# Task 1 – Design Documentation

Contents

[Task 1 – Design Documentation 1](#_Toc161839136)

[Database Design 1](#_Toc161839137)

[Tables 1](#_Toc161839138)

[Entity Relationship Diagram (ERD) 7](#_Toc161839139)

[Data Definition Language (DDL) 7](#_Toc161839140)

[Wireframes 9](#_Toc161839141)

[Defining Colours 9](#_Toc161839142)

[Figma Wireframes 10](#_Toc161839143)

[Design Guidelines 17](#_Toc161839144)

[Flowchart for various pages 18](#_Toc161839145)

[Testing 18](#_Toc161839146)

[Types of tests 18](#_Toc161839147)

[Types of data that will be used 18](#_Toc161839148)

It is essential that we include documentation on the database design. Below is a detailed database design followed by the entity relationship diagram (ERD). Many parts of the database will be designed throughout the project as we do not know exactly what is needed for them, but the tables below give us an idea of how it will work.

## Database Design

A draft database has been developed to create the base database model, the data definition language and entity relationship diagram have been automatically generated through a database management tool.

### Tables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tables** | | | | |
| Table | Description | | | |
|  |  | | | |
| **usergroups** | **Table storing role data – id of the role and the name of the role** | | | |
| Column Name | Column Type | | | Column Description |
| id | SMALLINT  PRIMARY KEY  UNIQUE  NOT NULL | | | The ID of the role, this will be used to reference the role in other tables via foreign keys |
| name | VARCHAR (10)  UNIQUE  NOT NULL | | | The name of the role, this can be used by the system to display the name of the role |
|  |  | | | |
| **users** | **Table storing user data e.g. usernames and passwords and user groups** | | | |
| Column Name | Column Type | | | Column Description |
| id | INTEGER  PRIMARY KEY  UNIQUE  NOT NULL | | | The primary key of the table, each user will be assigned a user id (uid), and this can be referenced to reference a user |
| username | VARCHAR (20)  NOT NULL  UNIQUE | | | The column holding the value of the username, this can be a maximum of ten characters and must be unique |
| email | VARCHAR (45)  UNIQUE  NOT NULL | | | The column holding the value of the email for the user |
| password | VARCHAR  NOT NULL | | | The column holding the value of the hashed password, the length constraint of the password column is validated by the system and not the SQLite database |
| datetime\_created | DATETIME  DEFAULT (CURRENT\_TIMESTAMP) NOT NULL | | | The column holding the value of the account creation date, this can be used by analytics tools for insights |
| last\_login | DATETIME  DEFAULT(CURRENT\_TIMESTAMP)  NOT NULL | | | The column holding the value of the last login datetime, this can be used by analytics tools for insights |
| usergroup | SMALLINT  DEFAULT ‘1’  NOT NULL | | | The column holding the integer reference to the role of the user, the default is 0 so that the user is automatically given the lowest permission level |
| FOREIGN KEY (usergroup) REFERENCES usergroups (id) | | | | |
|  |  | | |  |
| **loyalty\_rewards** | **Table storing the types of rewards that can be redeemed with loyalty points** | | | |
| Column Name | Column Type | | | Column Description |
| id | INTEGER  NOT NULL UNIQUE PRIMARY KEY | | | The id for the reward, this will be referenced when calculating the amount of reward points a user has redeemed and in the log of loyalty points being redeemed |
| name | VARCHAR  NOT NULL  UNIQUE | | | The name of the loyalty reward |
| icon | VARCHAR  NOT NULL | | | The icon that will be used on the site when displaying the loyalty rewards |
| cost | INTEGER  NOT NULL | | | The amount of loyalty points it costs to redeem the reward |
|  |  | | |  |
| **loyalty\_rewards\_redeemed** | **Table storing the log of reward points being redeemed, this can be used by analytics tools to show the value of the loyalty reward scheme** | | | |
| Column Name | Column Type | | | Column Description |
| id | INTEGER  NOT NULL  UNIQUE  PRIMARY KEY | | | The log id number |
| uid | INTEGER  NOT NULL | | | The id referencing the user that redeemed the reward |
| rid | INTEGER  NOT NULL | | | The id referencing the reward that was redeemed |
| datetime\_redeemed | DATETIME  DEFAULT (CURRENT\_TIMESTAMP)  NOT NULL | | | The datetime that the reward was redeemed, this will be useful when analysing the value of the reward scheme |
| FOREIGN KEY (uid) REFERENCES users (id) | | | | |
| FOREIGN KEY (rid) REFERENCES loyalty\_rewards (id) | | | | |
|  | |  |  | |
| **loyalty\_points** | | **The table storing the number of loyalty points each user has and how many points they have redeemed** | | |
| Column Name | | Column Type | Column Description | |
| id | | INTEGER  NOT NULL  UNIQUE  PRIMARY KEY | The id of the user | |
| gathered | | INTEGER  NOT NULL  DEFAULT ‘0’ | The number of points they have gathered | |
| redeemed | | INTEGER  NOT NULL  DEFAULT ‘0’ | The number of points they have redeemed | |
| FOREIGN KEY (id) REFERENCES users(id) | | | | |
|  | |  |  | |
| **rooms** | | **The table storing the information on all the types of rooms available in the hotel** | | |
| Column Name | | Column Type | Column Description | |
| id | | INTEGER  NOT NULL  UNIQUE  PRIMARY KEY | The ID of the room type that makes the rows easily accessible by the **room\_bookings** table | |
| name | | VARCHAR  NOT NULL  UNIQUE | The name of the type of room | |
| capacity | | INTEGER  NOT NULL | The maximum amount of people that can use the room | |
| price | | INTEGER  NOT NULL | The price of the room per night | |
| amount | | INTEGER  NOT NULL | The number of rooms in the on-site hotel of this type | |
|  | |  |  | |
| **room\_bookings** | | **The table showing the information on room accessibility (if a room is booked and who by)** | | |
| Column Name | | Column Type | Column Description | |
| id | | INTEGER  NOT NULL  UNIQUE  PRIMARY KEY | The ID of the room accessibility record | |
| room\_type | | INTEGER  NOT NULL | The type of room that the record is referencing | |
| is\_booked | | SMALLINT  NOT NULL  DEFAULT ‘0’ | The flag to say if a room is booked or not, this can be done by checking for a record in the booked\_by but it is better practice to have a column for this specifically | |
| booked\_until | | DATETIME | The datetime that shows when the room is booked until | |
| booked\_by | | INTEGER  UNIQUE | The reference to a user that shows who has booked the room | |
| booked\_on | | DATETIME | The datetime that shows when the room was booked | |
| FOREIGN KEY (room\_type) REFERENCES rooms (id) | | | | |
| FOREIGN KEY (booked\_by) REFERENCES users (id) | | | | |
|  | |  |  | |

### Entity Relationship Diagram (ERD)

The green highlight references a foreign key in the table. The primary key of a table is indicated by the key icon next to the column name and is also the first column in the list and written in bold.

A screenshot of a computer

Description automatically generated

### Data Definition Language (DDL)

-- usergroups definition

**CREATE** **TABLE** usergroups (

id **SMALLINT** **NOT** **NULL**,

name **VARCHAR**(10) **NOT** **NULL**,

**PRIMARY** **KEY** (id),

**UNIQUE** (id),

**UNIQUE** (name)

);

-- users definition

**CREATE** **TABLE** users (

id **INTEGER** **NOT** **NULL**,

username **VARCHAR**(20) **NOT** **NULL**,

email **VARCHAR**(45) **NOT** **NULL**,

password **VARCHAR** **NOT** **NULL**,

date\_created **DATETIME** **DEFAULT** (CURRENT\_TIMESTAMP) **NOT** **NULL**,

usergroup **SMALLINT** **DEFAULT** **'1'** **NOT** **NULL**,

points\_gathered **INTEGER** **NOT** **NULL** **DEFAULT** **'0'**,

points\_redeemed **INTEGER** **NOT** **NULL** **DEFAULT** **'0'**,

**PRIMARY** **KEY** (id),

**UNIQUE** (id),

**UNIQUE** (username),

**UNIQUE** (email),

**FOREIGN** **KEY**(usergroup) **REFERENCES** usergroups (id)

);

-- loyalty\_rewards definition

**CREATE** **TABLE** loyalty\_rewards (

id **INTEGER** **NOT** **NULL**,

name **VARCHAR** **NOT** **NULL**,

icon **VARCHAR** **NOT** **NULL**,

cost **INTEGER** **NOT** **NULL**,

**PRIMARY** **KEY** (id),

**UNIQUE** (id),

**UNIQUE** (name)

);

-- loyalty\_rewards\_redeemed definition

**CREATE** **TABLE** loyalty\_rewards\_redeemed (

id **INTEGER** **NOT** **NULL**,

uid **INTEGER** **NOT** **NULL**,

rid **INTEGER** **NOT** **NULL**,

datetime\_redeemed **DATETIME** **DEFAULT** (CURRENT\_TIMESTAMP) **NOT** **NULL**,

**PRIMARY** **KEY** (id),

**UNIQUE** (id),

**FOREIGN** **KEY**(uid) **REFERENCES** users (id),

**FOREIGN** **KEY**(rid) **REFERENCES** loyalty\_rewards (id)

);

-- rooms definition

**CREATE** **TABLE** rooms (

id **INTEGER** **NOT** **NULL**,

name **VARCHAR** **NOT** **NULL**,

capacity **INTEGER** **NOT** **NULL**,

price **INTEGER** **NOT** **NULL**,

amount **INTEGER** **NOT** **NULL**,

**PRIMARY** **KEY** (id),

**UNIQUE** (id),

**UNIQUE** (name)

);

-- room\_bookings definition

**CREATE** **TABLE** room\_bookings (

id **INTEGER** **NOT** **NULL**,

room\_type **INTEGER** **NOT** **NULL**,

is\_booked **SMALLINT** **NOT** **NULL** **DEFAULT** **'0'**,

booked\_until **DATETIME**,

booked\_by **INTEGER**,

booked\_on **DATETIME**,

**PRIMARY** **KEY** (id),

**UNIQUE** (id),

**UNIQUE** (booked\_by),

**FOREIGN** **KEY**(room\_type) **REFERENCES** rooms (id),

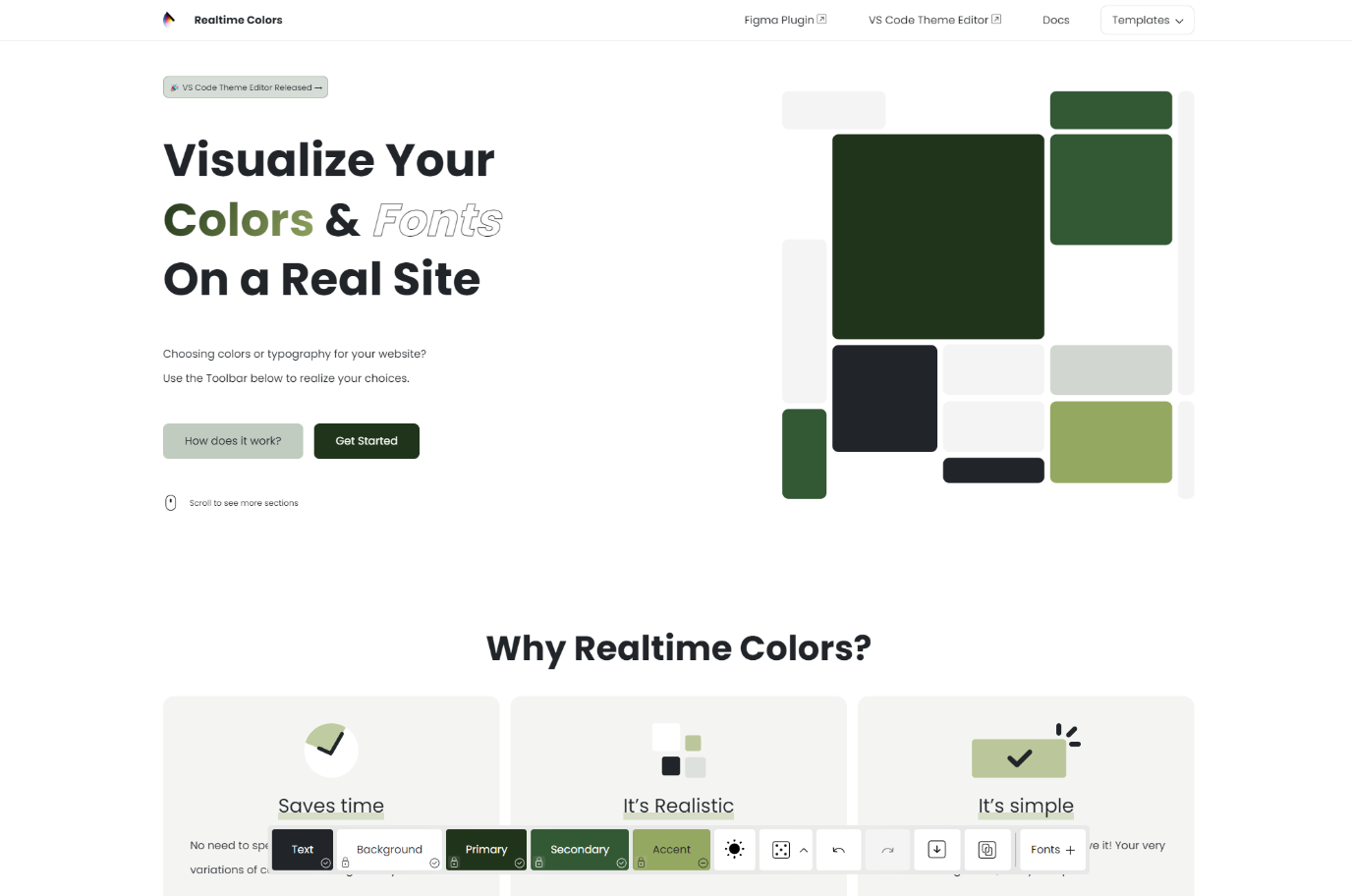
**FOREIGN** **KEY**(booked\_by) **REFERENCES** users (id)

);

## Wireframes

### Defining Colours

Source: [Realtime Colors](https://www.realtimecolors.com/?colors=212529-ffffff-20341b-335834-95a861&fonts=Poppins-Poppins)

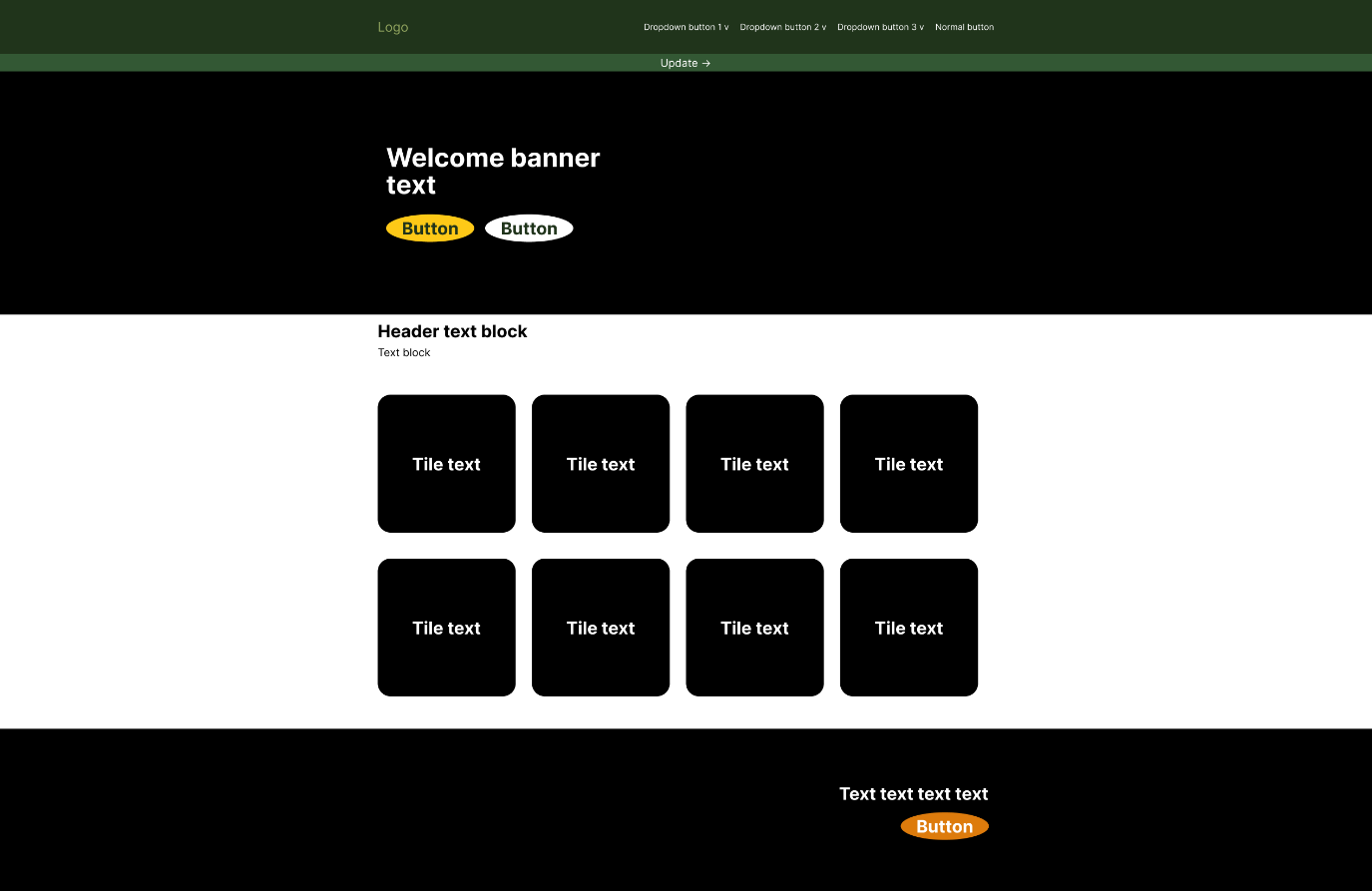


### Figma Wireframes

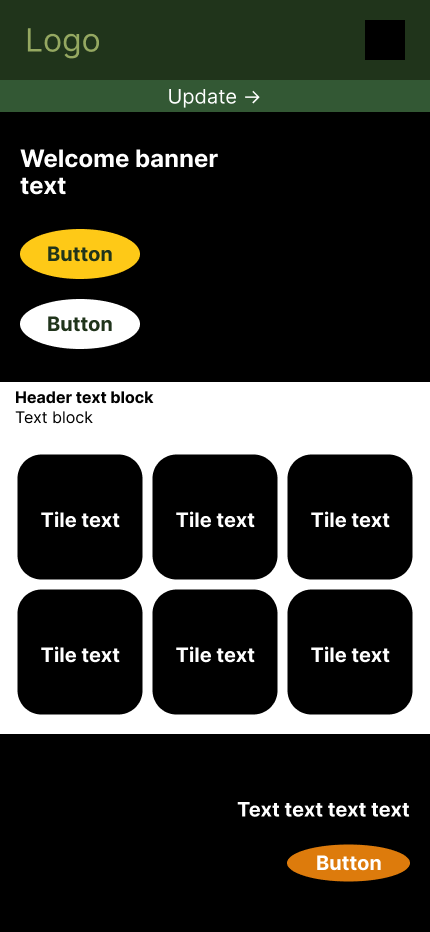
[Link to Figma Wireframes](https://www.figma.com/file/OzPuiQWTXvRR7sR1Itd0MP/Riget-Zoo-Wireframe?type=design&node-id=0%3A1&mode=design&t=QYk2xLYK9auXtYpQ-1)

#### Home page

##### Desktop



##### Mobile



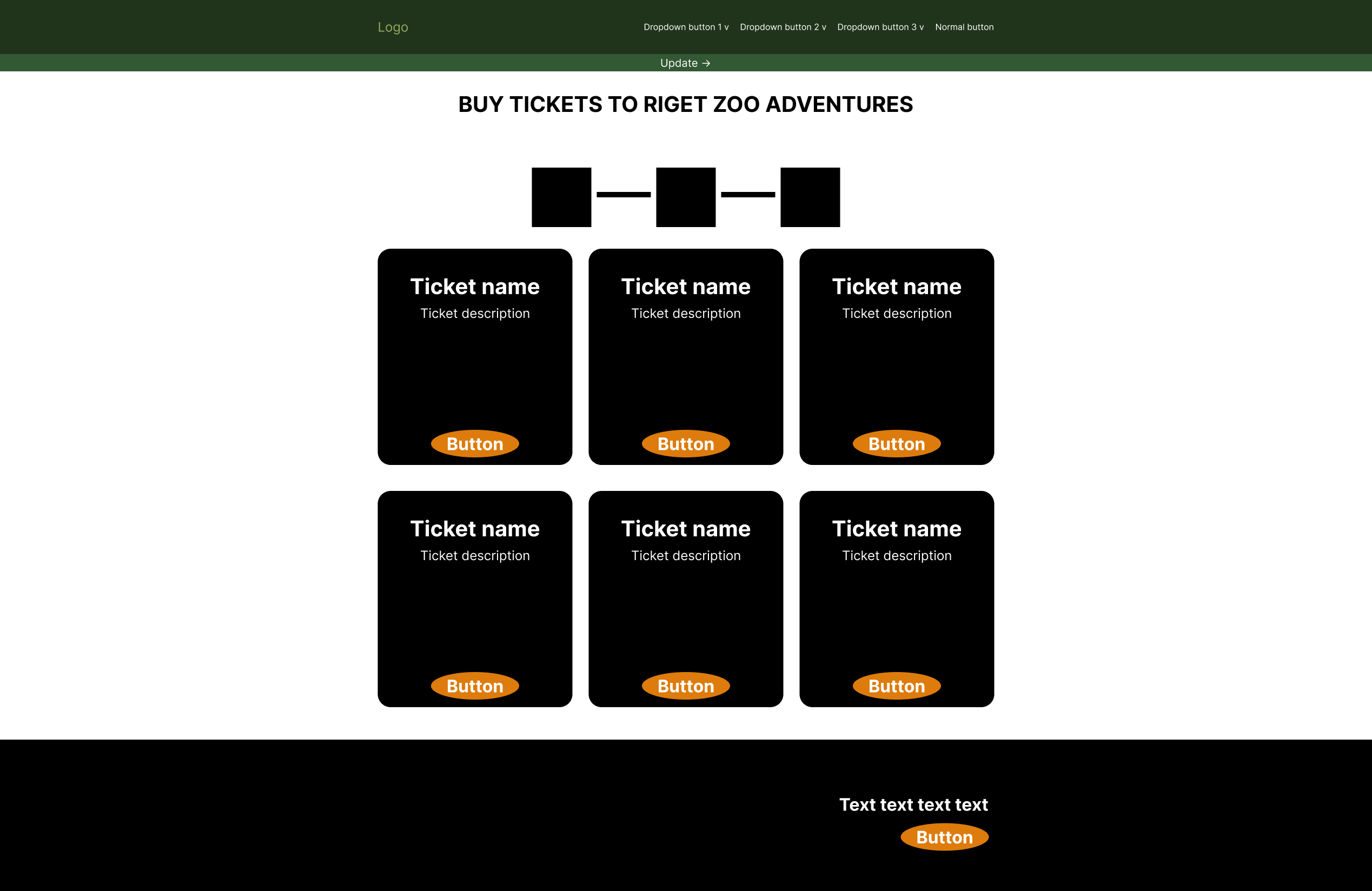
This is the wireframe for the homepage of the web app, the black sections are placeholders for images and all the text is placeholder text.

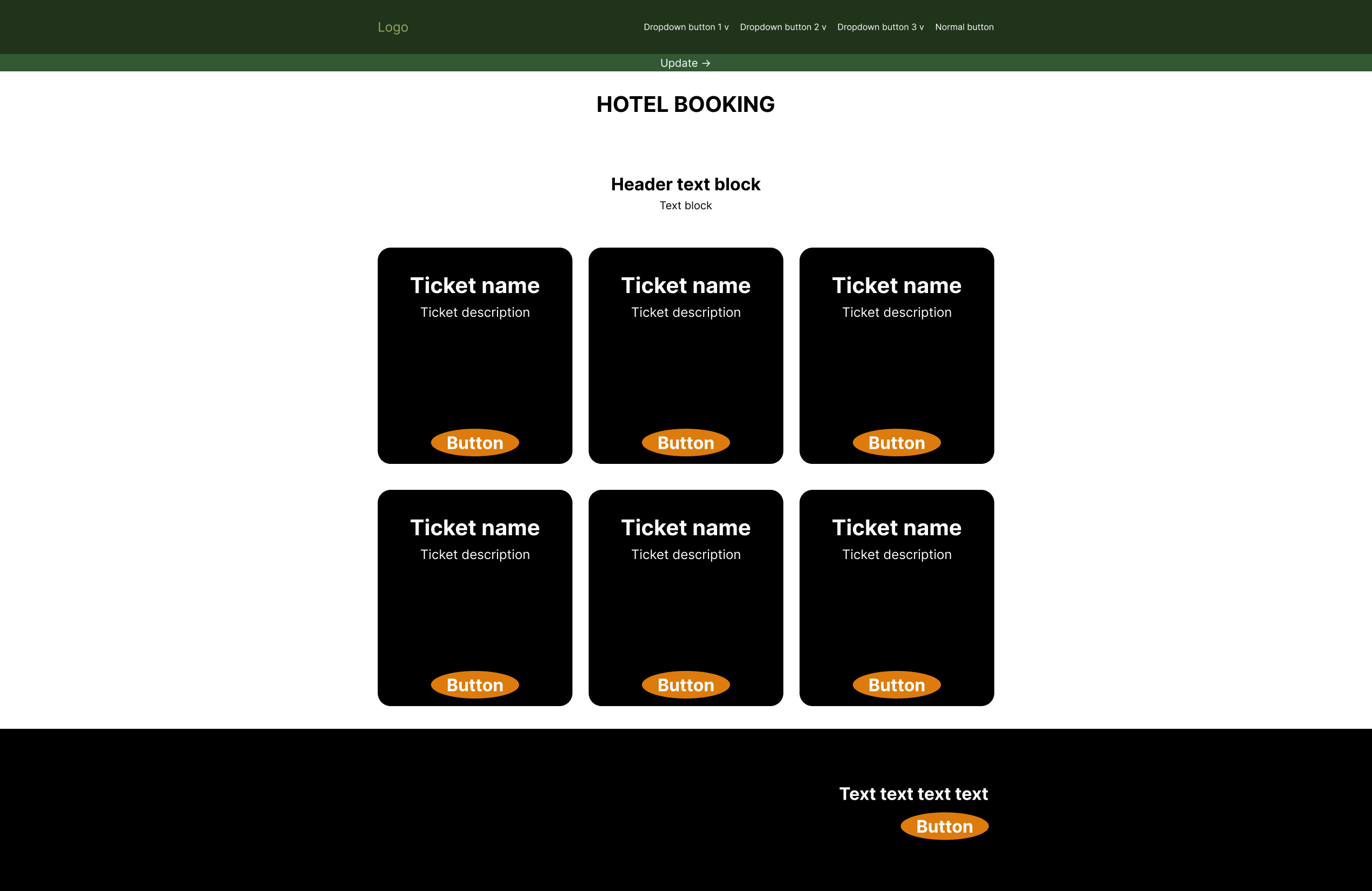
The “Logo” section is a placeholder for the Riget Zoo Adventures logo. The ellipse background on the buttons are placeholders for a similar shape that will be used – to be designed. The update part at the top would be a notification hyperlink that will be shown globally across the site. The tiles will be hyperlinks for different sections of the site. The dropdowns will not be hyperlinks but hovering over them will show hyperlinks. The last button in the navbar will be a hyperlink type of button for something link the user’s cart.

More sections on the home page can be developed as it will be put together in a scalable way. More button colours have been put into place and will be followed throughout the web app. All buttons with the ellipse type of background will be one of the three colours shown in the home page. All the components will be developed so that they can be used in HTML short handedly without having to put in the whole code again.

#### Ticket page

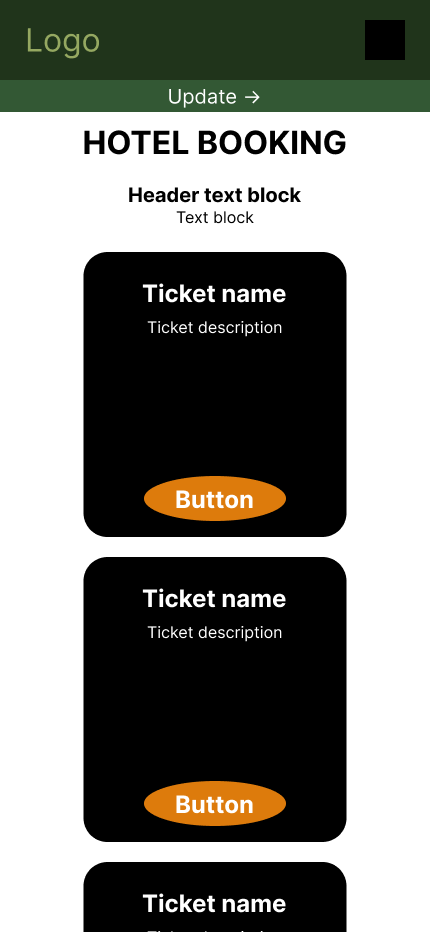
##### Desktop





##### Mobile



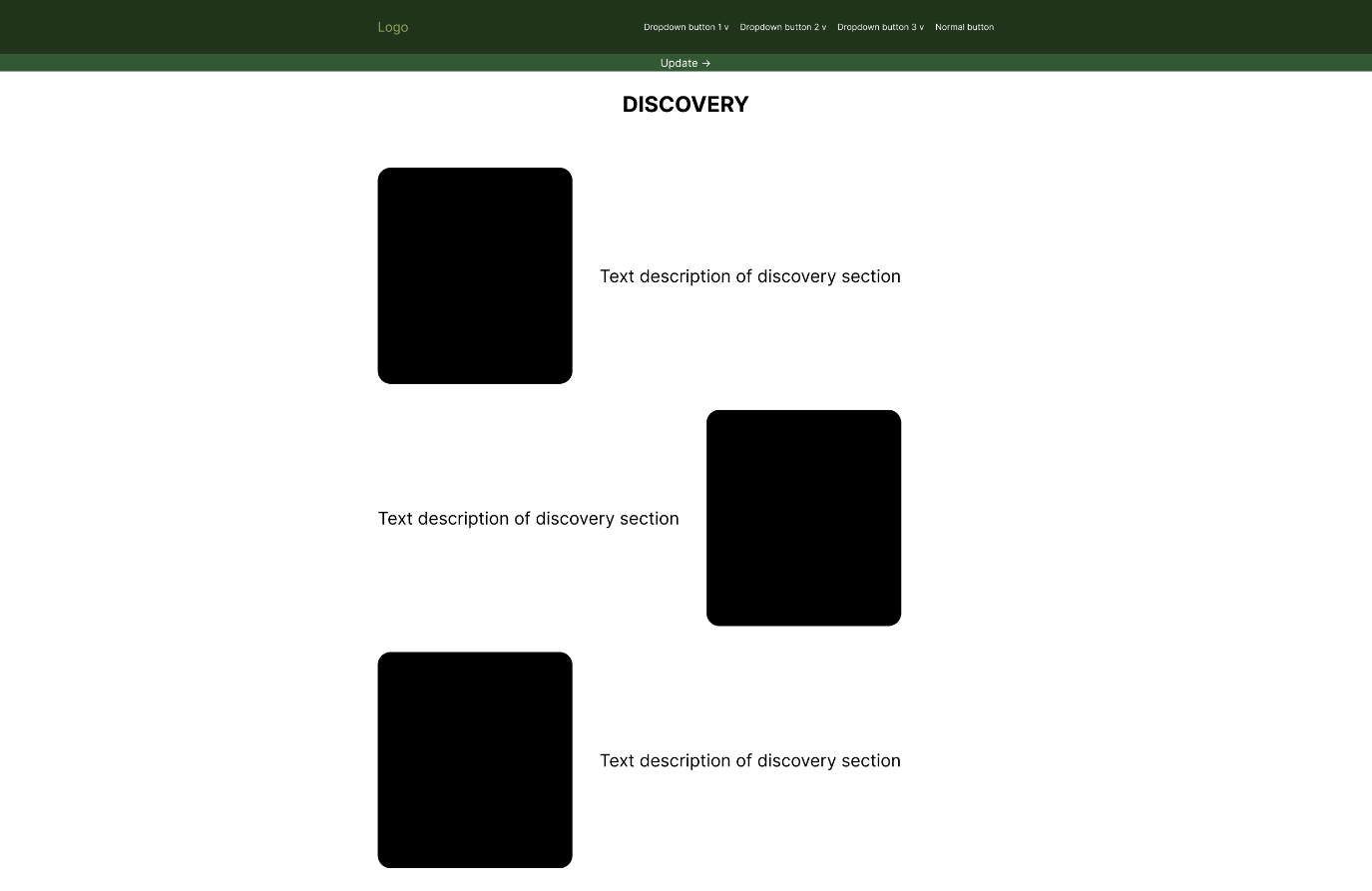


These are the wireframes for the ticket’s pages (safari tickets, on-site hotel tickets etc.) The black sections are placeholders for images and all the text is placeholder text.

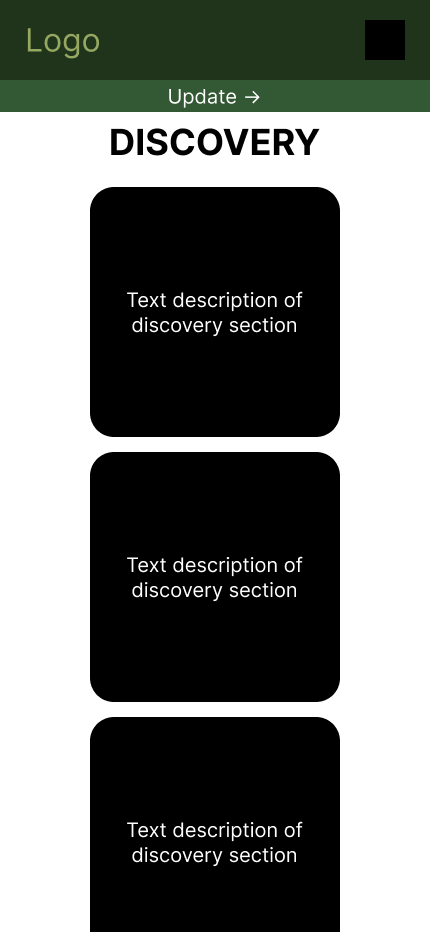
More sections on the page can be developed as it will be put together in a scalable way. All the components will be developed so that they can be used in HTML short handedly without having to put in the whole code again.

#### Discovery Page

##### Desktop



##### Mobile

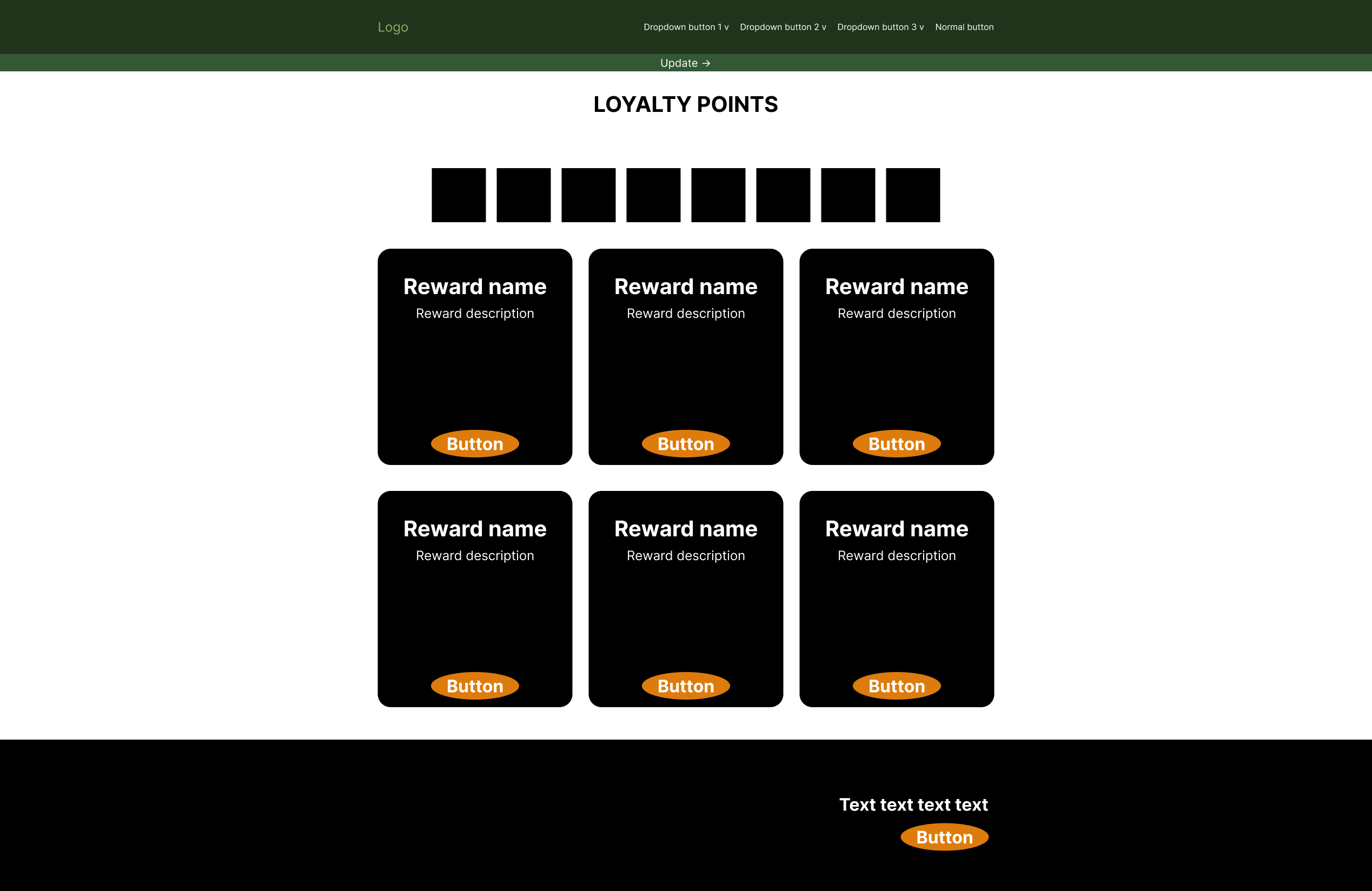


This is the wireframe for the discovery page of the web app. This page will show all the sections of the site on one page, the black sections are placeholders for images and all the text is placeholder text.

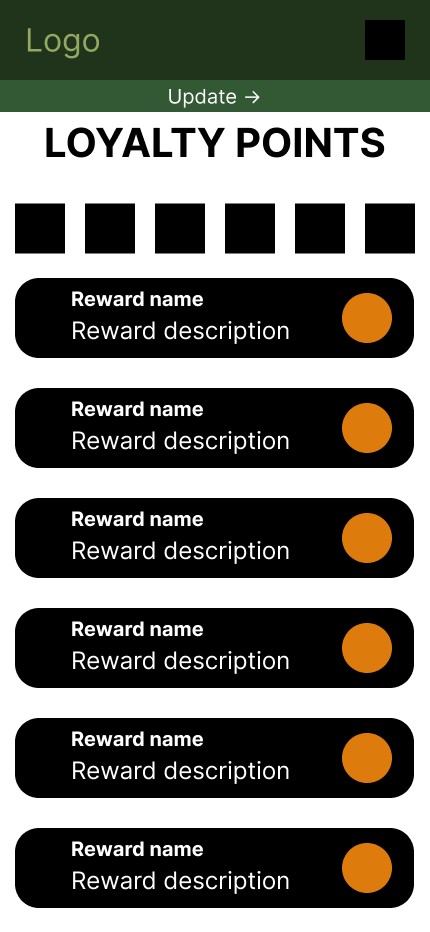
More sections on the page can be developed as it will be put together in a scalable way. All the components will be developed so that they can be used in HTML short handedly without having to put in the whole code again.

#### Loyalty Points Page

##### Desktop



##### Mobile

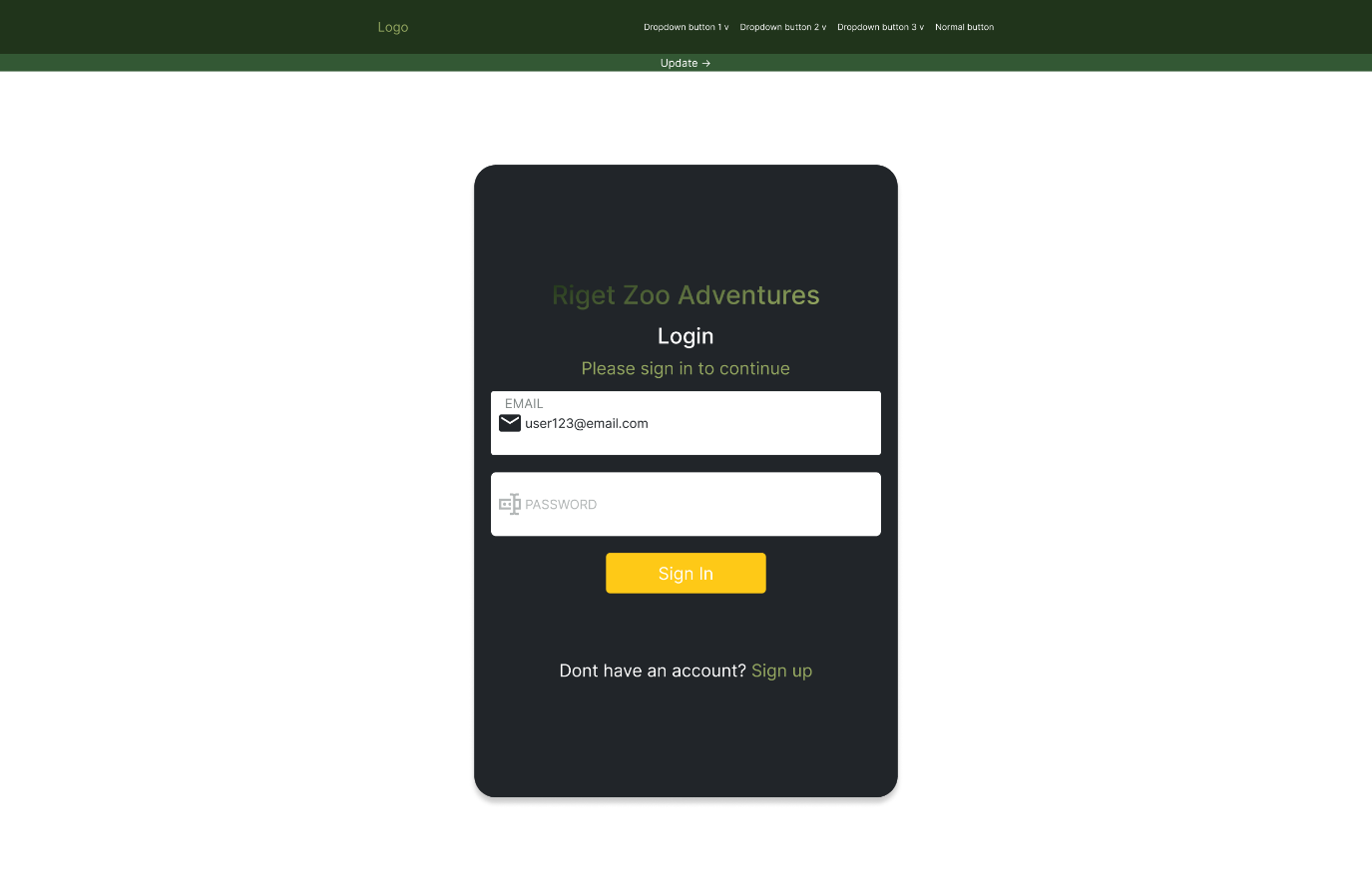


This is the wireframe for the loyalty points dashboard page of the web app. Each tile displays a reward that can be redeemed, the loyalty point cost will work similarly to the McDonalds loyalty reward system where you gather points as you get tickets and when you have a certain amount of points, they can be redeemed, some rewards will cost more than others e.g. you have 7 loyalty points and you redeem a reward that costs 5 points, if you redeem the reward and be left with 2 points. The points at the top may be displayed in a number format or as stars, the black squares are placeholders for stars, but this is subject to change.

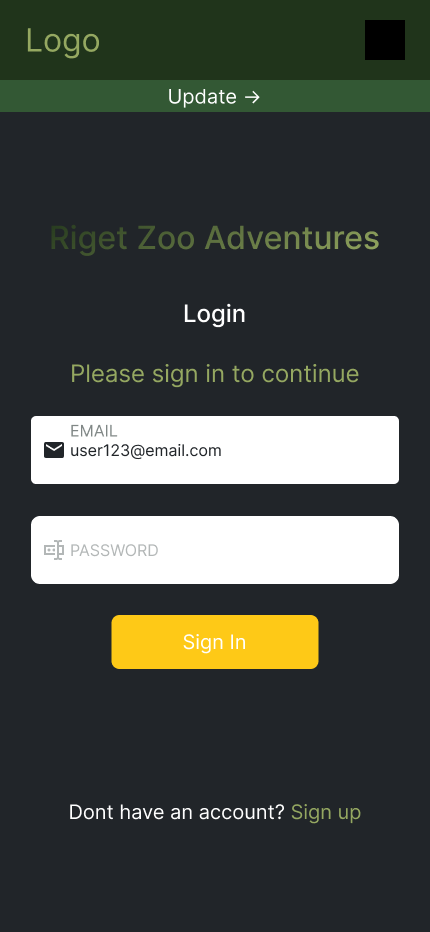
More sections on the page can be developed as it will be put together in a scalable way. All the components will be developed so that they can be used in HTML short handedly without having to put in the whole code again.

#### Login Page

##### Desktop



##### Mobile

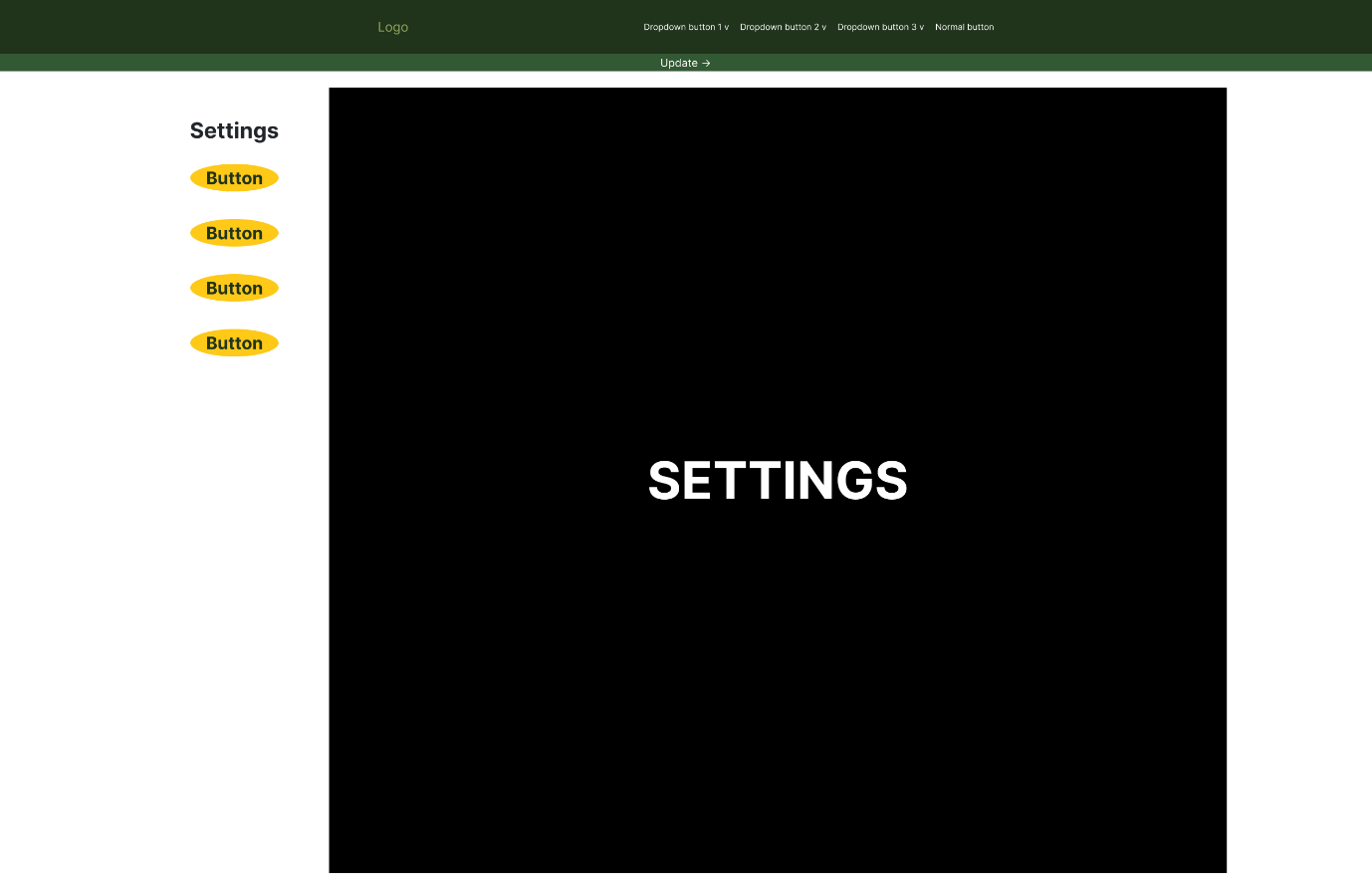


This is a simple login system that allows users to login. It also gives the option for the user to sign up if they have not got an account yet. Employee account integration will be added once the RZA database system is integrated with the site. This is done for data protection reasons as it is possible that a user could try to create an account as a employee (if that were possible) causing them to have access to features that they should not have access to. The same applies when creating admin accounts. Remember me will be integrated at a later stage and works by storing your login token for a set time instead of your browser handling that.

More sections on the page can be developed as it will be put together in a scalable way. All the components will be developed so that they can be used in HTML short handedly without having to put in the whole code again.

#### Settings page

##### Desktop



##### Mobile

To be developed

This is the settings page; this is where the user will be able to change their settings. The cause for low fidelity on this page is that the settings that will be necessary for users is currently unknown and will become clear throughout development, this design gives us a general layout for the design and can be developed on from this design.

### Design Guidelines

Text colour: **#212529**

Background colour: **#FFFFFF**

White colour: **#FFFFFF**

Primary colour: **#20341B**

Secondary colour: **#335834**

Accent colour: **#95A861**

Font family: "Inter", sans-serif;

Transition: all 0.3s ease 0s

Not every page’s design is known which results in less pre-project designing but more can be wireframes can be designed throughout the project. From the colours, transitions, font family etc. we can gather enough information for design guidelines that will be used throughout the project. More guidelines will be set as the project progresses to ensure maximum UI and UX quality.

### Flowchart for various pages

## Testing

### Types of tests

#### User Acceptance Testing (UAT)

User Acceptance Testing is essential to ensure that the web application meets the requirements and expectations of the end-users. This testing involves real users engaging with the application to validate its usability, accessibility, and overall user experience. Testers will perform tasks such as logging in, accessing tickets, and utilising the dashboard features to ensure that the application meets their needs effectively.

#### Accessibility Testing

Accessibility testing is vital to ensure that the web application caters to users with diverse needs. Testers will assess the application's compatibility with assistive technologies, screen readers, and keyboard navigation. They will also evaluate features such as adjustable font sizes to ensure equal access to information for all users.

#### Penetration Testing

Penetration testing is essential to identify and address vulnerabilities that could compromise the confidentiality and integrity of user data. Testers will perform penetration testing to identify potential security loopholes, assess data encryption methods, and validate user authentication processes. This testing aims to ensure that the web application adheres to best practices for data protection and privacy.

#### Functionality Testing

Thorough functionality testing is necessary. Testers will validate that admins have access to administrative features to oversee the application's functionality and content, while users enjoy a seamless ticketing and reward scheme focused experience. This testing will involve verifying the segregation of features based on user roles and ensuring that each user type has access only to the relevant dashboard features.

### Types of data that will be used

Erroneous Data

Testing with erroneous data involves simulating scenarios where the application receives incorrect or unexpected input. This type of testing ensures that the application can gracefully handle unexpected data and prevent system failures or incorrect outputs.

Normal Data

Normal data testing involves using typical, expected inputs to validate the standard behaviour of the application. For instance, in user acceptance testing, normal data would include typical user interactions such as logging in, accessing ticketing information, and utilising dashboard features. This type of testing ensures that the application functions as intended under regular usage conditions.

#### Boundary Data

Boundary data testing involves testing the application's behaviour at the edges of input ranges. This type of testing helps identify potential issues related to data limits and extremes.

Security Data

Security testing involves using several types of data to simulate security threats and vulnerabilities. For example, in security testing, testers may use simulated malicious input to assess the application's resilience against potential attacks such as SQL injection or cross-site scripting. This type of testing helps identify and address security vulnerabilities that could compromise user data and system integrity.