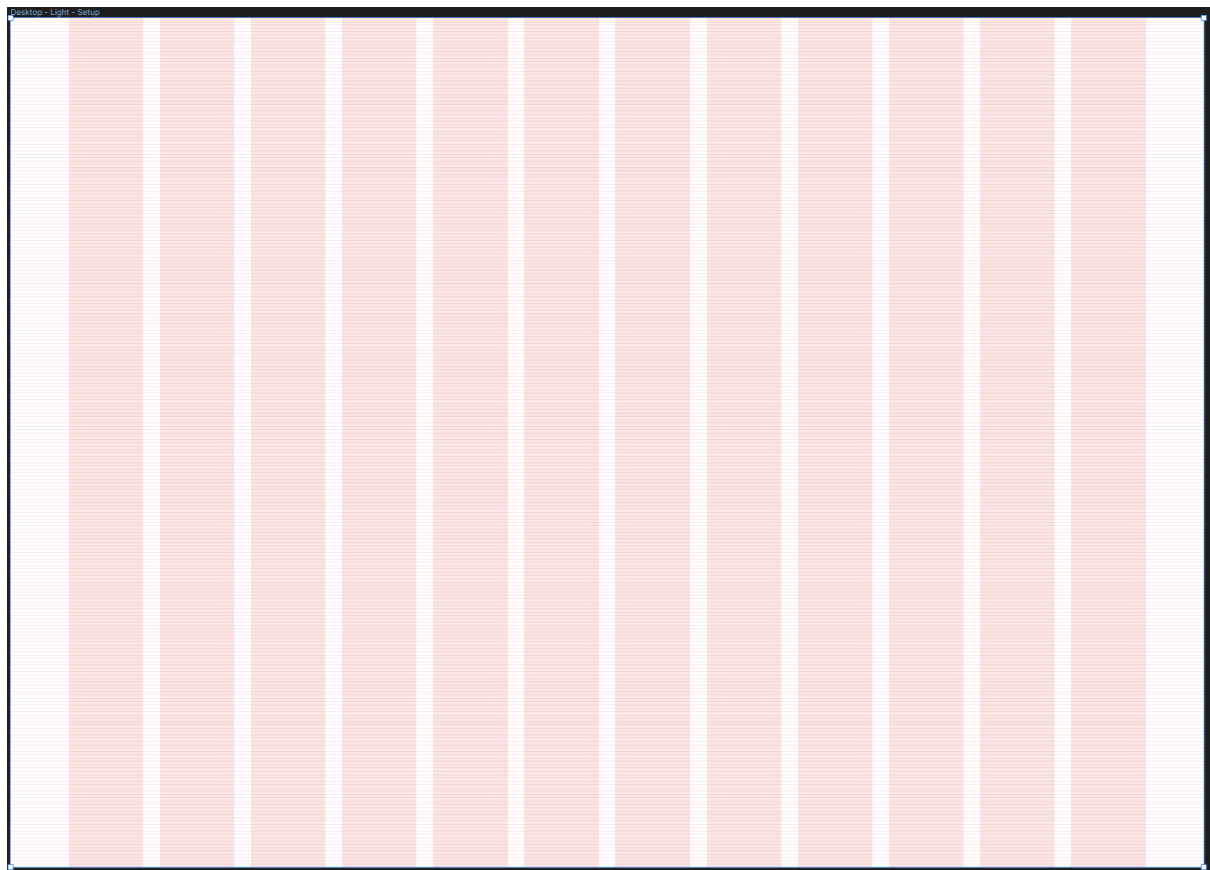
Primary

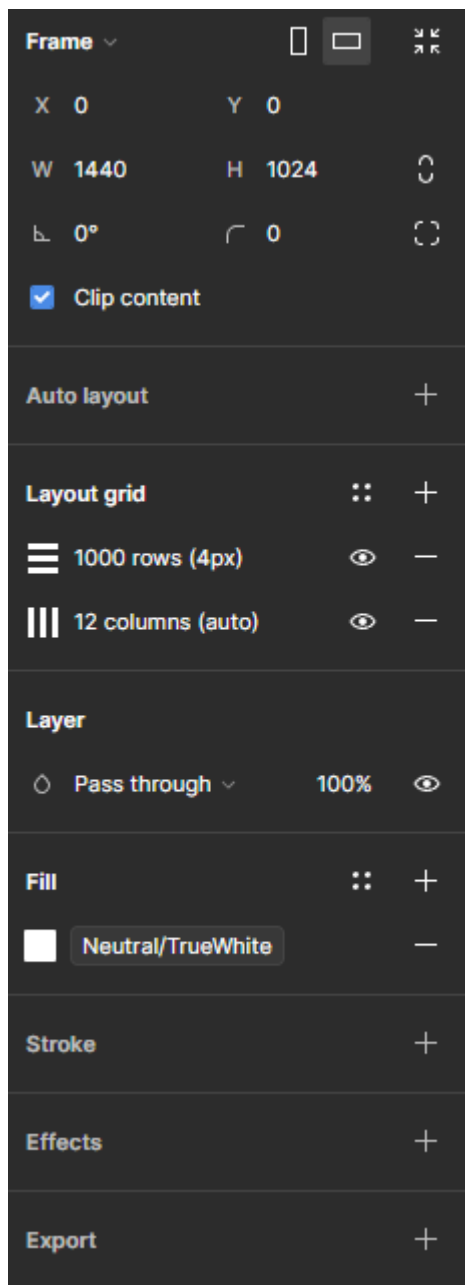
	Default	Hover	Focused	Active	Disabled
Buttons	  	  	  	  	  
Left Icon Buttons	  	  	  	  	  
Right Icon Buttons	  	  	  	  	  
Links	Link Link Link	Link Link Link	Link Link Link	Link Link Link	Link Link Link

Components And States - Secondary

	Default	Hover	Focused	Active	Disabled
Buttons	<div>Button</div> <div>Button</div> <div>Button</div>	<div>Button</div> <div>Button</div> <div>Button</div>	<div>Button</div> <div>Button</div> <div>Button</div>	<div>Button</div> <div>Button</div> <div>Button</div>	<div>Button</div> <div>Button</div> <div>Button</div>
Left Icon Buttons	<div>+ Button</div> <div>+ Button</div> <div>+ Button</div>	<div>+ Button</div> <div>Button</div> <div>Button</div>	<div>+ Button</div> <div>+ Button</div> <div>+ Button</div>	<div>+ Button</div> <div>+ Button</div> <div>+ Button</div>	<div>+ Button</div> <div>+ Button</div> <div>+ Button</div>
Right Icon Buttons	<div>Button +</div> <div>Button +</div> <div>Button +</div>	<div>Button +</div> <div>Button +</div> <div>Button +</div>	<div>Button +</div> <div>Button +</div> <div>Button +</div>	<div>Button +</div> <div>Button +</div> <div>Button +</div>	<div>Button +</div> <div>Button +</div> <div>Button +</div>
Links	<div>Link</div> <div>Link</div> <div>Link</div>	<div>Link</div> <div>Link</div> <div>Link</div>	<div>Link</div> <div>Link</div> <div>Link</div>	<div>Link</div> <div>Link</div> <div>Link</div>	<div>Link</div> <div>Link</div> <div>Link</div>

Dimensions



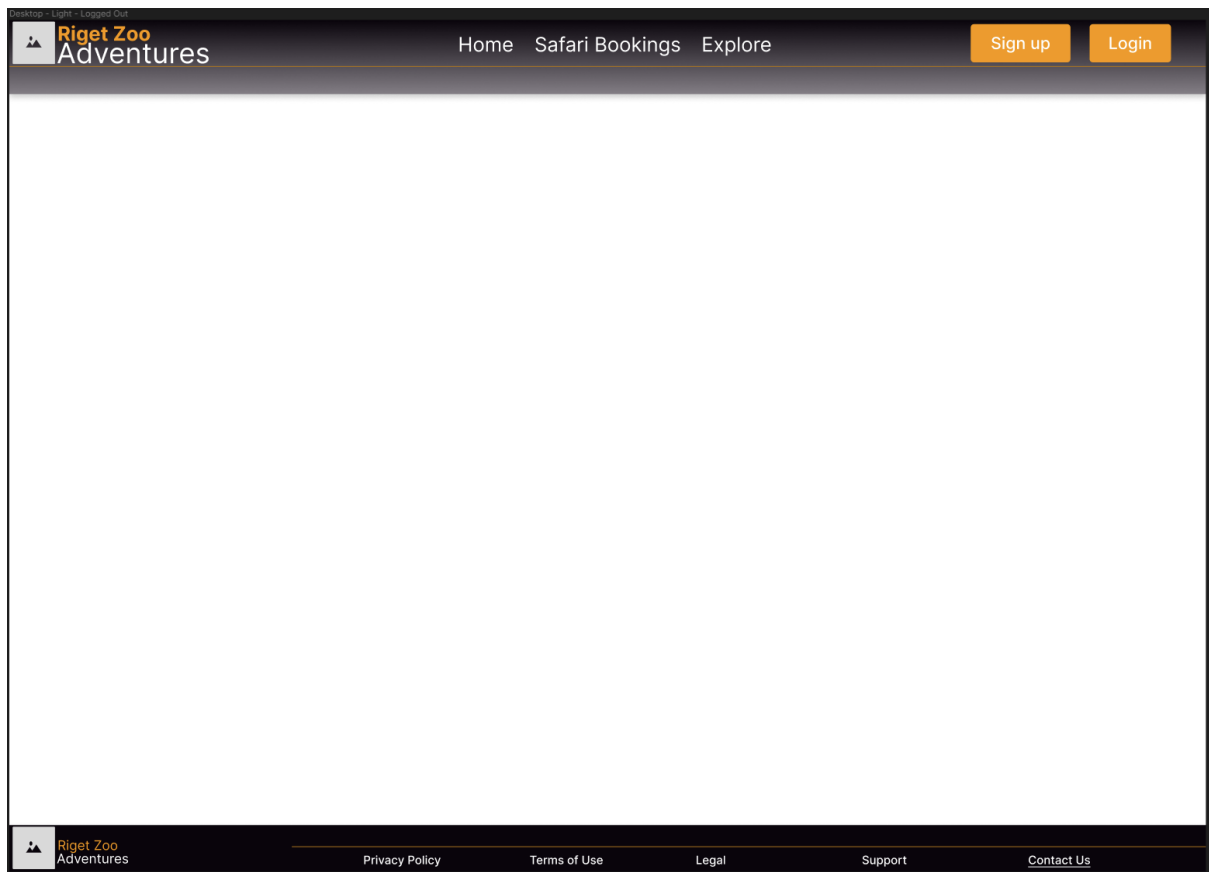


Above are the dimensions that have been used when designing the application, the frame is 1440 units wide and 1024 units high. This sizing scheme was chosen due to it being the standard development size of desktop applications due to it allowing for wide adaptability and efficiency. Within the image listed above there are red vertical columns and many very small red vertical rows, these are to be used as a guide scheme when designing the application to ensure that everything is laid out properly with accurate spacing between elements, it also ensure that content are placed in positions that make it easier for an individual to read due to the placing of the elements.

Base

Below is the base UI layout for the application, it will contain designs and explanations on how the header and footer look and why they do so. Alongside this it will contain header variations of when a user is logged in and interacting with their account dropdown, among other things. In the designs the majority of the page will be white/have no content, as this is where each page's content will load and as such not been filled in as these designs are meant to only showcase the header and footer elements.

Logged Out



Above is the applications layout for the header and footer elements when the user is not logged into the application.

Within the header on the right-hand side is where the client's image will go alongside the applications name of `Riget Zoo Adventures`, this was put here to showcase off who the company is alongside the name.

Within the middle of the header is the main navigation options of the site, the first button is the `Home` button that upon clicking will send the user to the home page, the next button is the `Safari Bookings` button that upon clicking will send the bookings system/page with the default focus set to rooms to encourage users to book a hotel stay/room instead of a ticket, the final navigation button is the `Explore` button, however this does not link the user anywhere as it is opens a dropdown upon being hovered over/clicked, within this dropdown it will showcase the various content within the application such as the Attractions the zoo contains, any information a visitor may need alongside information related

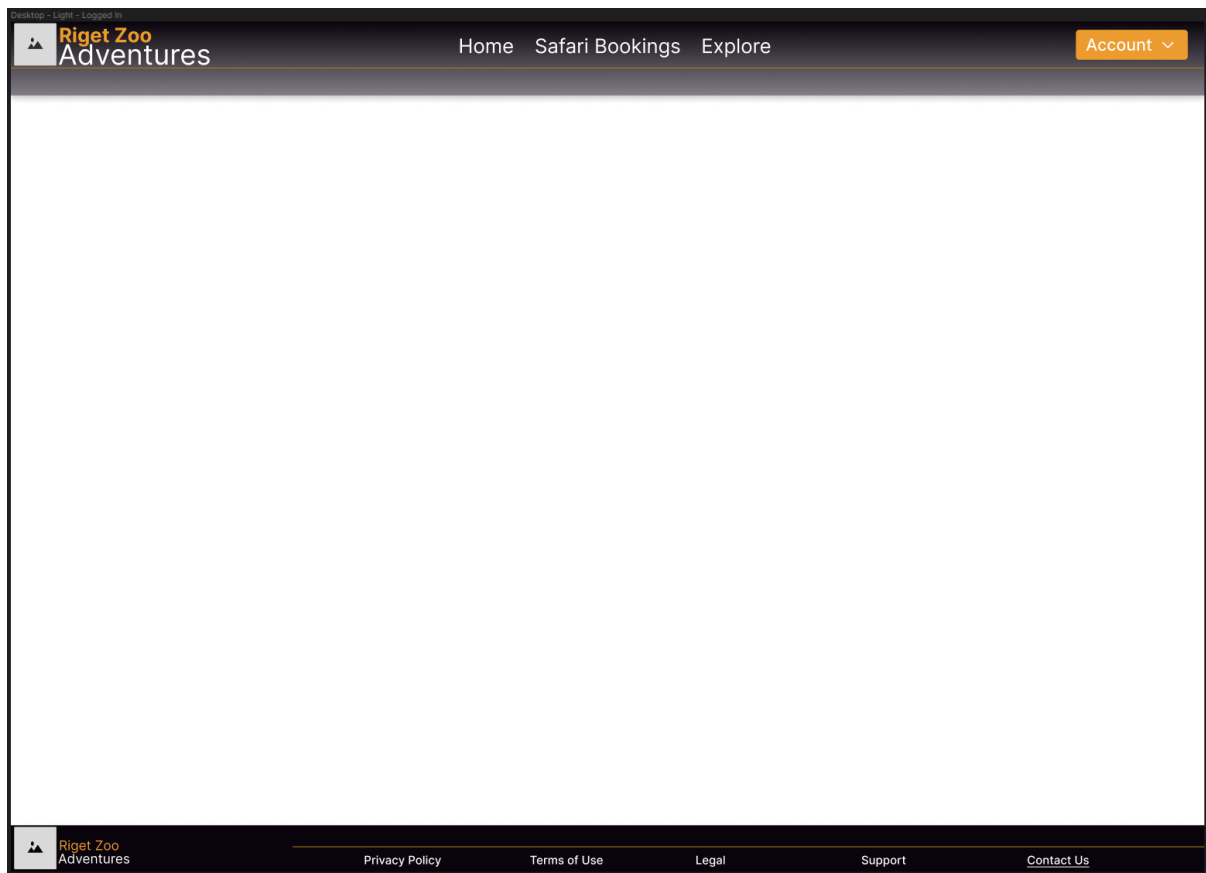
educational resources, to see this dropdown please see the heading `Base/Explore Dropdown – Interact`.

On the right-hand side of the header is the `Sign up` and the `Login` buttons, upon clicking will send the user to the respective pages so they could either signup for a new account with the application, or login to a pre-existing account that they have already made with the application.

As for the background of the header it is a gradient from pure black at the top to white/transparent at the bottom, this was done so that it can look like its integrating with the page itself, although it is not shown in this design when this is put together with a top image it makes the header standout and look integrated with the entire application, making it look clean and modern. To see an example of this see any page with content, such as the `Constant/Home Page`.

At the bottom of the design is the pages footer, on the left hand side it again contains the company to further reinforce the company that operates this application alongside who it is, and to the right of this are various links are user can click on to see information relating to how the application operates, such as the applications Privacy policy, the Terms of use, the legality page that explains the laws we follow alongside any attributions, the next option is to open a support page to get help with the application from the administrators/staff, and finally the contact us button allows the user to directly email the clients customer service address for any enquires that they may have.

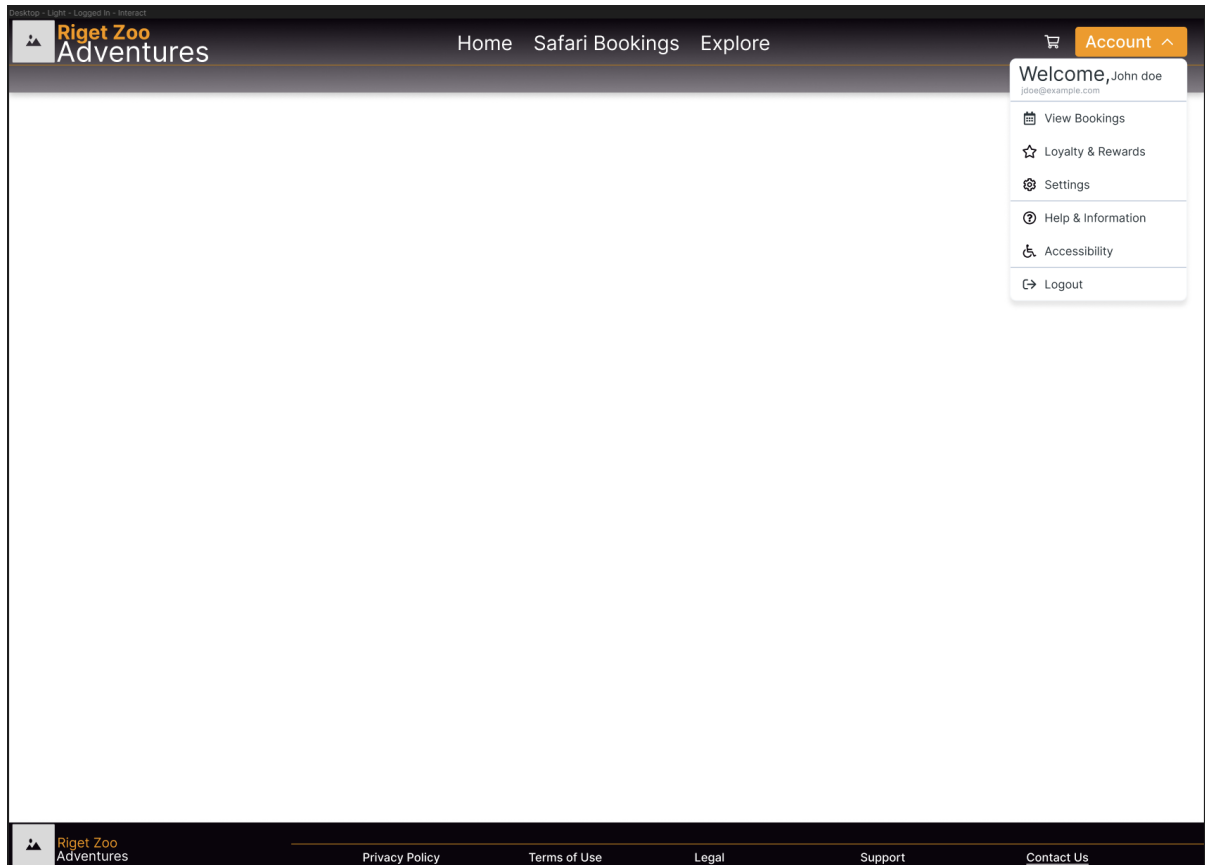
Logged In



In the design above, the layout of the footer remains consistent with the previous design, whereas the header changes slightly. Although the header still maintains the branding on the left and navigation in the centre, on the right instead of having options to sign up or login it contains a button, that upon clicking will

showcase a dropdown related to the users account, this has been shown by putting a down arrow to showcase something opens upon being clicked.

Logged In - Interact



In the design above the layout of the header and footer remain consistent, the only change is that it now showcases the account dropdown menu, alongside changing the down arrow to an up arrow to showcase that clicking it again will close the dropdown menu. The dropdown menu is split into four different sections.

The first section welcomes the user by name to make the application feel personalized to them, making the user feel appreciated and that they are cared about within the application.

Below this are the buttons for managing the users account and its corresponding information, the first button will direct the user to the page for managing their accounts booking they have done in the past, allowing them to view previous bookings along with managing current/upcoming bookings in the future. The next button directs the user to the Loyalty & Rewards page, which is a system to encourage the user to keep returning so they can earn points and exchange them for items, and as such has been put here to make this system easy to access. Lastly is the button to manage the accounts settings if the user requires, in which they can manage many points of information about the account, so as their name, email address, password, phone number, etc.

In the next section are the buttons related to support and help a user may need while using the application, the first button redirects the user to the support page that the footer also links to, which allows the user to receive support from the administrators/staff by asking questions. After this is the accessibility options so that individuals that may require these options can easily access them and enable the various accessibility options as are required.

In the last section there is only one button which is the logout button, which upon clicking logs the user out of their account as such requiring them to login again to continue using the application with this specific account.

Explore Dropdown - Interact



In the design above the layout of the header and footer remaining consistent with the `Logged in` layout, however there is now a dropdown option below the explore navigation option, and within contains the various content that is available in the application.

The first section of the explore dropdown is `Attractions` section, this section lists out the various attractions and facilities that are operating on the zoos premises, the first option directs the user to an interactable zoo map, within will showcase an image of the zoo alongside various different pins of information showcasing where the various attractions of the zoo are located, such as where the animal exhibits are, where the food options are, where the gift shop is alongside general information such as where toiletry is available amongst other information. Below this is the animal exhibits button, which will redirect the user to a page that lists all the animal exhibits that are within the zoo so the user can easily research what animal attractions are in the zoo. Below this is the shows and events table, which will functional similarity to the animal exhibits button, but instead of showing information about the animals within the zoo it will show information about what shows and events are going on in the zoo.

The next section is the `Visitor Information` sections, it contains various pieces of information that a visitor may need when planning their trip to the zoo, the first button would direct the user to a page explaining the operation hours of the various facilities within the zoo such as the zoo itself, the hotel, food options, etc so the user can easily plan. The next button would direct the visitor to a page explaining how

to get to the zoo itself alongside showing what parking is available on/near the zoo premises. The next option will direct the user to a list of food options on the zoo premises such as restaurants or food vendors. The gift shop option will showcase information relating to the zoos gift shop such as where it is and what you could purchase, it will also advertise the loyalty and reward system as you can exchange free merchandise through this system. Lastly it will give the visitor information relating to accessibility considerations within the zoo, so that individuals with disabilities can still freely access and visit the zoo no matter their own personal circumstances.

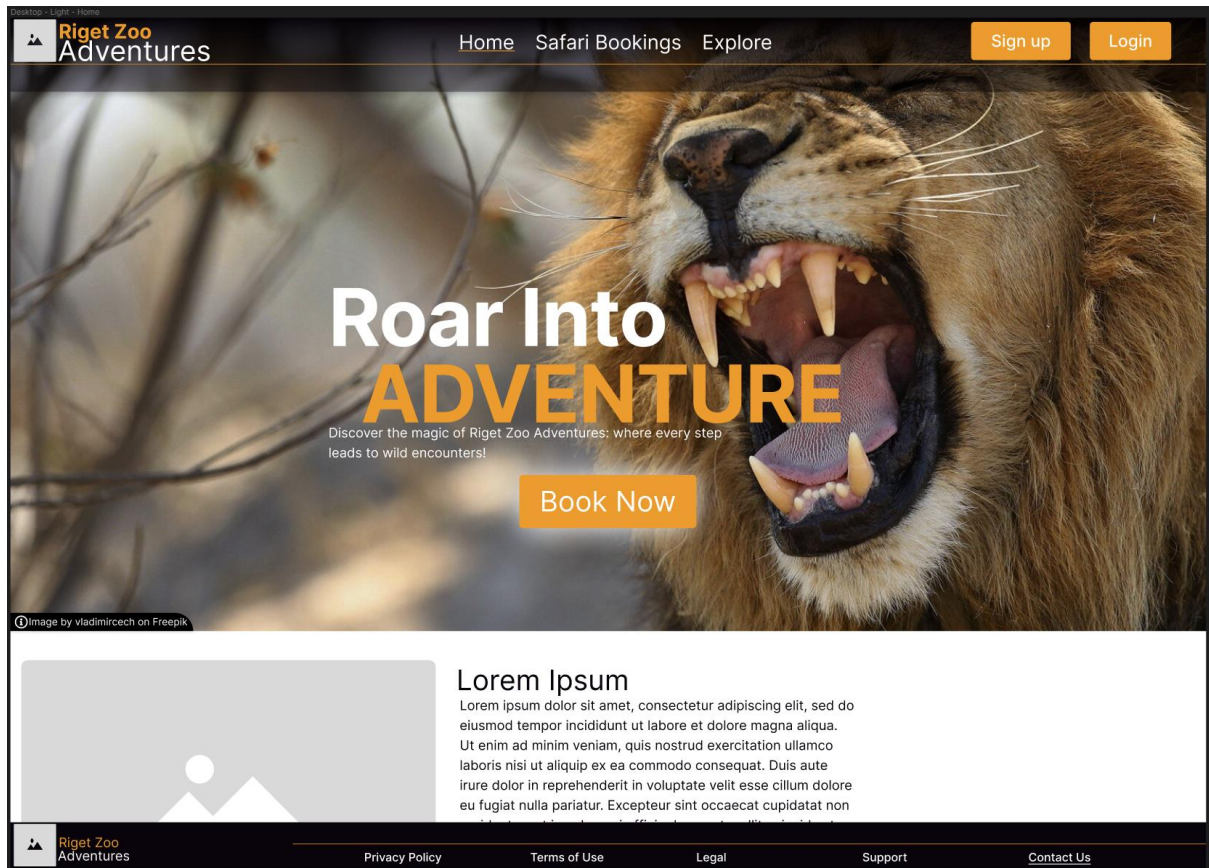
The last section is the `Education` section, this section contains various information related to educational resources that a visitor or the zoo may need to support their knowledge on the animals within the zoo. The first option is the learning resources, it will provide many resources within the application about the animals while also giving external sources of information for further support as required. The next option explains how field trips/educational tours can occur within the zoo, it will explain what would occur, how many people can come and what the price is alongside giving the user the method of contacting the zoo in case they wish to book a trip for their organisation, such as a school. The last button contains information about any workshops and classes that are occurring in the zoo, it will also state how any third-party organisations can contact the zoo in the case they wish to host their own.

In the dropdown above, each text has an icon of a question mark, this icon is used to showcase that this is information, however these could be changes to more specific icons to make it easier to understand, such as a map or a pin for the zoo map.

Constant

The designs listed below are constant throughout the application and does not have any modifications done to them depending on the user's role and are constant throughout the application.

Home Page



The home page is the first page an individual sees when visiting the site, as such it must be well designed and eye catching to the user to make it more likely they continue to use the application, to do so a big image is at the top of the content to gain the users attention due to the colours and for them to understand the image, and within the centre of the image is a slogan to make the user feel interested by the application alongside, below this is a call to action button that encourages the user to `Book Now` and to make it more noticeable it also has a subtle white background blur to make it pop out more, all of this makes it much more likely for the user to click this button and then be taken to the booking pages, increasing the chances of a user booking a ticket/hotel stay.

Below this will be several different images and explanations about the zoo to showcase what it offers, and to convince the individual to book. It is designed to be laid out, so an image is on the left side with text on the right, then flipping to the image on the right and text on the left and for each subsequent point to continue to do so, as it makes the content more visually appealing that way than having a list of items.

Alongside this an underlining of the `Home` navigation button in the header was added to showcase to the user that they are on the home page.

Sign up Page

Desktop - Light - Sign up

Riget Zoo Adventures Home Safari Bookings Explore Sign up Login

Sign up

Already have an account? [Log in](#)

First Name

Last Name

Email Address

Password OR

Confirm Password

☐ By creating an account, I agree to our [Terms of use](#) and [Privacy Policy](#)

[Continue with Google](#)

[Continue with Twitter](#)

[Sign up](#)

©Image by frimufilms on Freepik

Riget Zoo Adventures Privacy Policy Terms of Use Legal Support Contact Us

Above is the signup page an individual sees when they attempt to sign up for a new account with the application. To make this page more visually appealing an image of a panther has been used in the background to make the content look well designed and attractive. On the left-hand side is the form for handling the user's signup, it first begins asking if the user has an account, and if they do so give them a link they can click on to redirect to the login page, below this contains the two options a user can use to sign up for the application, the user can either signup by manually inputting their information or by using a third party social media account such as with Google or Twitter.

If they wish to signup manually for the application it asks the user for some information, this being the users first and last name, the email address they wish to use for the account along with the account's password, it asks for the password twice to confirm the user has typed the password correctly. Below this is a toggable form element that must be ticked for the user to sign up to the application, this element states that by creating an account with the application they agree to the applications `Terms of use` and the applications `Privacy Policy`. If all the data above is valid the user can then click on the `Sign up` button to create an account.

However, if they wish to use a third-party/social media account, they simply need to click on the button that corresponds to which account they wish to use to create the account, from which a popup will appear as stated with the third-parties signup system, that states that the user is attempting to create an account and the user needs to login/confirm this. Upon confirmation it will then register an account with the information that has been received from the third-party such as the users first name, last name and email address, we do not need to get a password as the user will login to the application via the same way they made an account.

Login Page

Above is the login page an individual sees when they attempt to login to an account with the application. In the backend an imager of a panther like in the signup page was used, however a different image was used. This was done to showcase that these systems work together as they have similar background image focuses, the panther, but they are different as they are not the same image, and as such different pages/content that are similar to one another. On the left-hand side is the form for handling the user's login, it first begins by checking if they user does not have an account, and if they don't have an account, it gives them a link to redirect to the signup page. Below this it contains the two options a user can use to login for the application, the user can either login by manually inputting their login details during the signup step, or they can login with a third-party account like they did during the signup step.

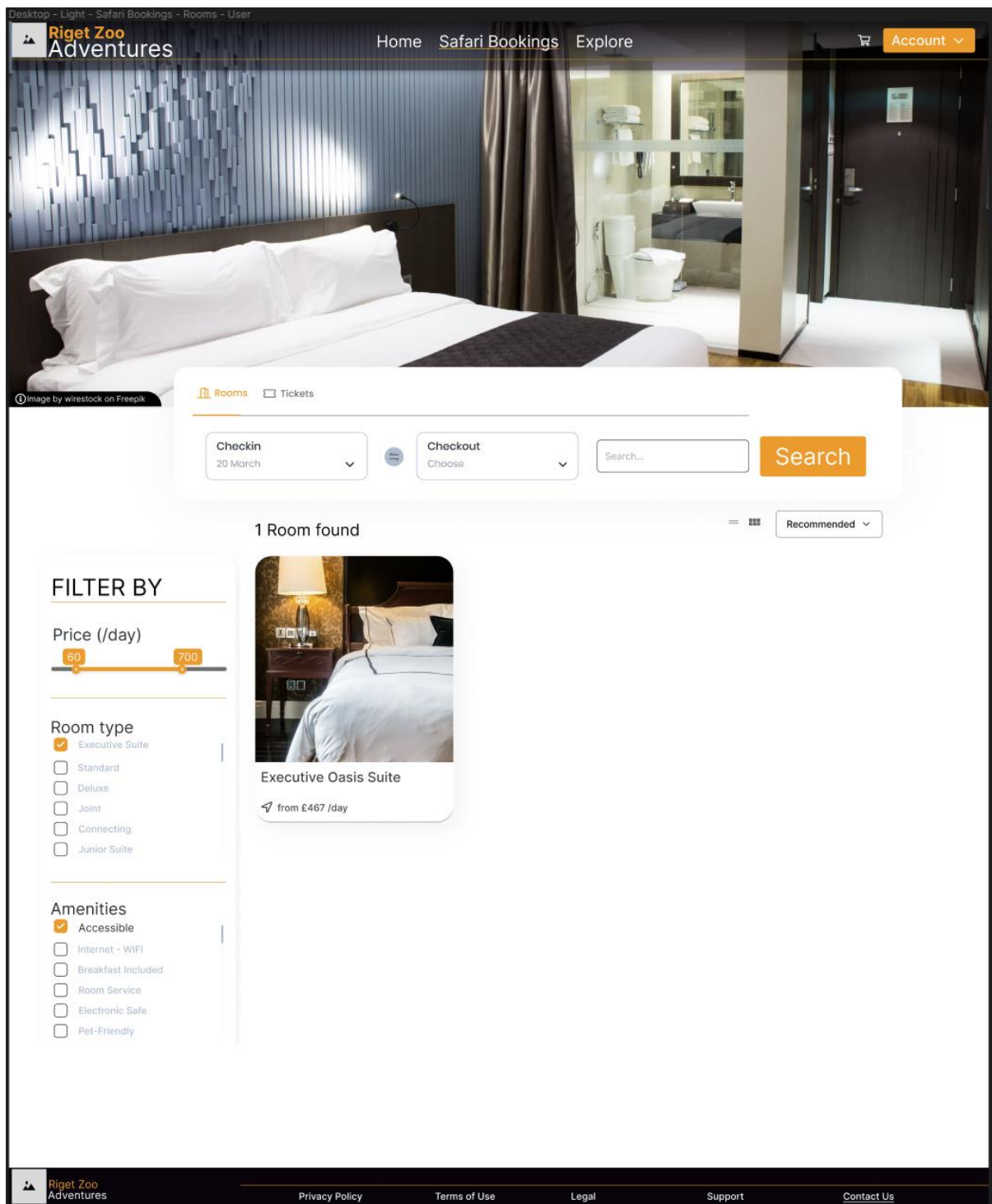
If the user wishes to manually login to the application it asks the user for the account's information, this being the accounts email address and password. Below this is a toggable option that allows users to make their session remaining logged in over multiple visits, if not ticked every time the user visits the application they have to login again, but if ticked they stay logged in until they either manually logout or the users account session has expired/changed. Beside this is a link to the page to reset your password in the case you have forgotten your password, on this page it would ask the user for their email address, then send a link to the email address to reset their password. Once all the data has been inputted and is confirmed as a valid login session, the user session is then set with the required data to login to their account.

However, if they used a third-party/social media account, they would need to login with their account they used during the signup process by clicking on the corresponding buttons. After clicking it will check with the third-party to ensure the user is who they are, and if upon confirmed it will modify the user's session with the correct data required to login to the user's account.

User

The designs listed below may look similar to the designs that other roles/administrators have access to, however the user may sometimes have access to features the administrators don't, while not having access to certain features/data an administrator may have access to.

Bookings – Room



Above is the page for booking hotel stays, within it contains an image at the top to attract user attention and make them interested in what type of rooms are available, below this is the feature to search for specific rooms the user requires. At the top, the user can swap between searching for rooms or for tickets, currently it is searching for rooms. Below this it asks the user for their check in and check out dates so the application can filter the results to rooms that are available during the specified time period, on the right is a text box that a user can use if they wish to search for a room by a specific word/text.

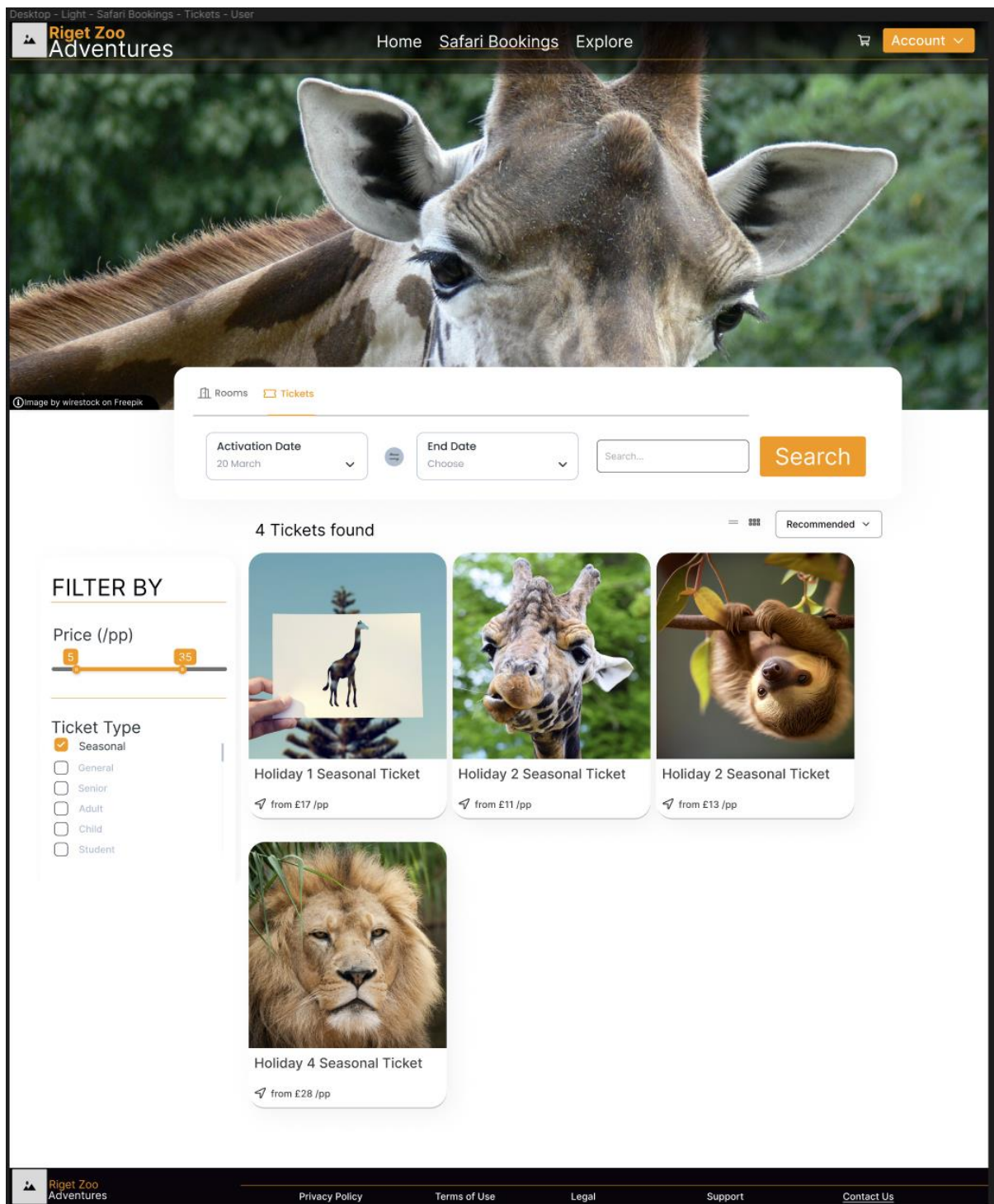
Below this on the left is further controls to filter the results more, within this at the top the user can specify the price of a room per day in GBP, this is a slider where the user can specify the minimum price and the maximum price, and the application will only show rooms between these prices. Below this is further control over what type of room the user wishes to stay in such as a standard room, deluxe room,

executive room, etc. In the last filter options are various amenities that the rooms provide that the user can specify such as being accessible to individuals with disabilities, having internet access, having room service, etc. All these filter points are useful for an individual so they can determine what room is the best for their current goal.

On the rest of the page, it will showcase each room depending on the visualization method the user specifies, in the design above it is the grid/card design but it also support a list style of visualization. Before being shown they are first sorted according to the style the user selects such as recommended, cheapest, most expensive, etc, allowing the user further fine control over search results. And below this it will show each room that matches all the users' criteria's, in the card design it showcases an image of the room, the name of the room and the price per day, in the case the user wants further information on this room they can click it and get taken to a separate page that will give further information relating to the room, along with giving the user the option to adding the room to their shopping cart.

Alongside this an underlining of the `Safari Bookings` navigation button in the header was added to showcase to the user that they are on the bookings page.

Bookings – Tickets



Above is the page for booking tickets, within it contains an image at the top to attract the user's attention and make them interested in what type of rooms are available, below this is the feature to search for specific tickets the user requires. At the top, the user can swap between searching for rooms or for tickets, currently it is searching for tickets. Below this it asks the user for the dates and time they wish for the tickets to start and end so the application can filter the results to tickets that are available during the specified time period, on the right is a text box that a user can use if they wish to search for a ticket by a specific word/text.

Below this on the left is further controls to filter the results more, within this at the top the user can specify the price of a ticket per person in GBP, this is a slider where the user can specify the minimum price and the maximum price, and the application will only show tickets between these prices. Below this

is further control over what type of ticket the user wishes to book such as Seasonal tickets, General tickets, adult tickets, Child tickets, etc. All these filter points are useful for an individual so they can determine what ticket is best for their current goal.

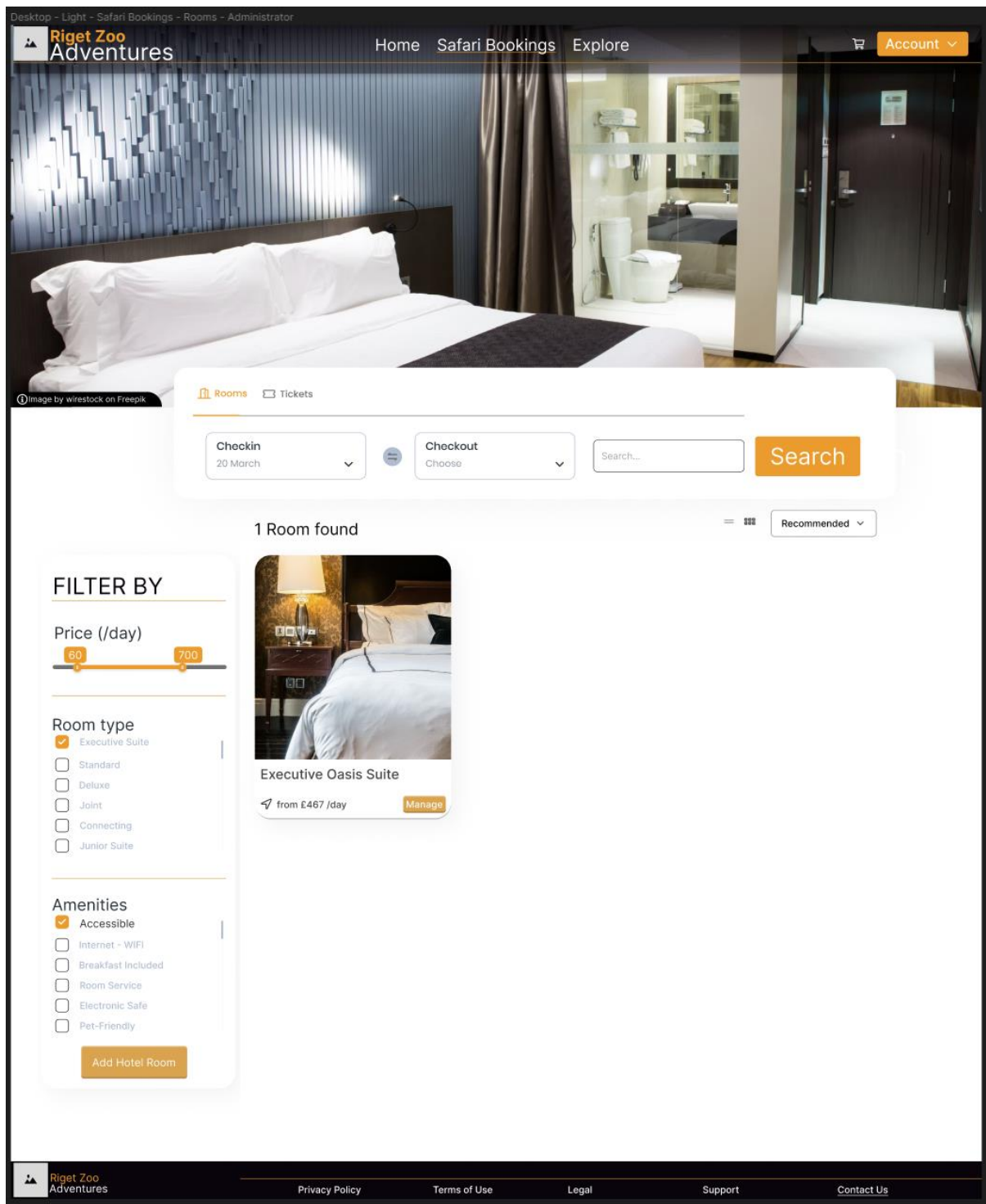
On the rest of the page, it will showcase each ticket depending on the visualization method the user specifies, in the design above it is the grid/card design but it also support a list style of visualization. Before being shown they are first sorted according to the style the user selects such as recommended, cheapest, most expensive, etc, allowing the user further fine control over search results. And below this it will show each ticket that matches all the user's criteria's, in the card design It showcases an image relating to the ticket to make it pop up and make the user have a higher chance of researching it, below this it will specify the name of the ticket and its price per person, in the case the user wants further information on this ticket they can click it and get taken to a separate page that will give further information relating to the ticket, along with giving the user the option to adding the ticket to their shopping cart.

Alongside this an underlining of the `Safari Bookings` navigation button in the header was added to showcase to the user that they are on the bookings page.

Administrator

The designs listed below may look similar to the designs that other roles/users have access to, however the administrator may sometimes have access to additional features/data that a standard user will not have access to, while not having access to certain features a standard user may have access to.

Bookings – Room





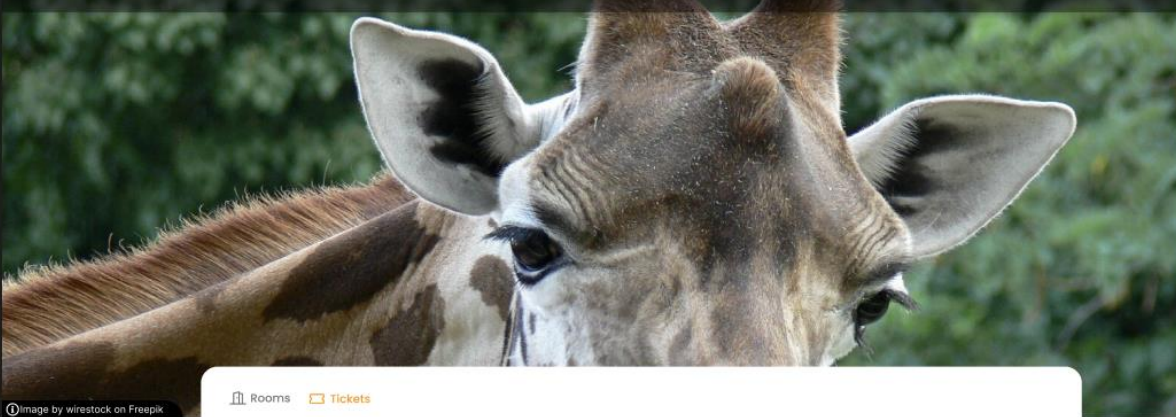
The layout of the hotel room booking is near identical to the standard user, as the administrator still has full control over specify dates, searching, filtering by parameters and sorting by styles. However, the administrator does not have access to the option of adding these rooms to their own shopping cart, as they do not have access to it.



On the other hand, the administrators have further access and control over the rooms that are listed, giving them the ability to add new hotel rooms to the site so they are available for booking, while at the same time managing already existing hotel rooms to change any parameters about them, such as what amenities they have, or their pricing as is required.




Bookings – Tickets



Desktop - Light - Safari Bookings - Tickets - Administrator

Riget Zoo Adventures Home Safari Bookings Explore  **Account** 




 Rooms  Tickets

Activation Date: 20 March  End Date: Choose  Search... 


4 Tickets found  Recommended 


FILTER BY

Price (/pp)




Ticket Type


- ☒ Seasonal
- ☐ General
- ☐ Senior
- ☐ Adult
- ☐ Child
- ☐ Student








Holiday 1 Seasonal Ticket

 from £17 /pp 






Holiday 2 Seasonal Ticket

 from £11 /pp 





Holiday 2 Seasonal Ticket

 from £13 /pp 



Holiday 4 Seasonal Ticket

 from £28 /pp 

Riget Zoo Adventures Privacy Policy Terms of Use Legal Support Contact Us

The layout of the ticket booking is near identical to the standard user, as the administrator still has full control over specifying dates, searching, filtering by type and sorting by styles. However, the administrator does not have access to the option of adding these tickets to their own shopping cart, as they do not have access to it.

On the other hand, the administrators have further access and control over the tickets that are listed, giving them the ability to add new tickets to the site so they are available for booking, while at the same time managing already existing tickets to change any parameters about them, such as whether they are active, or their pricing as is required.

Algorithms / Flowcharts

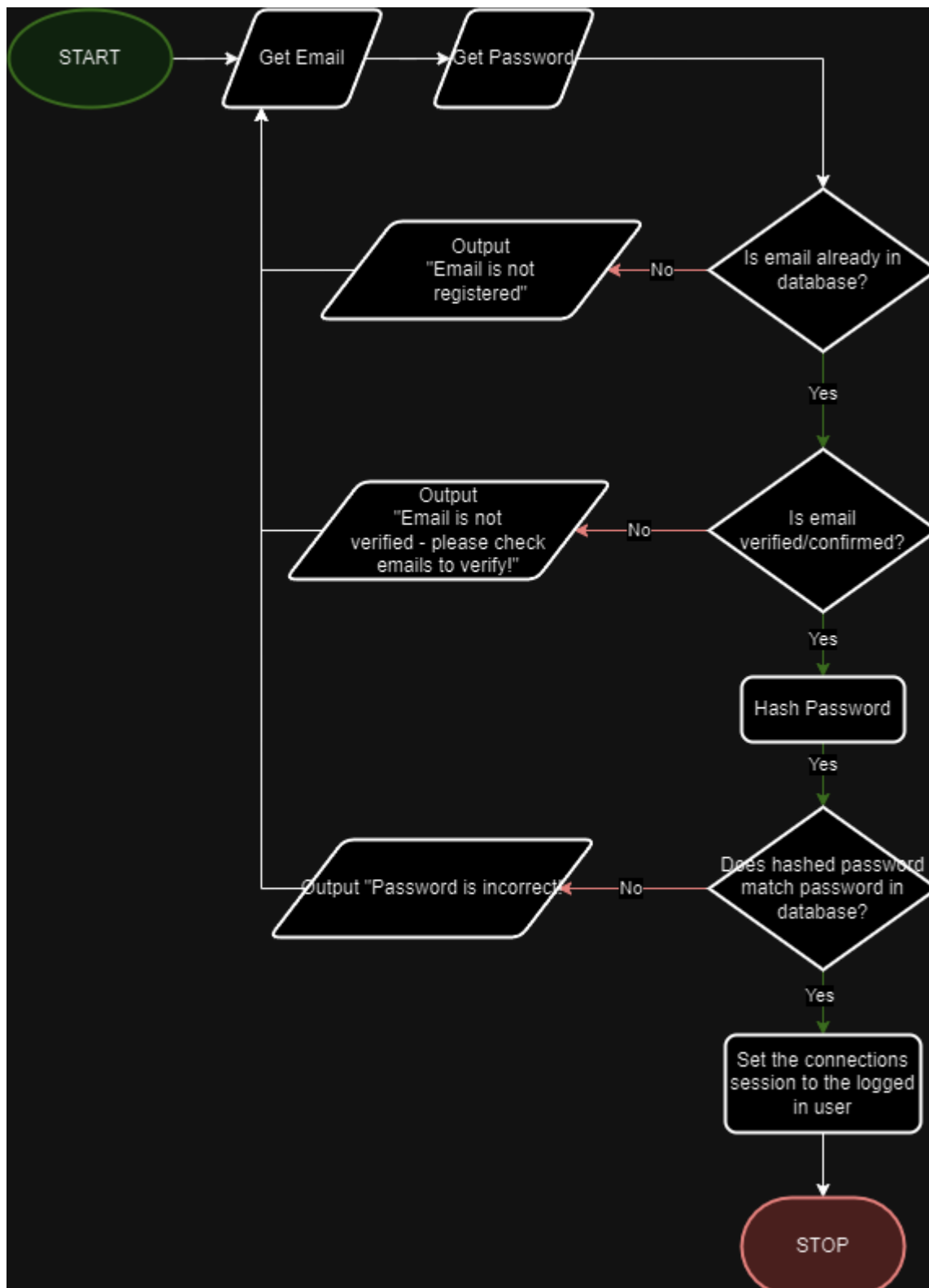
Below are the various algorithms that will be implemented within the proposed application to simplify this, flowcharts have been used to showcase how the algorithm will perform with a high-level overview. Some adjustments have been made to make the flowcharts to be as easy as possible for readers to understand while still effectively showcasing the algorithms to be implemented.

Register System



Above is the register system, it first starts by getting information from the user, this information being their Firstname, Lastname, Email address, Password and Confirm Password. Afterwards it performs its checks to ensure that the inputted information is correct, it first makes sure that the Firstname and Lastname is not too short, more than or equal to 1 character, and not too long, less than or equal to 20 characters. Next it ensures the email address is a valid email in the correct layout an email should be of `name@domain.tld`. It then ensures that the password and the confirm password match exactly, finally it performs the password security check to ensure it passes the security criteria such as containing a number, capital letter, special symbol, etc. The final check that occurs is to ensure that the provided email address is not already within the database, this is done to prevent duplicate accounts being linked to one email address. In the case any of these checks fail, it outputs a corresponding error message and then goes back to asking the user for their data again. If all the checks pass and are good it then hashes the password to ensure the password is kept securely, and from this it creates a new user in the application that they can login when provided the correct email address and password combination. As a final security measure, it then requests for the user to verify their email address, this is done by clicking on the verification link we have sent to their email address, once it has been detected this has occurred it updates the database and sets the email address as being verified, and now it is all good and allows for a user to login to their account.

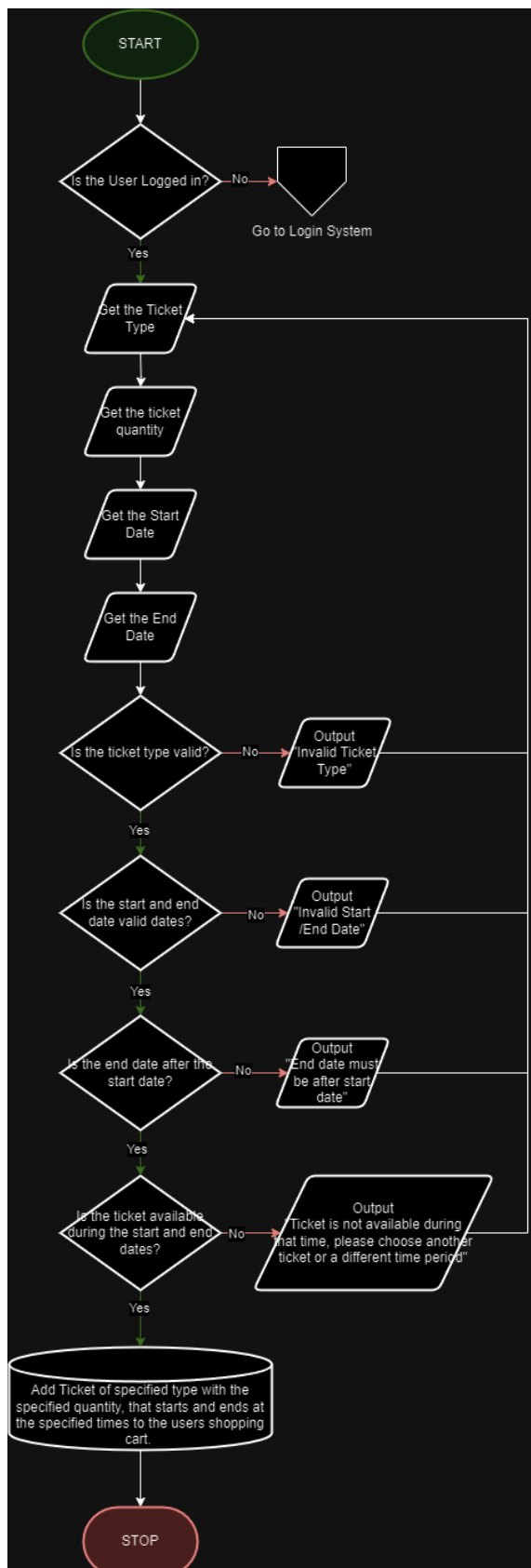
Login System



Above is the login system, it first starts by getting information from the user, this information being their accounts email address and password. Afterwards it performs its checks to determine whether the given combination was correct to login to an account. First it determines if the email address is in the database or not. If it is it, then confirms that the email address has been verified. If the email address has been verified it, then hashes the password that it has been given and checks it against the password in the database to determine if the correct password was given. In the case any of these checks fail, it outputs a corresponding error message and then goes back to asking the user for their data again. If all the checks

pass and are good it then modifies the user's session data accordingly to log them into the account, so they can now use the application as a logged in user.

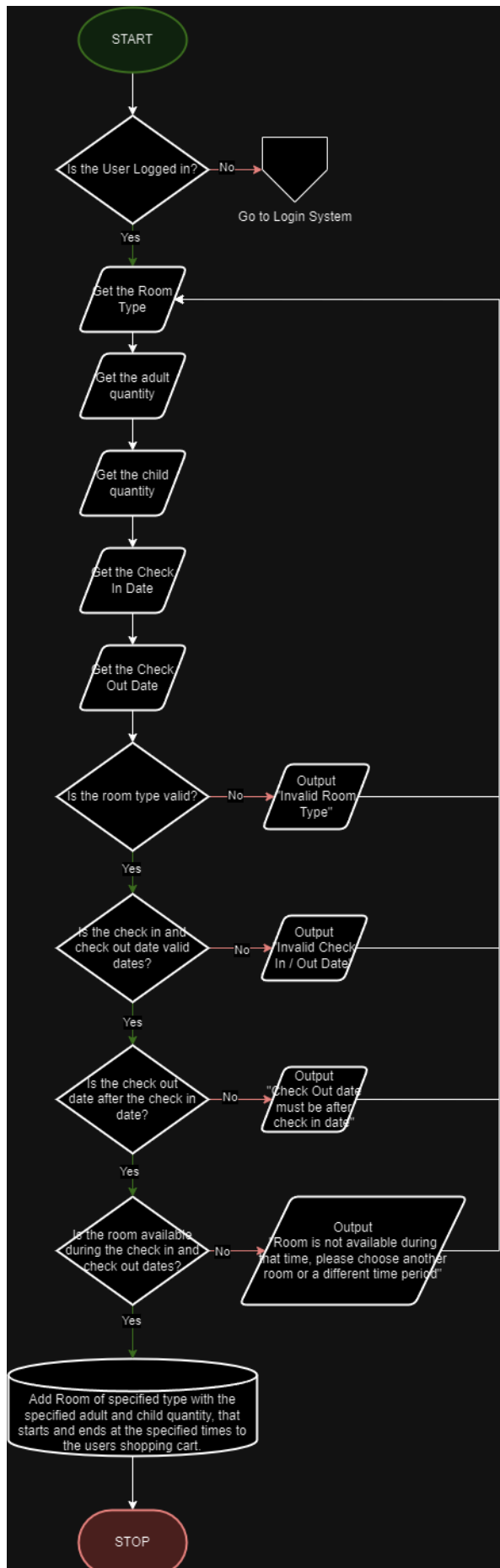
Shopping Cart - Ticket



Above is the implementation of adding a Ticket to the users shopping cart, it first starts by ensuring that the user is logged into an account, and if they are not, it sends them to the login system to make them login. If they are logged into an account, it instead gets information relating to the ticket the user wishes to book, the information it collects is the tickets type, the quantity of tickets the user wishes to book, the

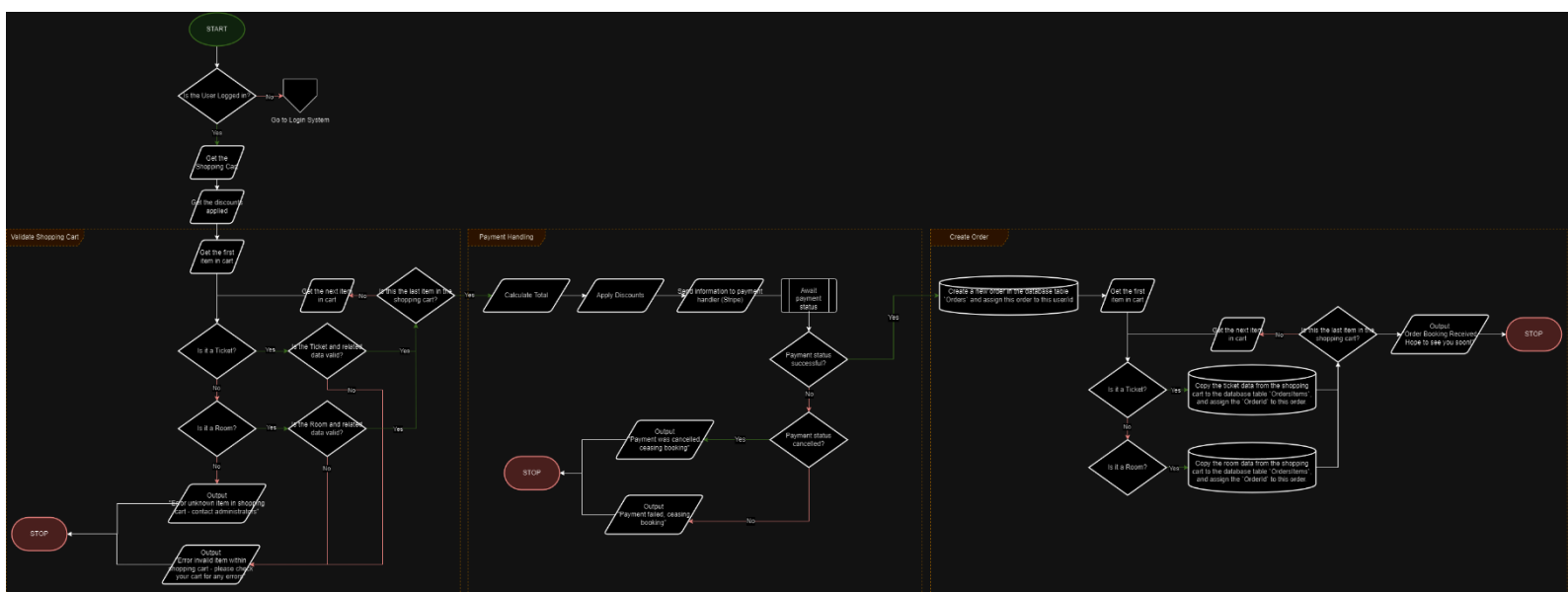
start date of the ticket and the end date of the ticket. Afterwards it performs its checks to ensure that the inputted information is correct, it first makes sure the given ticket type is a valid ticket type available for booking, next it performs a check to test if the start and end dates are valid dates, if they are it then checks to see if the end date is **after** the start date, if it is it then performs a check to determine if the ticket is available during the specified time period. In the case any of these checks fail, it outputs a corresponding error message and then goes back to asking the user for their data again. If all the checks pass and are good, it then adds this ticket and relevant data to the users shopping cart so they can then complete their booking once they have finished adding items to their cart.

Shopping Cart - Rooms



Above is the implementation of adding a Hotel Room to the users shopping cart, it first starts by ensuring the user is logged into an account, and if they are not, it sends them to the login system to make them login. If they are logged into an account, it instead gets information relating to the hotel room the user wishes to book, the information it collects is the rooms type, the number of adults that will be staying in the room, the number of children that will be staying in the room, the check in date and the check-out date. Afterwards it performs its checks to ensure that the inputted information is correct, it first makes sure the given room type is a valid room type available for booking, next it performs a check to test if the check in and check out dates are valid dates, if they are it then checks to see if the check-out date is **after** the check in date, if it is it then performs a check to determine if the hotel room is available during the specified time period, as somebody else may have booked this room sometime between your check in and check out dates. In the case any of these checks fail, it outputs a corresponding error message and then goes back to asking the user for their data again. If all the checks pass and are good, it then adds this hotel room and relevant data to the users shopping cart so they can then complete their booking once they have finished adding items to their cart.

Booking System



Above is the implementation of the booking system, it first starts by ensuring the user is logged into an account, and if they are not, it sends them to the login system to make them login. If they are logged into an account, it instead gets the users shopping cart and any discounts that has been applied to the shopping cart. The next process is split into three steps, the first step is validating the users shopping cart, this is done as a backup check in case one of the items have become invalid while it was in the cart, such as somebody else booking that room that will overlap your check in and check out dates, this simply reperforms the checks done during adding it to the cart to ensure that everything is still valid. In the case the data is not invalid it alerts the user so they can then modify anything that is giving an alert/error, after fixing it they can then attempt to book again.

If the validation check passes correctly, it then performs the process of charging the user for the booking, it first calculates the total of all the items in the shopping cart to get the final price, it then applies any discounts the user has used on the total to get the true total. Next it sends the price information to the payment handler, Stripe, so the payment can be setup. Now the flowchart waits for the user to finish paying and processing their order within the payment handler. Depending on the result of the payment is what occurs next, of the payment was cancelled or failed, it will give a corresponding output message then stop the booking. However, if the payment succeeds it goes onto the final step.

During the final step it converts the users shopping cart into an order, this order is then assigned to this user so only this user can view and manage it, excluding administrators. Afterwards it loops over all the items in the cart again and converts it to its corresponding types that is to be stored within the database tables `OrdersItems`, `TicketOrders` and `RoomOrders`. Once completed it will output a message stating it is complete, and then finish.

Project Data

The data structure for the applications database is an important consideration that must be considered carefully, this is due to both technical and legal reasons. For technical it must be simple to adapt and expand upon in the future to allow for new features to be added, along with being well designed to allow for replication/database operation over multiple servers. For legal we must know what data is collected, and how that data is stored so we can apply appropriate security measures and give information to the end-user about what data we collect.

To easily visualize the data structure for the application and the underlying system, an Entity Relationship Diagram (ERD) has been made and attached below. This table contains the tables required to implement the login and role system, along with the features requested by the client.

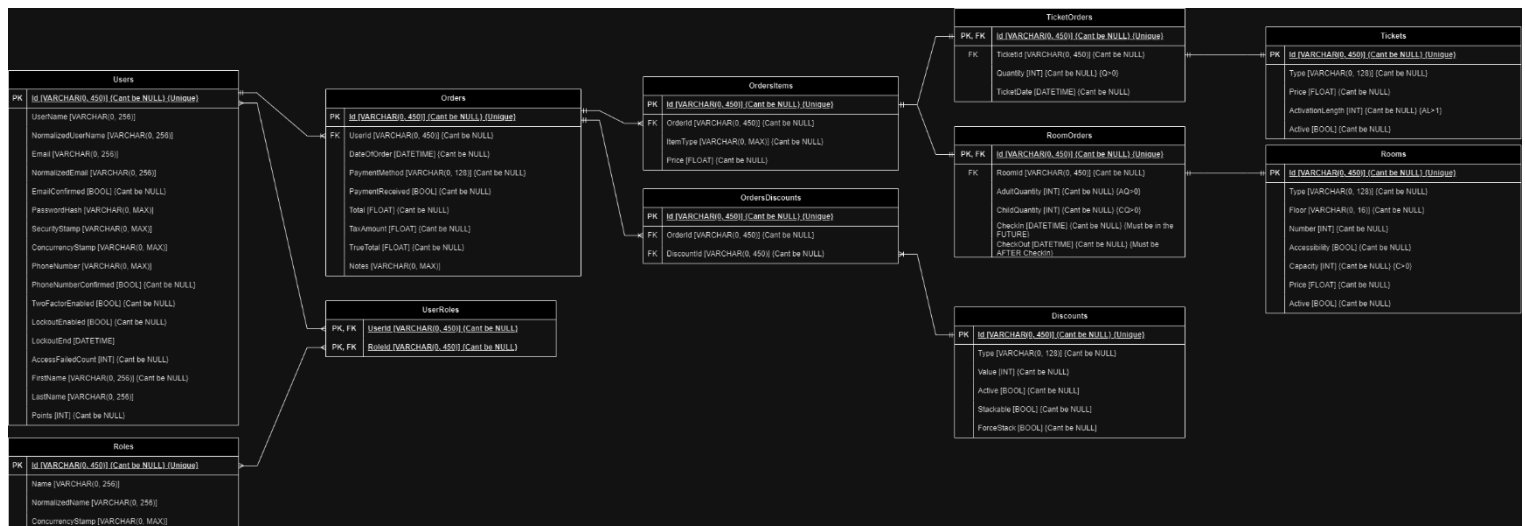
To understand this diagram, please note that any text that is specified as a `PK` is a primary key, whereas `FK` is a foreign key. Any text within [square] brackets are specific data types, to understand what this type means see the table below which explains each datatype along with giving an example. Any text within {curly} brackets are specific rules that must be applied to this data, such as a primary key being unique.

Data Type	Description
CHAR(NUM)	A string of characters which must be the length of the provided number. I.e. `CHAR(5)` must have an input of length 5, meaning `12345` is accepted but any longer/shorter text is rejected such as `1234` or `123456`.
VARCHAR(LNUM, UNUM)	A string of characters which must be equal to or larger than the first number, while at the same time less than the second number. I.e. `VARCHAR(2,5)` means that any inputs that is between 2 and 4 characters long is accepted (2,3,4). The following inputs are accepted; `12`, `123`, `1234`. Whereas the following inputs are rejected; ``, `1`, `12345`, `1234...`
INT	An integer value is a numerical value which is a whole number, contains no decimals, and without any non-numerical characters. I.e. The following inputs are accepted; `100`, `259174`. Whereas the following inputs are rejected; `100.123`, `10F`, `Hi`
FLOAT	A numerical value which contains a decimal, and without any non-numerical characters. I.e. The following inputs are accepted; `100.0`, `100.123`, `259174.0` Whereas the following inputs are rejected; `10F`, `10.0F`, `Hi`

BOOL	A value which can be in either one of two possible states, these states being `True` for yes/positive, while `False` for no/negative.
DATETIME	Contains a set Date and Time, and a UTC offset as well. I.e. `11/03/2024 14:19 +0:00` is a date and time with a UTC offset of 0 hours (equal to Coordinated Universal Time (UTC)). Whereas `11/03/2024 14:19 +3:00` is the same date and time but offset by 3 hours based on UTC and time zones.

Please note, that when showcasing example inputs and number such as `1/2/3/...` are used, that is used to simply show the length of the input, in place of these numbers any letter/characters can be used, but numbers have been used simplify the explanation as much as possible.

Below is the Entity Relationship Diagram (ERD), it showcases how the database will be implemented within the proposed application, it contains information related to how the users account data is stored, how the roles system is to be implemented along with how the overall ordering system will be implemented to allow for ticket purchases and room bookings.



In the diagram above there are a total of eleven (11) tables, of which three (3) are related to the account system while eight (8) of the tables is related to the booking/ordering system that allows for booking of tickets and booking of hotel rooms/stays.

To allow for a further understanding of these tables they have been converted to text alongside given explanations for each table existing alongside reasonings for each column that exists within the table, there are also examples provided to highlight how real data would look like within the database.

For the relationships between each table with their primary and foreign keys, please see the list below, it will describe the relationship between each table such as if it is a one-to-one relationship, one-to-many relationship, or a many-to-many relationship, alongside this it will also specify which columns are linked together to create this relationship.

- Users (Id, PK) ==One to Many=> Orders (UserId, FK)
- Users (Id, PK) ==Many to Many=> UserRoles (UserId, PK, FK)
- Roles (Id, PK) ==Many to Many=> UserRoles (RoleId, PK, FK)
- Orders (Id, PK) ==One to Many=> OrdersItems (OrderId, FK)
- Orders (Id, PK) ==One to Many=> OrdersDiscounts (OrderId, FK)

- Discounts (Id, PK) ==One to Many=> OrdersDiscounts (DiscountId, FK)
- OrdersItems (Id, PK) ==One to One=> TicketOrders (Id, PK, FK)
- OrdersItems (Id, PK) ==One to One=> RoomOrders (Id, PK, FK)
- TicketOrders (TicketId, FK) ==One to One=> Tickets (Id, PK)
- RoomOrders (RoomId, FK) ==One to One=> Rooms (Id, PK)

In each sub-heading listed below, they have been given a reason for existing, alongside having a table that showcases each column within that table alongside its corresponding information such as its data type, constraints, rules, reasoning for existing and example data, this has been done to make the database layout easily readable and understandable.

Users Table

This is the core table that stores all the information related to users and their accounts within the system/application, it is essential for user management and authentication as if this table doesn't exist, users cannot make/login to accounts.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 Max 450 PRIMARY KEY	Can't be NULL Must be unique	There must be a unique id for each user within the table, so it can be used as a direct reference to this account and its subsequent data.	17674779-3dbd-4961-a4bc-f591384b0ca2
Username	VARCHAR	Min 0 Max 256	N/A	An additional unique identifier a user can specify.	JohnDoe
NormalizedUsername	VARCHAR	Min 0 Max 256	N/A	Stores a consistent version of all usernames in same format/style, in all uppercase.	JOHNDOE
Email	VARCHAR	Min 0 Max 256	N/A	The email for the account, used and stored for being able to contact a user over email as required, along used by the user to login to the application itself.	John.doe@example.com
NormalizedEmail	VARCHAR	Min 0 Max 256	N/A	Stores a consistent version of all emails in same format/style, in all uppercase.	JOHN.DOE@EXAMPLE.COM
EmailConfirmed	BOOL	TRUE or FALSE	Can't be NULL	A Boolean to determine whether the user has confirmed their email address.	TRUE
PasswordHash	VARCHAR	Min 0 Max MAX	N/A	The user's password after is has been hashed to make sure it is secured safely, it uses the algorithm `PBKDF2` along with HMAC-SHA256, a 128-bit salt, 256-bit subkey and 10,000 iterations.	AQAAAAIAAYagAAAEF3o0kCqrnpIWWTfdttNhZRDTBn+jvUH+Ae2SVjVMU4NZYEmDnk7okJmnZA7LTy3zA==

SecurityStamp	VARCHAR	Min 0 Max MAX	N/A	The security stamp is a randomly generated value that is used as an additional security measure to invalidate authentication tokens and to prevent various types of cyber-attacks such as session fixation and CSRF (Cross-Site Request Forgery). It changes every time whenever a security-sensitive piece of information related to the users account changes, such as password updates or enabling/disabling 2FA, requiring any users who fail the check to login again. This helps ensure the users tokens are valid sessions, and enhancing overall security	IPGGP3QPZBNXTIO ZJ6Y53YUUCABPD3 NY
ConcurrencyStamp	VARCHAR	Min 0 Max MAX	N/A	The concurrency stamp is a value to determine whether the data you are adding should be accepted or not, this is done to ensure you do not accidentally overwrite another transactions modification with old/incorrect data.	c6de07dd-e233-448e-b4a9-7fe9c0462d22
PhoneNumber	VARCHAR	Min 0 Max MAX	N/A	The user may opt into giving us their phone number, this is as an extra security measure for the account alongside for account recovery purposes, it can also be used to contact the user for important alerts such as upcoming bookings	+1234567890
PhoneNumberConfirmed	BOOL	TRUE or FALSE	Can't be NULL	A Boolean to determine whether the user has confirmed their phone number, extra security.	TRUE
TwoFactorEnabled	BOOL	TRUE or FALSE	Can't be NULL	A Boolean to determine whether the user has enabled 2FA security.	FALSE
LockoutEnabled	BOOL	TRUE or FALSE	Can't be NULL	A Boolean to determine whether the account has the automated lockout security enabled.	TRUE
LockoutEnd	DATETIME	Must be DATETIME	N/A	The date when the automated lockout is disabled, as such allowing for a user to attempt to login again. If no data/value/set to `NULL`, it means that the account is not locked, and can be logged into.	NULL
AccessFailedCount	INT	N/A	Can't be NULL	This is used to determine how many fail access attempts has occurred on this account, if it	0

				reaches a certain amount, it triggers the automated lockout system.	
FirstName	VARCHAR	Min 0 Max 256	Can't be NULL	Used so we can determine who this individual is legally, so that in any scenario this information needs to be checked against it can. It is also used for displaying personalized messages, making the application more user-friendly.	John
LastName	VARCHAR	Min 0 Max 256	N/A	Used so we can determine who this individual is legally, so that in any scenario this information needs to be checked against it can. It is also used for displaying personalized messages, making the application more user-friendly.	Doe
Points	INT	N/A	Can't be NULL	This column is used to store how many points the user has within the loyalty and rewards point-based system, these points can then be used to exchange it for rewards such as free items / bookings / merchandise.	1731

Roles Table

This table is responsible for defining, managing, and storing the various types of roles that are within the database, which can then be added/given to a user within the `UserRoles` table. This table is essential for role-based control as stated within `Non-Functional` point `1.2` and point `1.3`

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 Max 450 PRIMARY KEY	Can't be NULL Must be unique	There must be a unique id for each role within the table, so it can be used as a direct reference to this role and its subsequent data.	23b73ad7-6948-4612-8112-a435164303be
Name	VARCHAR	Min 0 Max 256	N/A	The name of the role, such as `User`, `Employee`, `Administrator`, etc.	User
NormalizedName	VARCHAR	Min 0 Max 256	N/A	Stores a consistent version of all names in same format/style, in all uppercase.	USER
ConcurrencyStamp	VARCHAR	Min 0 Max MAX	N/A	The concurrency stamp is a value to determine whether the data you are adding should be accepted or not, this is done to ensure you do not accidentally overwrite another transactions modification with old/incorrect data.	29fc9370-fbbe-4c58-954e-0f8567fd52f9

UserRoles Table

This table servers as a junction/connection table between the `Users` table and the `Roles` table, allowing for a many-to-many relationship between each foreign key from each table, it is used to manage the roles a user has been given within the system as to give a user a new role all that is required is to add a new record to this table with their UserId and the RoleId of the role that they are to be given, whereas to remove a user from a role you simply need to remove the record with the corresponding UserId and RoleId.

Column Name	Data Type	Constraints	Rules	Reason	Example
UserId	VARCHAR	Min 0 Max 450 PRIMARY KEY FOREIGN KEY	Can't be NULL	This is to specify which user has this role, by making it a foreign key, it references the primary key `Id` in the `Users` table.	17674779-3dbd-4961-a4bc-f591384b0ca2
RoleId	VARCHAR	Min 0 Max 450 PRIMARY KEY FOREIGN KEY	Can't be NULL	This is to specify what role the user has received, by making it a foreign key it references the primary key `Id` in the `Roles` table.	23b73ad7-6948-4612-8112-a435164303be

Orders Table

This table stores all the orders that have occurred within the application, within each record it stores the details related to the order such as when it occurred, how much it cost along with any additional notes that have been added to the order itself.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 Max 450 PRIMARY KEY	Can't be NULL Must be unique	There must be a unique id for each order within the table, so it can be used as a direct reference to this order and its subsequent data.	6e388f83-e79a-449b-b35a-56bc1236837e
UserId	VARCHAR	Min 0 Max 450 FOREIGN KEY	Can't be NULL	A foreign key to establish a relationship between the `Users` table, and this table, the `Orders` table, via the other tables primary key.	17674779-3dbd-4961-a4bc-f591384b0ca2
DateOfOrder	DATETIME	Must be DATETIME	N/A	A datetime record of when the order occurred, used for record keeping.	2024/20/03 14:27
PaymentMethod	VARCHAR	Min 0 Max 128	Can't be NULL	This is a record of the payment method used to pay for the order, this is recorded so in the case this order needs to be modified/verified in the future, we can easily determine what service the used to pay.	PayPal
PaymentReceived	BOOL	TRUE or FALSE	Can't be NULL	A Boolean to determine whether we have received payment yet.	TRUE
Total	FLOAT	N/A	Can't be NULL	The total price of the all the items added together before any tax or discounts are applied to the total. Stored in GBP.	200.000

TaxAmount	FLOAT	N/A	Can't be NULL	The amount of tax applied to this order/purchase. Stored in GBP.	20
TrueTotal	FLOAT	N/A	Can't be NULL	The total price of this order after all the orders are added together with taxes, alongside any discounts that have been applied by the user to decrease the overall total.	181.94
Notes	VARCHAR	Min 0 Max MAX	N/A	This column contains any notes that have been added to this order, used to store additional information or comments. This will not be accessible to the user, and is mainly used for customer service/staff to fill in data, such as if the user has reported they are bringing a pet with them into the hotel room, this will be put into notes here	N/A

OrdersDiscounts Table

This table serves as a junction/connection table between the `Orders` table and the `Discounts` table, allowing for a one-to-many relationship between each foreign key from each table as each order can have multiple discounts, but each order discount record is to only be applied to one singular order that is specified, this also correspond to the `Discounts` table accordingly.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 Max 450 PRIMARY KEY	Can't be NULL Must be unique	There must be a unique id for each order discount within the table, so it can be used as a direct reference to this order discount and its subsequent data.	b0e635ab-cc11-4f9a-a90c-3e1247cabdc5
OrderId	VARCHAR	Min 0 Max 450 FOREIGN KEY	Can't be NULL	A foreign key to establish a relationship between the `Orders` table, and this table, the `OrdersDiscounts` table, via the other tables primary key.	6e388f83-e79a-449b-b35a-56bc1236837e
DiscountId	VARCHAR	Min 0 Max 450 FOREIGN KEY	Can't be NULL	A foreign key to establish a relationship between the `Discounts` table, and this table, the `OrdersDiscounts` table, via the other tables primary key.	f2bb2689-a1c6-4bdb-8122-63c3e51f19d3

Discounts Table

This table contains all the types of discounts that can be applied to an order, alongside this it also specifies parameters about the discount such as its type of discount and how much the discount is worth. It also allows for a discount to be stacked with other discounts, along with a bypass method to forcefully stack a discount even if another discount that is already applied does not allow stacking, this is

done to allow for the loyalty rewards to be applied to any order without having to worry about any discounts pre-applied to the order.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 Max 450 PRIMARY KEY	Can't be NULL Must be unique	There must be a unique id for each discount within the table, so it can be used as a direct reference to this discount and its subsequent data.	f2bb2689-a1c6-4bdb-8122-63c3e51f19d3
Type	VARCHAR	Min 0 Max 450	Can't be NULL	The type of discount this is, is it a percentage discount or an exact discount, i.e. take off 5%, or take of £10.00	Percentage
Value	INT	N/A	Can't be NULL	The value of the discount, depending on type of discount is how it is used, see below. If this is a percentage this will be the total percentage off. i.e. value=5%, price=£10, final=£9.5, discount amount=£0.50 If this is an exact amount it will take off this from final price. i.e. value=£5, price=£10, final=£5, discount amount=£5	5
Active	BOOL	TRUE or FALSE	Can't be NULL	Whether this discount is active, used to determine whether a user can add this discount to their order, if TRUE they can add this discount, if FALSE they cannot add this discount.	TRUE
Stackable	BOOL	TRUE or FALSE	Can't be NULL	Whether this discount can be stacked with other discounts.	FALSE
ForceStack	BOOL	TRUE or FALSE	Can't be NULL	This determines whether the discount can forcefully be applied to an order, even if another discount code does not allow stacking, can be used for discounts such as loyalty discount which can be applied on top of any other discounts.	FALSE

OrdersItems Table

This table stores every item that has been ordered within the application, along with what order this item corresponds to. It also states this items price and what type of item that was purchased such as if it was a `Ticket`, `Room`, etc, from which you could then query these corresponding tables to get further, and more specific, information related to this item.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 Max 450 PRIMARY KEY	Can't be NULL Must be unique	There must be a unique id for each order item within the table, so it can be used as a direct reference to this order item and its subsequent data.	c7a86c41-c2f1-412a-a6eb-a4e0e7065ddd

OrderId	VARCHAR	Min 0 Max 450 FOREIGN KEY	Can't be NULL	A foreign key to establish a relationship between the `Orders` table, and this table, the `OrdersItems` table, via the other tables primary key.	6e388f83-e79a-449b-b35a-56bc1236837e
ItemType	VARCHAR	Min 0 Max MAX	Can't be NULL	This column is used to determine what table the rest of the data for this specific item is in, such as if it is a ticket order, then it is in the `TicketOrders` table, whereas a room order is in the `RoomOrders` table.	Ticket
Price	FLOAT	N/A	Can't be NULL	The price for this item, as this is a generic piece of data all items will have, it is stored within this, so each table doesn't have to reimplement this column. The values are stored in GBP.	27.98

TicketOrders Table

This table has a one-to-one relationship with the `OrdersItems` table primary key of `Id`, as this table is meant to simply be a continuation of that records data, within this table it contains more specific information related to ordering of tickets that would not be generic amongst other orders such as a `Room` order, etc.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 Max 450 PRIMARY KEY FOREIGN KEY	Can't be NULL Must be unique	As this is a one-to-one relationship with a row of data within the `OrdersItems` table, it is meant to represent a continuity between the two tables, as this table stores more specific data related to ticket orders, whereas `OrdersItems` stores more generic information such as the OrderId and Price.	23a839e4-0c7f-4055-899b-ae259a39524f
TicketId	VARCHAR	Min 0 Max 450 FOREIGN KEY	Can't be NULL	A foreign key to establish a relationship between the `Tickets` table, and this table, the `TicketOrders` table, via the other tables primary key.	f548edc5-5f36-455f-9e01-73c7ca3f5b32
Quantity	INT	N/A	Can't be NULL Must be greater than `0`	This stores how many individuals are going to come under the specified ticket type/id, used to determine the final price after multiplying this value against the tickets price.	2
TicketDate	DATETIME	Must be DATETIME	Can't be NULL	The date the ticket is activated, from which the individuals with the ticket can then use them to enter the zoo until they run out/deactivated.	2024/22/03 8:00:00

ActivationLength	INT	N/A	Can't be NULL Must be greater than `1`	How long the ticket is active in hours, used so the ticket can be automatically disabled and become invalid once the day is over.	10
------------------	-----	-----	---	---	----

Tickets Table

This table contains all the tickets that are can be purchased within the application, each record corresponds to a specific ticket type, within is the price of the ticket, how long the ticket is valid for and whether the ticket is available for purchase.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 MAX 450 PRIMARY KEY	Can't be NULL Must be unique	There must be a unique id for each ticket within the table, so it can be used as a direct reference to this ticket and its subsequent data.	f548edc5-5f36-455f-9e01-73c7ca3f5b32
Type	VARCHAR	Min 0 MAX 128	Can't be NULL	This is done to easily identity what type of ticket it is, this is done by using standard naming conventions such as "Adult", "Child", "Senior", etc.	Adult
Price	FLOAT	N/A	Can't be NULL	The price of the ticket per individual. Price is stored in GBP.	13.99
ActivationLength	INT	N/A	Can't be NULL Must be greater than `1`	How long the ticket is active in hours, used so the ticket can be automatically disabled once the specified period of time is over,	10
Active	BOOL	TRUE or FALSE	Can't be NULL	This Boolean is used to determine whether this ticket is currently available for purchase or not. This is done so specific tickets can only be purchased during certain periods of time, such as for seasonal promotions, limited time offers etc.	TRUE

RoomOrders Table

This table has a one-to-one relationship with the `OrdersItems` table primary key of `Id`, as this table is meant to simply be a continuation of that records data, within this table it contains more specific information related to booking of hotel rooms that would not be generic amongst other orders such as a `Ticket` order, etc.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 Max 450 PRIMARY KEY FOREIGN KEY	Can't be NULL Must be unique	As this is a one-to-one relationship with a row of data within the `OrdersItems` table, it is meant to represent a continuity between the two	8bbccc4d-f924-4e25-89f6-8f13ebd1b7a6

				tables, as this table stores more specific data related to room orders, whereas `OrdersItems` stores more generic information such as the OrderId and Price.	
RoomId	VARCHAR	Min 0 Max 450 FOREIGN KEY	Can't be NULL	A foreign key to establish a relationship between the `Rooms` table, and this table, the `RoomOrders` table, via the other tables primary key.	1d6778b2-eb77-4132-86ff-6fe492c21f2a
AdultQuantity	INT	N/A	Can't be NULL Must be greater than `0`	This stores how many adults that are going to be staying within this specific room, used to determine whether they can fit/the room has capacity.	2
ChildQuantity	INT	N/A	Can't be NULL Must be greater than `0`	This stores how many children that are going to be staying within this specific room, used to determine whether they can fit/the room has capacity.	0
CheckIn	DATETIME	Must be DATETIME	Can't be NULL	The date and time the individual who has booked the room can checking and enter the room and stay within it until their period has ended.	2024/22/03 13:00:00
CheckOut	DATETIME	Must be DATETIME	Can't be NULL Must be AFTER `CheckIn`	The date and time their bookings end, by this time they should be packed and out the room as after this time they are not allowed to stay within the room as their booking has ended, and the staff must clean the room and put in new items for the next individuals that use this room.	2024/24/03 10:00:00

Rooms Table

This table contains all the rooms that are available for booking within the application, each record corresponds to a specific hotel room, from which specific data has been specified such as what type of room it is, what floor it is on, what is the rooms number, is it accessible to individuals with disabilities, what is the daily price for the room, etc.

Column Name	Data Type	Constraints	Rules	Reason	Example
Id	VARCHAR	Min 0 MAX 450 PRIMARY KEY	Can't be NULL Must be unique	There must be a unique id for each room within the table, so it can be used as a direct reference to this room and its subsequent data.	1d6778b2-eb77-4132-86ff-6fe492c21f2a
Type	VARCHAR	Min 0 MAX 128	Can't be NULL	This is done to easily identity what type of room each room is, this is done by using standard naming conventions such as "Standard", "Deluxe", etc	Standard

Floor	VARCHAR	Min 0 Max 16	Can't be NULL	Used to specify what floor this room is on, made a VARCHAR instead of INT, as in case of wanting a `Ground` floor instead of floor 0, or `B1` instead of `-1` for basement.	2
Number	INT	N/A	Can't be NULL	The rooms number, so we can know what room this data entry represents.	237
Accessibility	BOOL	TRUE or FALSE	Can't be NULL	A Boolean to showcase whether the room is accessible to individuals with disabilities or special needs, in the case they do these rooms have certain factors considered such as being on low floors/near elevators, having guards installed, etc.	FALSE
Capacity	INT	N/A	Can't be NULL Must be greater than `0`	This is to specify how many people the room is designed to hold.	2
Price	FLOAT	N/A	Can't be NULL	The price of the room per night/day, the price is stored in GBP.	100.00
Active	BOOL	TRUE or FALSE	Can't be NULL	This Boolean is used to determine whether this room is currently available for booking or not, regardless of current usage status. This is to mainly be used when a room needs to be removed from booking due to factors, such as remodelling or maintenance.	TRUE

Project Testing

For the successful implementation of the proposed features for the proposed solution, various types of data and methodologies will need to be used depending on what features is being tested, what data is being used to test, what data the feature can access and how the feature is implemented.

For testing of the data inputs within the application, various data types will be used to test the inputs and ensure they operate properly without any internal errors, while giving appropriate textual errors/alerts to the user so they can fix their inputs.

- Normal Data – Normal data is data/an input that conforms to all the validation rules relating to the input, this is used as a baseline/control test to ensure that the system works, and as such normal data should always work and be accepted, in the scenario that the data is not accepted/the input is rejected it means there is a problem with the implementation of the feature which must be fixed before any further testing on this feature can occur.
- Erroneous Data – Erroneous data is data/an input that fails the input's validation rules, this is done to ensure that validation rules are in place and are working as intended. An example of this

would be testing if you can input characters within a number only field and ensuring that they can't.

- **Boundary Data** – Boundary data is data/an input that is at the boundary of being accepted/rejected according to the input's validation rules, this is done to ensure that they are implemented and are displayed correctly to the user. This can be seen with the following example, where an input field only accept numbers **below** or **equal** to 1000, to test this is correctly implemented we will provide it with the following: ` 999 `, ` 1000 `, and ` 1001 ` to test if valid numbers that are below or equal to the limit, ` 999 ` / ` 1000 `, are accepted and ensure that invalid numbers, ` 1001 `, are rejected.
- **Extreme Data** – Extreme data is data/an input that contains very large pieces of content that is used to test the input, determine how an input handles large inputs and ensuring that the input is accepted/rejected accordingly. This can be done by repeating a character several times and inputting that into the input field.

There are also various methods of testing the application that can be further used to ensure that the application works as intended and is secure in its operations. Below is a list of different types of testing methodologies and what their use cases are for.

- **Data Testing** – This testing is to simply perform standard unit tests under a standard and controlled environment, to ensure that all the implemented tests can occur without any external/unpredictable factors affecting the test results.
- **Performance Testing** – This testing is used to determine and evaluate how the application performs under various conditions, such as high/low concurrent users, how it handles new content being added, etc. Several points such as how long pages take to load, how long new content to upload takes among others are then analysed to determine how well it performs against predictions/earlier versions. From which possible modifications can then be implemented to make the application more efficient.
- **Stress Testing** – Like performance testing, this is used to evaluate the applications performance, but under specifically high stress situations, such as if there are too many users online, the server overloading or under a potential cyber-attack such as a d/dos. This is then analysed to determine how the application responds to each scenario, and how to make it handle each case better and to mitigate these problems that may occur in the future before it does occur.
- **Security Testing** – This testing is done to ensure that the applications inputs are secure, and that there are no potential cyber security risks with inputs that could possibly interact with the database/other users. To do this they must be thoroughly tested to ensure that they are secure, this would be done via several cyber security attacks such as SQL injections to test for database leaks/vulnerabilities, along with stored XSS attacks to test that HTML sanitization works as intended.

Although there is a variety of different methods and data types listed above, many additional types can be used as well, however the ones previously stated are the primary and most used types to test systems during their development and deployment process. As such, testing of the previous types must occur

against the application sometime during its development process, or its public deployment/release process to ensure the application operates under a variety of different conditions.

From the proposed data and methodology listed above, various tests will be conducted during the developer process of the proposed application. Due to not knowing all the features that will be implemented a full-scale test plan cannot be developed, however a structure for a test plan can be made with the already known algorithms that will be implemented, this test plan will list out what the test is, what data to be inputted, what data type it is and the expected outcome when implemented. This test plan can then be tested against the algorithms that will be implemented within the final solution, these algorithms being the Register system, Login system, Shopping cart system for tickets, Shopping cart system for rooms and the booking/payment system, for a visual representation of these algorithms please see the design document heading labelled `Algorithms / Flowcharts`.

Please note, whenever something is `-(N)` , that means the letter was inputted N times, such as if it was `-(10000)` the letter was inputted ten thousand (10,000) times. To save space and to increase the readability of the test plan, the text has been shortened with the annotation above.

Description	Data	Expected Outcome	Actual Outcome	Comments
Attempt Invalid Register - Boundary – Bad First Name 1	First Name: S Last Name: Rogers Email: steve@steve.com Password: P@55word1 CPassword: P@55word1	Fail Register – First Name too short		
Attempt Invalid Register - Boundary – Bad First Name 2	First Name: S(*21) Last Name: Rogers Email: steve@steve.com Password: P@55word1 CPassword: P@55word1	Fail Register – First Name too long		
Attempt Invalid Register - Extreme – Bad First Name 3	First Name: S(*10000) Last Name: Rogers Email: steve@steve.com Password: P@55word1 CPassword: P@55word1	Fail Register – First Name too long		
Attempt Invalid Register - Boundary – Bad Last Name 1	First Name: Steve Last Name: R Email: steve@steve.com Password: P@55word1 CPassword: P@55word1	Fail Register – Last Name too short		
Attempt Invalid Register - Boundary – Bad Last Name 2	First Name: Steve Last Name: R(*21) Email: steve@steve.com Password: P@55word1 CPassword: P@55word1	Fail Register – Last Name too long		
Attempt Invalid Register - Extreme – Bad First Name 3	First Name: Steve Last Name: R(*10000) Email: steve@steve.com Password: P@55word1 CPassword: P@55word1	Fail Register – Last Name too long		

Attempt Invalid Register – Boundary – Bad Email 1	First Name: Steve Last Name: Rogers Email: Password: P@55word1 CPassword: P@55word1	Fail Register – Email is invalid		
Attempt Invalid Register – Boundary – Bad Email 2	First Name: Steve Last Name: Rogers Email: steve Password: P@55word1 CPassword: P@55word1	Fail Register – Email is invalid		
Attempt Invalid Register – Boundary – Bad Email 3	First Name: Steve Last Name: Rogers Email: steve@steve Password: P@55word1 CPassword: P@55word1	Fail Register – Email is invalid		
Attempt Invalid Register – Extreme – Bad Email 4	First Name: Steve Last Name: Rogers Email: A(*10000) Password: P@55word1 CPassword: P@55word1	Fail Register – Email is invalid		
Attempt Invalid Register – Boundary – Mismatch Password 1	First Name: Steve Last Name: Rogers Email: steve@steve.com Password: P@55word1 CPassword: P@55word2	Fail Register – Password doesn't match confirm password		
Attempt Invalid Register – Boundary – Mismatch Password 2	First Name: Steve Last Name: Rogers Email: steve@steve.com Password: P@55word2 CPassword: P@55word1	Fail Register – Password doesn't match confirm password		
Attempt Invalid Register – Extreme – Mismatch Password 3	First Name: Steve Last Name: Rogers Email: steve@steve.com Password: A(*10000) CPassword: P@55word1	Fail Register – Password doesn't match confirm password		
Attempt Invalid Register – Extreme – Mismatch Password 4	First Name: Steve Last Name: Rogers Email: steve@steve.com Password: P@55word1 CPassword: A(*10000)	Fail Register – Password doesn't match confirm password		
Attempt Valid Register – Normal - 1	First Name: Steve Last Name: Rogers Email: steve@steve.com Password: P@55word1 CPassword: P@55word1	Register Success – everything good		
Attempt Valid Register – Normal - 2	First Name: Steve1 Last Name: Rogers5 Email: steve@steve.com Password: P@55word1 CPassword: P@55word1	Register Success – everything good		Although uncommon, some individuals have numbers in their numbers, so it is accepted.
Attempt Invalid Login – Boundary – Bad Email - 1	Email: Password: P@55word1	Failed Login – Bad Email		
Attempt Invalid Login – Boundary – Bad Email - 2	Email: steve Password: P@55word1	Failed Login – Bad Email		

Attempt Invalid Login – Boundary – Bad Email - 3	Email: steve@steve Password: P@55word1	Failed Login – Bad Email		
Attempt Invalid Login – Extreme – Bad Email - 4	Email:A(*10000) Password: P@55word1	Failed Login – Bad Email		
Attempt Invalid Login – Boundary – Bad Password - 1	Email: steve@steve.com Password:	Failed Login – Bad Password		
Attempt Invalid Login – Boundary – Bad Password - 2	Email: steve@steve.com Password: Password	Failed Login – Bad Password		
Attempt Invalid Login – Extreme – Bad Password - 3	Email: steve@steve.com Password: A(*10000)	Failed Login – Bad Password		
Attempt Valid Login – Normal – User	Email: steve@steve.com Password: P@55word1	Success Login – All Good – Login to USER account		
Attempt Valid Login – Normal - Administrator	Email: admin@admin.com Password: S3cu!e	Success Login – All Good – Login to ADMINISTRATOR account		
Attempt Invalid Add Ticket to Cart – Boundary – Not logged in	Logged in? No Ticket Type: Adult Ticket Quantity: 1 Start date: 20/03/2024 10:00 End Date: 20/03/2024 20:00	Failed to add ticket to cart – redirected to login page		
Attempt Invalid Add Ticket to Cart – Erroneous – Invalid Ticket Type 1	Logged in? Yes Ticket Type: Steve Ticket Quantity: 1 Start date: 20/03/2024 10:00 End Date: 20/03/2024 20:00	Failed to add ticket to cart – bad ticket type		
Attempt Invalid Add Ticket to Cart – Extreme – Invalid Ticket Type 2	Logged in? Yes Ticket Type: A(*10000) Ticket Quantity: 1 Start date: 20/03/2024 10:00 End Date: 20/03/2024 20:00	Failed to add ticket to cart – bad ticket type		
Attempt Invalid Add Ticket to Cart – Erroneous – Invalid Start Date 1	Logged in? Yes Ticket Type: Adult Ticket Quantity: 1 Start date: Steve End Date: 20/03/2024 20:00	Failed to add ticket to cart – bad start date		
Attempt Invalid Add Ticket to Cart – Extreme – Invalid Start Date 2	Logged in? Yes Ticket Type: Adult Ticket Quantity: 1 Start date: A(*10000) End Date: 20/03/2024 20:00	Failed to add ticket to cart – bad start date		
Attempt Invalid Add Ticket to Cart – Erroneous – Invalid End Date 1	Logged in? Yes Ticket Type: Adult Ticket Quantity: 1 Start date: 20/03/2024	Failed to add ticket to cart – bad end date		

	10:00 End Date: Steve			
Attempt Invalid Add Ticket to Cart – Extreme – Invalid End Date 2	Logged in? Yes Ticket Type: Adult Ticket Quantity: 1 Start date: 20/03/2024 10:00 End Date: A(*10000)	Failed to add ticket to cart – bad end date		
Attempt Invalid Add Ticket to Cart – Erroneous – End date before start date	Logged in? Yes Ticket Type: Adult Ticket Quantity: 1 Start date: 20/03/2024 20:00 End Date: 20/03/2024 10:00	Failed to add ticket to cart – end date before start date		
Attempt Valid Add Ticket to Cart – Normal	Logged in? Yes Ticket Type: Adult Ticket Quantity: 1 Start date: 20/03/2024 10:00 End Date: 20/03/2024 20:00	Added ticket to the user's cart with the specified data		
Attempt Invalid Add Room to Cart – Boundary – Not logged in	Logged in? No Room Type: Deluxe Adult Quantity: 1 Child Quantity: 0 Check in date: 20/03/2024 14:00 Check out Date: 25/03/2024 11:00 Room Available during specified time: Yes	Failed to add room to cart – redirected to login page		
Attempt Invalid Add Room to Cart – Erroneous – Invalid Room Type 1	Logged in? Yes Room Type: Steve Adult Quantity: 1 Child Quantity: 0 Check in date: 20/03/2024 14:00 Check out Date: 25/03/2024 11:00 Room Available during specified time: Yes	Failed to add room to cart – bad room type		
Attempt Invalid Add Room to Cart – Extreme – Invalid room Type 2	Logged in? Yes Room Type: A(*10000) Adult Quantity: 1 Child Quantity: 0 Check in date: 20/03/2024 14:00 Check out Date: 25/03/2024 11:00 Room Available during specified time: Yes	Failed to add room to cart – bad room type		
Attempt Invalid Add Room to Cart – Erroneous – Invalid Check in Date 1	Logged in? Yes Room Type: Deluxe Adult Quantity: 1	Failed to add room to cart –		

	Child Quantity: 0 Check in date: Steve Check out Date: 25/03/2024 11:00 Room Available during specified time: N/A	bad check in date		
Attempt Invalid Add Room to Cart – Extreme – Invalid Check in Date 2	Logged in? Yes Room Type: Deluxe Adult Quantity: 1 Child Quantity: 0 Check in date: A(*10000) Check out Date: 25/03/2024 11:00 Room Available during specified time: N/A	Failed to add room to cart – bad check in date		
Attempt Invalid Add Room to Cart – Erroneous – Invalid Check out Date 1	Logged in? Yes Room Type: Deluxe Adult Quantity: 1 Child Quantity: 0 Check in date: 20/03/2024 14:00 Check out Date: Steve Room Available during specified time: N/A	Failed to add room to cart – bad check out date		
Attempt Invalid Add Room to Cart – Extreme – Invalid Check out Date 2	Logged in? Yes Room Type: Deluxe Adult Quantity: 1 Child Quantity: 0 Check in date: 20/03/2024 14:00 Check out Date: A(*10000) Room Available during specified time: N/A	Failed to add room to cart – bad check out date		
Attempt Invalid Add Room to Cart – Erroneous – Check out date before check in date	Logged in? Yes Room Type: Deluxe Adult Quantity: 1 Child Quantity: 0 Check in date: 25/03/2024 14:00 Check out Date: 20/03/2024 11:00 Room Available during specified time: N/A	Failed to add room to cart – check out date before check in date		
Attempt invalid Add Room to Cart – Erroneous – Room is not available	Logged in? Yes Room Type: Deluxe Adult Quantity: 1 Child Quantity: 0 Check in date: 20/03/2024 14:00 Check out Date: 25/03/2024 11:00 Room Available during specified time: No	Failed to add room to cart – room is not available between these times, either choose another room or new times		
Attempt Valid Add Room to Cart – Normal	Logged in? Yes	Added room to the user's cart		

	Room Type: Deluxe Adult Quantity: 1 Child Quantity: 0 Check in date: 20/03/2024 14:00 Check out Date: 25/03/2024 11:00 Room Available during specified time: Yes	with the specified data		
Attempt Invalid Booking – Boundary – Not logged in	Logged in? No Shopping Cart: 1x Adult Ticket – All data within is valid. 1x Room booking – All data within is valid. Payment: N/A	Failed to do booking – redirected to login page		
Attempt invalid Booking – Erroneous – Invalid Shopping Cart data 1	Logged in? Yes Shopping Cart: 1x Adult Ticket – Data within is invalid. 1x Room booking – All data within is valid. Payment: N/A	Failed to do booking – bad data		
Attempt invalid Booking – Erroneous – Invalid Shopping Cart data 2	Logged in? Yes Shopping Cart: 1x Adult Ticket – All data within is valid. 1x Room booking – Data within is invalid. Payment: N/A	Failed to do booking – bad data		
Attempt invalid Booking – Erroneous – Invalid Shopping Cart data 3	Logged in? Yes Shopping Cart: 1x Adult Ticket – All data within is valid. 1x Room booking – All data within is valid. 1x Unknown Item – This is an invalid item. Payment: N/A	Failed to do booking – bad data		
Attempt invalid Booking – Erroneous – Payment Failed	Logged in? Yes Shopping Cart: 1x Adult Ticket – All data within is valid. 1x Room booking – All data within is valid. Payment: Failed	Failed to do booking – payment failed lack of funds		
Attempt invalid Booking – Erroneous – Payment Cancelled	Logged in? Yes Shopping Cart: 1x Adult Ticket – All data within is valid. 1x Room booking – All data within is valid. Payment: Cancelled	Failed to do booking – payment cancelled		
Attempt valid booking – Normal	Logged in? Yes Shopping Cart:	All is good, payment		

	1x Adult Ticket – All data within is valid. 1x Room booking – All data within is valid. Payment: Paid	received, make a new order and assign to user. Then make 2 new `OrderItems` 1 of Ticket, 1 Of room with this data, and then assign it to this Order		
--	---	---	--	--