RIGET ZOO ADVENTURES – TASK 1 – PROJECT PROPOSAL

Business Context

The client, Riget Zoo Adventures, operates within the tourism and leisure sector and requires a digital platform to improve how users interact with local attraction. Current process relies on onsite booking, causing inefficiencies and inconsistent user experience. Users need an intuitive interface to complete tasks such as being able to reserve and book tickets for the zoo, check availability and book a stay at the hotel, and be able to get help and information about the attractions and facilities. Administrators require structure tools for managing bookings and tickets.

Research Conducted by the Client

The client has carried out market research with existing use4rs to identify features that should be included in the digital solution. The suggested improvements include:

- Account registration to allow customers to manager their bookings
- Accessibility features to support a wide range of users
- A loyalty and reward scheme

These findings guide the priorities of the proposal and ensure that solution aligns with the user expectations.

Functional Requirements

- Users can register, log in securely, and update their personal details.
- Users can browse services. View information, filter options, and check availability
- Users can complete bookings, modify them, and receive confirmation messages
- Users automatically earn reward points and can redeem them for offers
- The system prevents double bookings and validates all user input
- Administrators can add, update, deactivate, and delete service items
- Administrators can manage pricing availability and review analytics
- Administrators access a secure panel protected by role-based permissions.

Non-Functional Requirements

- **Usability:** The interface must be easy to understand and require no training
- **Security:** All sensitive data must follow best practices such as encryption and access control
- **Performance:** All pages and process should load within 3 seconds
- **Reliability:** The system should maintain an uptime of 99.9% during operational hours.
- **Scalability:** The system must support user growth and future features.
- **Accessibility**: All content must comply with WCAG 2.1 AA guidelines

Problem Decomposition

1. Account Management – user registration, authentication, and profile updates.
2. Browsing Module – Viewing services, search, filtering and exploring information.
3. Booking Module – booking creation, modification, cancellation, and confirmation.
4. Rewards System – awarding points, updating balances, and redeeming rewards.
5. Admin Module – managing services, pricing, content, availability, and analytics.
6. Data Management – secure storage, retrieval, and processing of system data.

Project Data

The system requires a structured database design to ensure scalability, consistency, and legal compliance with UK GDPR. Clear identification of the data collected and how it is stored allows secure processing, minimised retention, and appropriate access control.

An **Entity Relationship Diagram (ERD)** has been included to model how the system stores user accounts, login details, bookings, and the loyalty scheme. In this diagram:

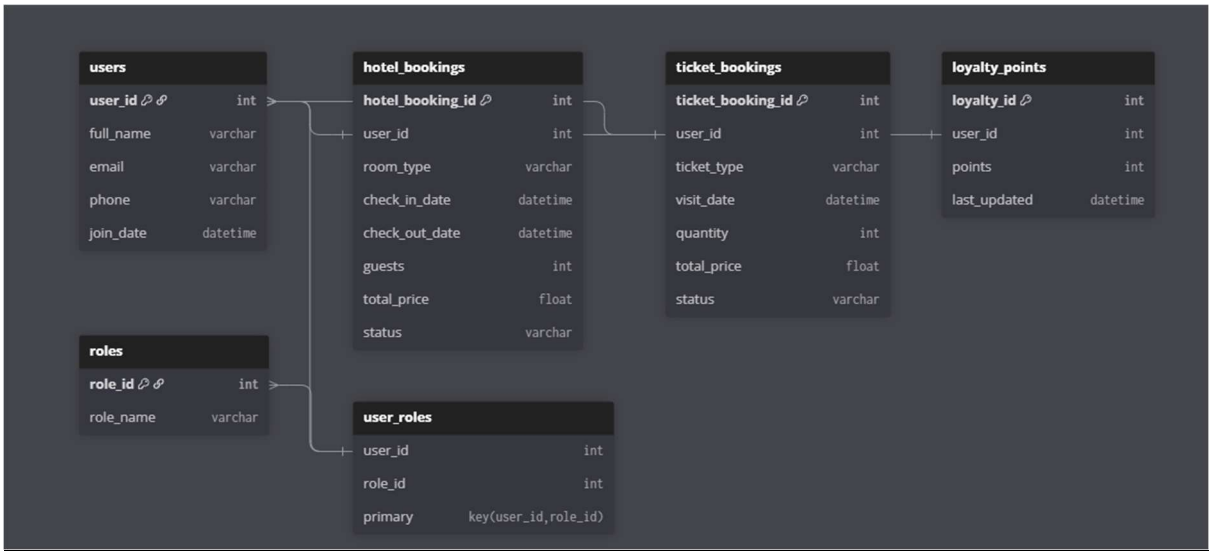
- PK indicates a **Primary Key**, which uniquely identifies each record in a table.
- FK indicates a **Foreign Key**, which links related data between tables.
- Text within **[square brackets]** specifies a data type (see the table below for explanations and examples).
- Text within **{curly brackets}** defines a rule or constraint applied to that field (for example, `{unique}` ensures no duplicate values).

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.

Entity Relationship Diagram (ERD)



Below is the database tables used in the system, including their fields, data types, and constraints.

Users Table

Column Name	Data Type	Constraints	Rules	Reason	Example
User id	VARCHAR	N/A	Primary Auto Increment Unique Not null	Uniquely identifies each user	1
full name	VARCHAR	Min 0 Max 255	Not null	Stores the users full name for identification	Bob English
Email	VARCHAR	Min 0 Max 255	Not null unique	Used for login and communication	test@gmail.com
Password hash	VARCHAR	Min 0 Max	Not null	Stores hashed password securely	$2Y$10x£5asd!Xx-
Join date	VARCHAR	DATETIME	Not null	Tracks when the account was created, useful for reporting/loyalty	21/11/2025

Hotel Bookings Table

Column Name	Data Type	Constraints	Rules	Reason	Example
Hotel booking ID	INT	N/A	PK, auto-increment	Uniquely identifies each hotel booking	1001
User Id	INT	N/A	FK, users.user_id required	Links booking to the user who made it	1
Room type	VARCHAR	N/A	Required {values: single / double / family}	Need to know what room they booked (single / double / family)	Family suite
Total price	FLOAT	Min 0 Max 500000	Required	Calculate automatically (Price x Nights / quantity)	140.00
Check in date	DATETIME	DATETIME	Required	To know when stay starts	2025-11-11 10:00:00
Check out date	DATETIME	DATETIME	Required	To know when stay ends	2025-11-18 10:00:00
Guests	INT	Min 0 Max 5	Required, must be 0	Ensures booking has at least 1 guest + capacity checks	2
Status	VARCHAR	N/A	Required, {values: booked/cancelled}	Track lifecycle of a booking: active or cancelled	Booked/ cancelled

Ticket Bookings Table

Column Name	Data Type	Constraints	Rules	Reason	Example
Ticket booking ID	INT	N/A	PK, auto-increment	Unique ID per ticket booking	5001
User ID	INT	N/A	FK- users.user_id, Required	Links ticket purchase to a user	1
Ticket type	VARCHAR	N/A	Required {values: Child/adult/family}	Calculate the cost of the tickets	Family Ticket
Total price	FLOAT	Min 0 Max 500000	Required	Calculate automatically (Price x Nights / quantity)	140.00
Visit date	DATETIME	N/A	Required	Date of visit	2025-12-24 10:00:00
Quantity	INT	Min 1 Max 30	Required, must be > 0	At least 1 ticket	4
Status	VARCHAR	N/A	Required, {values: booked/cancelled}	Track lifecycle of a booking: active or cancelled	Booked

Loyalty Points Table

Column Name	Data Type	Constraints	Rules	Reason	Example
Loyalty Id	INT	N/A	PK, auto-increment	Unique record per loyalty row	3001

User id	INT	N/A	FK-users.user_id, Required	Connects points to the correct user	1
Points	INT	Min 0 Max 100000	Required, Default 0, >= 0	Points can't be negative	1200
Last update	DATETIME	N/A	Required	Shows last time points changed	2025-11-13 10:00:00

Roles Table

Column Name	**Data Type**	**Constraints**	**Rules**	**Reason**	**Example**
Role id	INT	N/A	PK	Unique role ID	1
Role name	VARCHAR	N/A	Required, unique	Defines roles (user/admin)	Admin

User Roles Table

Column Name	**Data Type**	**Constraints**	**Rules**	**Reason**	**Example**
User id	INT	N/A	FK – users, part of PK	Links user to role	1
Role id	INT	N/A	FK – roles, part of PK	Links role to user	2
Primary		N/A	Composite PK (user/role rows)	Prevents duplicate user/role rows	

Relationships:

- One **users** -- > many **hotel_bookings**
- One **users** -- > many **ticket_bookings**
- One **users** -- > **loyalty_points**
- One **users** -- > many **user_roles,** many-to-many via **roles**

Key Performance Indicators (KPIs)

- 95% successful booking completion rate
- Error rate below 5% across all user tasks
- Average page load time under 3 seconds
- System uptime at or above 99%
- 90% satisfaction rating or higher in user surveys
- Booking processing time below 2 seconds

User Acceptance Criteria (UACs)

Users must be able to:

- Complete core tasks such as:
    - Reserve and book tickets for the zoo
    - Check availability and book a stay at the hotel
    - Manage their bookings through account registration
- Access essential information within 3 interactions of being on the website
- Navigate the interface with no specialist
- Login with valid credentials
- Update personal information

Administrators must be able to

- Manage booking and ticket system without technical assistance
- Adjust permissions and oversee activity

Proposed Solution Summary

The proposed solution is a responsive web-based platform structure to support both users and administrators. It prioritised clear navigation, consistent layout, and accessibility to ensure users can complete tasks such as to booking services and redeeming rewards efficiently.

The backend handles authentication, data validation, and secure processing while maintaining a modular structure that allows future expansion. Administrators are provided with a dedicated dashboard for updating hotel and ticket availability, reviewing activity, and managing permissions. The system is designed for scalability, stability, and long-term maintainability.

Justification

This solution aligns with the client's goals by improving usability, accessibility, and operational efficiency. Its modular structure supports future enhancements, such as integrating a shopping system. Strong data handling practices ensure legal compliance and maintain user trust. This approach provides a balance, scalable, and future proof solution.

Risk and Mitigations

Risk Description	Likelihood (1-10)	Impact (1-10)	Mitigations
Data breach	4	10	Encryption, secure authentication, hashed passwords
Server downtime	5	9	Loading balancing, failover systems, monitoring tools
Double-bookings	6	8	Server-side validation and locking mechanisms
Payment failures	5	8	Retry logic, user alerts, API monitoring
GDPR breaches	3	10	Access control, privacy audits
Data loss	2	10	Backups, restore points, redundancy
User input errors	7	6	Strong validation
Accessibility issues	4	8	WCAG compliance, assistive tech testing

Regulatory & Legal Compliance

UK GDPR – data is minimised, encrypted, securely stored, and fully transparent to users.

WCAG 2.1 AA – interface designed with semantic structure, alt text, keyboard navigation, and readable colours.

Copyright and Licensing – all assets, images, and third-part resources are obtained legally and properly referenced

PCI-DSS – secure payment processing through trusted gateways.