Nottinghamshire County Show

Mobile Application

Project Report

Jacob Boddey, Chin Tong Fung, Thomas Harrison, Jordan Rudge, Rishi Singh, Will Treston

Abstract

This is a report that details the process our group went through to create an app that people who were attending the Nottinghamshire County Show could download and use. The purpose of this app was to provide its users with information about the various events, exhibitors and competitions happening at the show. The project started as a prototype app for the 2019 Midlands Machinery show which was developed by Jacob. From there we fine-tuned and tailored various aspects of the app to better suit the needs of the County Show as a group of six. Unfortunately, due to Covid-19 the event was cancelled, and the app was not used this year, however it may be used next year.

Contents

1 Introduction

1.1 Background

Over the past year Nottingham Trent University (NTU) have built a working relationship with Nottinghamshire and Newark Agricultural Society as the vice-chancellor of NTU holds the appointment of president for the agricultural society. This has meant that both parties are benefitting by overcoming issues by sharing skills and experience with each other.

1.2 Client

The client of this project is Nottinghamshire and Newark Agricultural Society. As a registered charity they support the agricultural industry and practices within the region that they cover. They do this by utilising their assets (notably Newark Showground) in order to provide networking events, to support businesses and individuals and to raise money for various schemes ran by the charity. The agricultural society organise and manage many events including the Midlands Machinery Show and the Nottinghamshire County Show (Newak and Nottinghamshire Agricultural Society, 2019).

The agricultural society currently have little technology in use at the showground. However, current infrastructure and tools used consist of a satellite provided WiFi point (accessible over the entire site) and a third-party event management tool called EventHalo. After communicating with the team at the showground we have found that they are aware of the lack of technology and are very keen to implement technology to make their current processes much more efficient as well as providing a better experience for the visitors and exhibitors of the showground.

Last year a small team of students created a prototype mobile application for the Midlands Machinery Show to prove the concept of implementing mobile technology at the Newark Showground to assist visitors during their experience at an event. Features that were most successful were the map and the list of exhibitors. Overall, this project proved to be a success and has paved the route for future mobile application development for the showground.

Looking forward, the Nottinghamshire County Show is perhaps the biggest event at the showground each year with a very high visitor count. Similar to other county shows throughout the country, we believe that a mobile application would be beneficial for the visitors by providing navigation around the site – allowing visitors to find the stands they wish to visit more easily – as well as many other features that would make information more accessible.

2 Survey of Existing Solutions

2.1 Brecon Agriculture County Show

Features:

- Shows event times
- Detailed showground map
- Information about the exhibitors

Strengths:

- Requirements for device specs are low
- Intuitive map
- Able to add multiple favorite exhibitors
- Shows where and where the events are held
- Able to switch units (Km to M)
- Have multiple languages
- The app is smoothly run with little bugs encountered
- The map shows where you are, so you will not get lost

Weaknesses:

- Not available on Android
- Only one sorting option for exhibitors
- Some pictures of the exhibitors are missing
- The exhibitor's page is not categorized
- The map is not clear and messy
- The about page is impractical
- The event page is not Live Updated
- The events time are all incorrect
- Some of the exhibitor's information is missing
- No sorting options for the favorite page
- The map has a low definition
- The app is always tracking you even if it is not on

2.2 Lambeth Country Show 2019

Features

- Discover: Exhibitors/events are presented in a grid format with each tile displaying a title and image.
- Schedule: Each day contained in its own tab.
- Artists and Activities: organised in a list alphabetically. Can be filtered by category.
- Exhibitors: organised in a list alphabetically, or by proximity. Can be filtered by category.
- Social: Displays pages for the show's Instagram, Facebook, and Twitter.
- Information: Contains FAQ, details about the show itself, contact info, credits, etc.

Strengths

- Pages for social media are displayed within the app; users can switch between tabs to view the different social media pages.
- Exhibitor list contains large images that mostly do a good job at illustrating what each of them is about.
- Search feature allows users to quickly find an event by name.
- Schedule can be sorted by time or place or shown as a timeline.
- Multiple maps which also display the surrounding area.

Weaknesses

- Colour scheme (namely the title bars) may be considered too loud.
- Timeline view only has gridlines for each hour – could be for every half or quarter hour.
- Some categories may be too specific, only containing one or two entries.

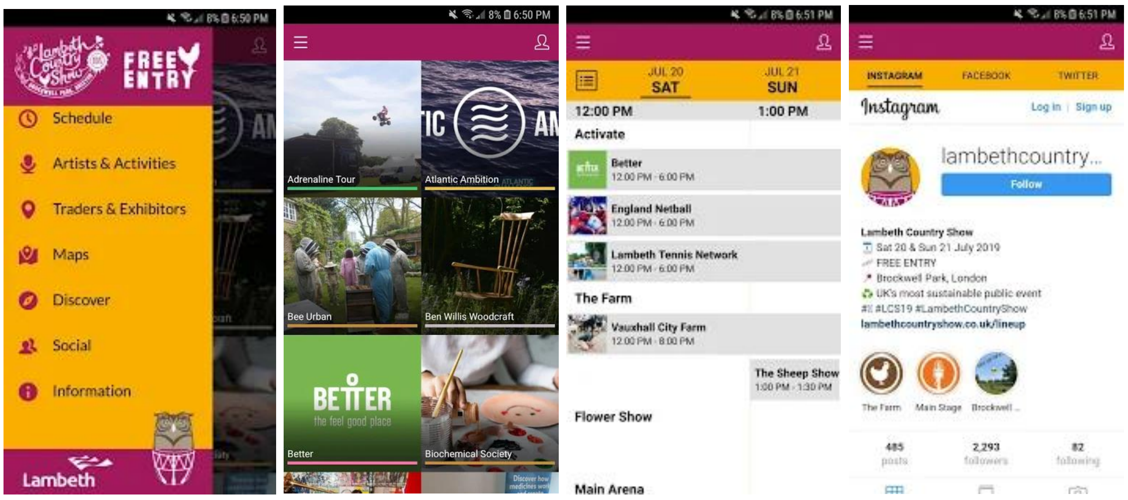


Figure 1: Lambeth County Show

2.3 Statistics on App Usage Among Various Demographics

- 2.8 million apps on play store and 2.2 million on Apple store.
- 21% of millennials open an app 50+ times a day.
- 49% of people open an app 11+ times a day.
- 57% of all digital media usage comes from mobile apps.
- Average smartphone user uses 30+ apps each month.
- 18-24-year olds spend 66% of digital media time on mobile app usage.
- Millennials are more willing to pay for apps. ⅓ buying 1+ a month.
- In 2017, 21 percent of millennials said they deleted a mobile app because they did not like the app logo.

(Blair, 2019)

(Mindsea, 2019)

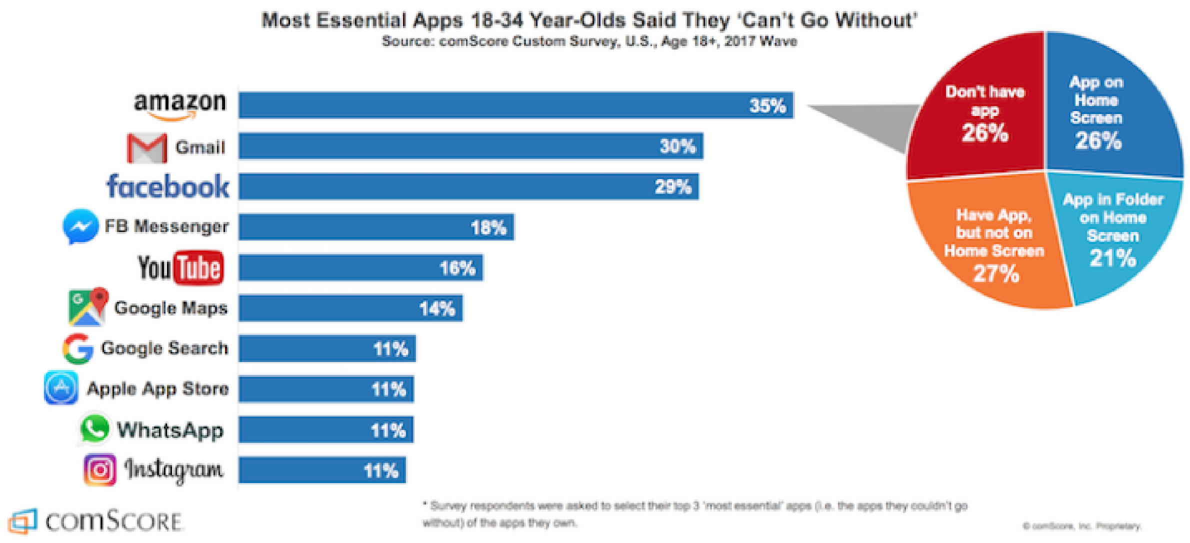


Figure 2: Most Essential Apps 18-34 Years-Olds Said They Can't Go Without

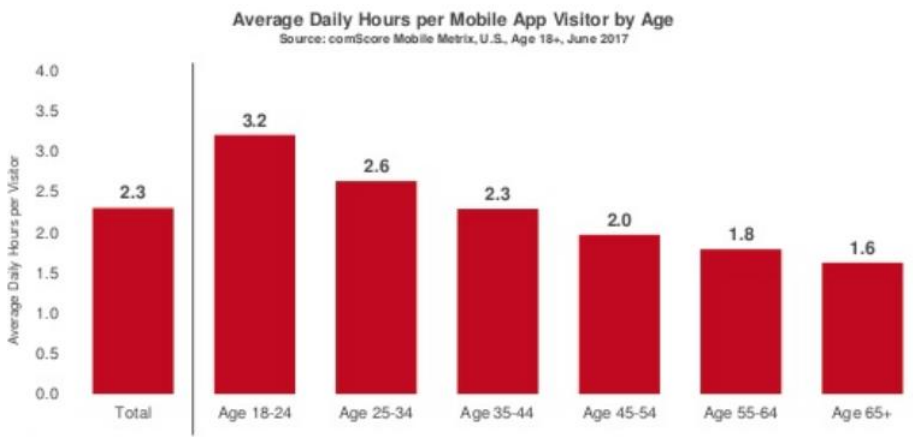


Figure 3: Average Daily Hours per Mobile App Visitor by Age

These statistics show most of the people using mobile apps are among the millennial demographic and showing a positive correlation with younger adults and digital media usage. Therefore, it is fair to conclude that teenagers and young adults are the more likely of visitors to the county show to be using the application. As the Nottingham County Show is a family event, assuming effective advertisement, there is a good chance that younger members visiting will download the app and use it to aid them and their respective families when planning their day there.

3 New Ideas

3.1 The Aim

The aim of this project is to design and implement an application which will be suitable for the Nottinghamshire County Show – it will follow the design principles of other materials used for the event - especially for the visitors so that it can enhance their event experience. This application should provide an overall showground map with a filter system, so it is easier for the guest to search for what they want to see and have access to their preferred areas easily. The application's home page design (GUI) should have an intuitive design that is suitable for everyone. It is also important that the organisers can use the program to provide real–time information updates about their activity/program for the user. The application should have a timetable for the activities and programmes that is happening on that day. Unlike the normal timetable, this one can allow the organisers provide real time updates to it, so if they want to change their initial activity/program's time the user will get updated too.

Members chosen to make this program and their role is listed below within this report. It is our responsibility to ensure the final product complies with all the Nottinghamshire and Newark Agricultural Society's criteria and that it is ideal for the consumer for its design and functionality.

3.2 Functional Requirements

The following functional requirements have been derived from the previous mobile application prototype used at the Newark Showground. A previous interview with staff from the showground allowed us to form an initial list of requirements whilst the testing and feedback from the prototype has developed the existing list of requirements. Those requirements that could be implemented will be developed into the application where and when possible, provided that the crucial requirements will be met and unaffected.

The mobile application must:

- Show a splash screen on launch
- Display an interactive map of the showground
- Allow location markers to be toggled on and off
- Show the locations of important utilities
- Provide information about utilities such as the name and type
- Provide a list of exhibitors
- Include important information about exhibitors
- Show the location of exhibitors
- Allow exhibitors to collect contact details of visitors or vice-versa
- Provide a list of scheduled events
- Include information about events such as description, the presenter and timings
- Allow the user to save their parking location
- Show the user where they have parked their car if saved when required
- Display details about the various competitions taking place

- Display notifications sent by administrative users on the web portal in real time
- Display live social media feeds from the showground's social media accounts
- Show information about the application (authors, version, description)

The mobile application could:

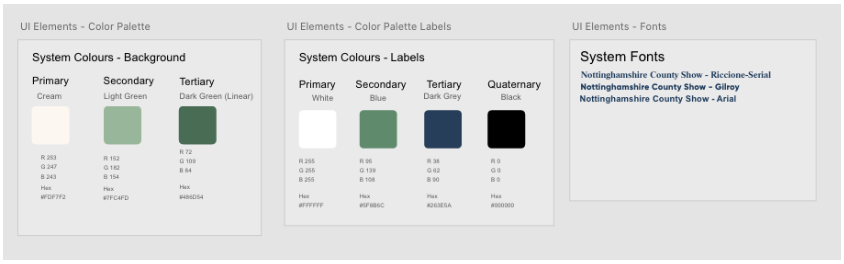
- Show the user's location on the interactive map
- Allow communication by message between the visitors and exhibitors
- Allow registration and ticket purchasing for the event
- Show live results for the competitions
- Provide an interactive activity for younger users of the application

By analysing the client's needs, existing solutions, and the technology available, we have built a product proposal which we believe will solve the problem and meet the requirements set by the client.

The project is split into three main components within an integrated system to provide up-to-date data, an attractive design and functionality.

3.3 Mobile Application

The mobile application will be a cross-platform application built using the Flutter framework. The aim of the mobile application will be to display information on an interactive map to the user with additional information for individual exhibitors. In order to achieve the required functionality, Flutter plugins will be used. These will include a map plugin and the http plugin – which allows for communication on the http protocol. Designs will be done in Adobe XD and will follow design patterns and colour schemes from the new Nottinghamshire County Show website.



UI Elements - Color Palette

System Colours - Background

Primary	Secondary	Tertiary
Cream	Light Green	Dark Green (Linear)
R 253 G 247 B 243	R 152 G 182 B 154	R 72 G 109 B 84
Hex #FDF7F2	Hex #7FC4FD	Hex #486D54

UI Elements - Color Palette Labels

System Colours - Labels

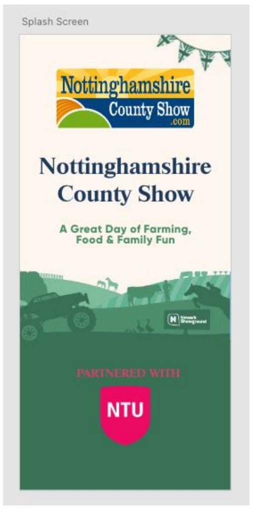
Primary	Secondary	Tertiary	Quaternary
White	Blue	Dark Grey	Black
R 255 G 255 B 255	R 95 G 139 B 108	R 38 G 62 B 90	R 0 G 0 B 0
Hex #FFFFFF	Hex #5F8B6C	Hex #263E5A	Hex #000000

UI Elements - Fonts

System Fonts

Nottinghamshire County Show - Riccione-Serial
Nottinghamshire County Show - Gilroy
Nottinghamshire County Show - Arial

Figure 4: Colour Palette and Branding and Splash Screen


Splash Screen
Nottinghamshire County Show .com
Nottinghamshire County Show
A Great Day of Farming, Food & Family Fun
PARTNERED WITH
NTU

3.4 API

The API will be used by the app to get the data from the database to the app. It will be a restful API and require no authentication as it does not return any sensitive information, however it will have a rate limiting feature to ensure that malicious users don't try and overwhelm it.

Below is the requirements list for the API:

The API must:

- Return accurate requested data
- Be able to handle large numbers of requests simultaneously without a decrease in speeds

3.5 Admin Panel

We wanted the admin panel to be familiar to the Newark Showgrounds current website to allow for a sense of familiarity for the people who would be using the admin panel. For this reason, we decided to base its design from https://www.newarkshowground.com/. One idea we had to increase the ease of use when creating the location for a Zone is using a map on which a user can click to get the longitude and latitude of the zone's location. This would then get stored in the database. Finally, one of the concerns expressed by the client was the fact that her team have limited technical capabilities. Due to this reason we aim to make the website as simple to use as possible so there is no learning curve involved which may take away time from other aspects of the user's jobs.

Below is the requirements list for the Admin Panel

The web portal must:

- Protect content with a secure authentication form (email and password)
- List the events occurring at the showground
- Allow events to be created and saved in the database
    - As well as edited and deleted
- Allow administrative users to send real time messages to visitors in real time
- Display social media keyword tracking in real time

The web portal could:

- Provide a preview of the mobile application
- Show usage statistics of the mobile application
    - Including the live location of visitors (traffic monitoring)

4 API and Admin Panel Design & Development

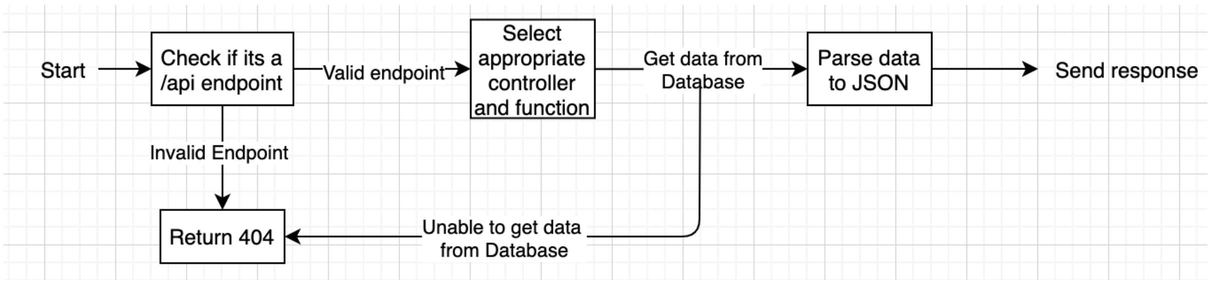
The backend was split into two separate parts for the purposes of this project – the API and the Admin Panel. The design phase consisted of creating flow diagrams, researching the various technologies we had available to us and creating a mock-up of our user interface.

4.1 Flow Diagrams

We designed flow diagrams to describe our system in a visual way. In total we designed three diagrams. One which describes the user interface and two which describe the two parts of the backend – the API and the Admin Panel.

4.1.1 API Flow Diagram

As our API is only required to GET information, and none of the information we store is personal data, we did not need to implement a way of authorising the sender of the requests we were retrieving. This made the flowchart for the API part of the backend simple. It can be seen below:

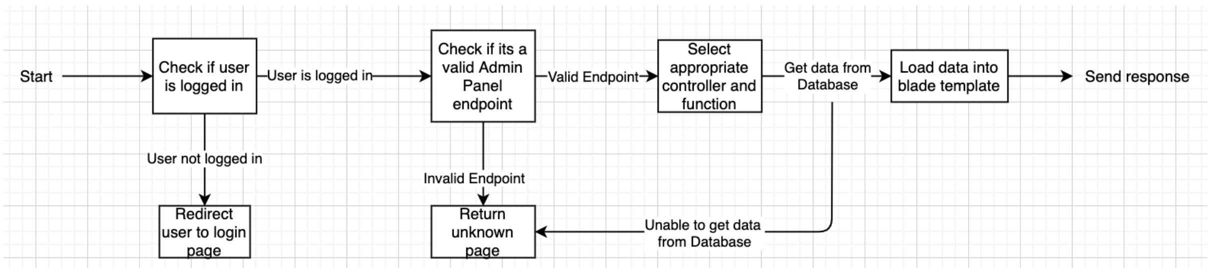


Start
Check if its a /api endpoint
Valid endpoint
Select appropriate controller and function
Get data from Database
Parse data to JSON
Send response
Invalid Endpoint
Return 404
Unable to get data from Database

Figure 5: API flowchart

4.1.2 Admin Panel Flow Diagram

The administration panel required a slightly more sophisticated flow diagram as an authorisation process was put in place. As our system does not make use of different levels of user permissions, once you were logged in you were able to perform all commands. This meant that we only needed to add one additional step to the flow chart (user authentication). The final flow chart can be seen below:



Start
Check if user is logged in
User is logged in
Check if its a valid Admin Panel endpoint
Valid Endpoint
Select appropriate controller and function
Get data from Database
Load data into blade template
Send response
User not logged in
Redirect user to login page
Invalid Endpoint
Return unknown page
Unable to get data from Database

Figure 6: Admin Panel Flow Diagram

4.1.3 Admin Panel Front End Flow Diagram

We designed a flowchart that visually displays how a user can interact with the administration panel. The flow chart below shows all the options a user has when on a specific page. One thing to note is that on any given page, the user can:

- View the dashboard
- View and create events, event locations, exhibitors, zones and competitions pages.

These additional options are done through the navigation bar and are **not** displayed in the flowchart below as it would have made the chart difficult to read. All the sections within the dotted rectangle require authentication. If a user tried to access these without being logged in, they will be redirected to the login page.

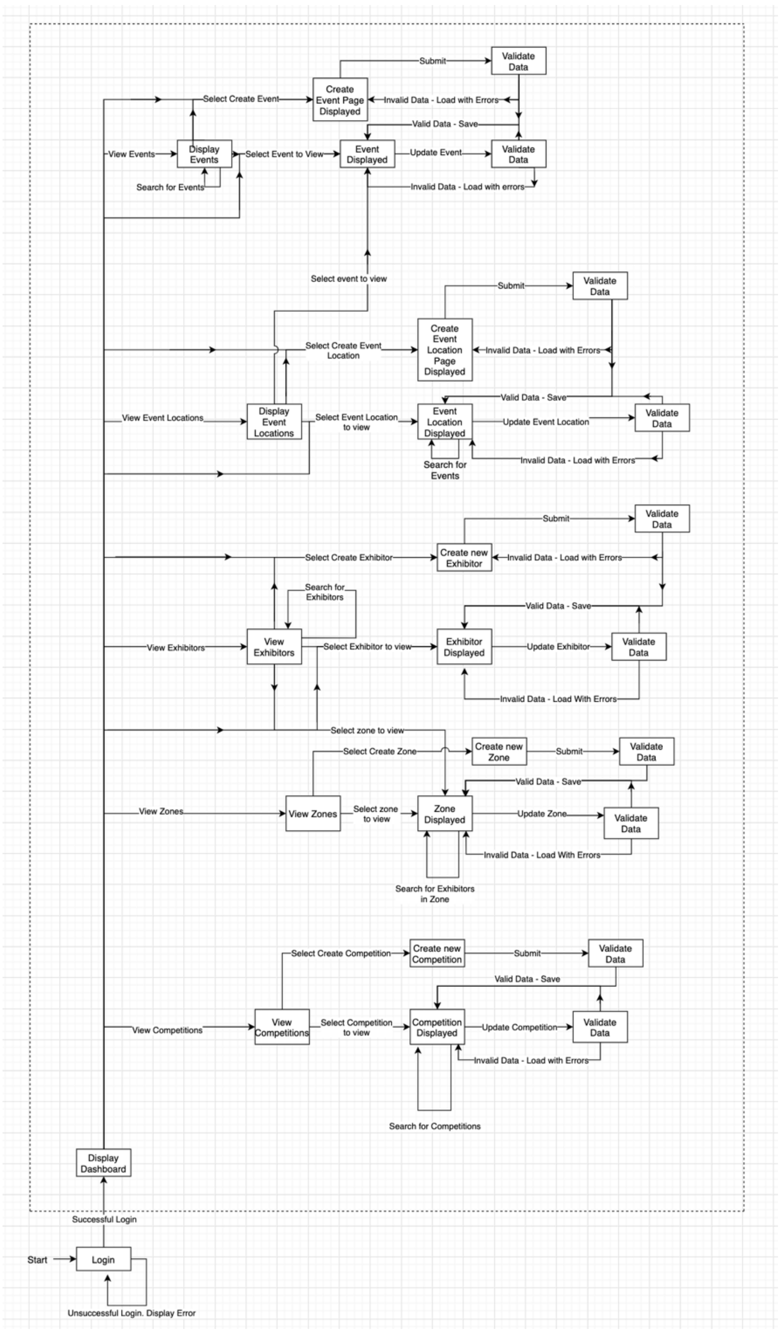


Figure 7: Admin Panel Frontend Flow Diagram

4.2 API and Backend of Admin Panel Research

Prior to the start of the development phase of the API and Admin Panel, we researched and talked about the various technologies that would allow us to implement the features in the way we wanted, in the timeframe we had. We narrowed down our choices to four options: Django and Flask which are both Python based, Spring Framework, which is Java Based, Node coupled with Express which is Javascript based, and finally Laravel. Below is a table which summarises the main points that are for and against each technology. These main points are what influenced our final decision

NAME OF TECHNOLOGY	LANGUAGE	PROS	CONS
DJANGO	Python	- Previous experience with Django - Fast to develop in - Built in Admin panel	- Django is very monolithic - Templating errors are silent and therefore hard to find
FLASK	Python	- Easy URL routing - Very flexible	- Large learning curve - Very “barebones”
SPRING FRAMEWORK	Java	- Easy to Develop - Easy testing	- Large deployment file due to unused dependencies - Limited control of the application
NODE WITH EXPRESS	JavaScript	- Easy Routing - Large number of public packages available - Very well documented	- As Node is asynchronous due to the large number of expected users making API calls, there may be issues with the amount of callbacks being made. - Is not multithreaded which could cause performance issues
LARAVEL	PHP	- Easy URL routing splitting web routes and API routes apart. - Quick development speed as the “outline” of functions and classes is auto-generated	- Requires a lot of dependencies – composer, blade etc - Does not work with shared hosting without making a patch for it.

After weighing up our options and carefully taking into consideration the points above and various other points, we decided we would use Laravel. Laravel implements a Model, View, Controller software design pattern which makes developing in it very easy. For every type of object in the database you create a model, which allows you to interact with it in an object oriented manner, you create a controller that allows you to manipulate data and control the objects, and finally implement views to display the information to your users. The biggest factor which influenced our choice to use Laravel was the ease of splitting up the API part of the backend, and the Web Panel side. The way Laravel allows us to do this is by having different routing files for each (web.php and api.php) and allows us to have different controllers for each part. Laravel also makes it very easy to add new parts to both the API and the Web Panel, by pre-generating the stub definitions of the functions you want to make. Finally, the templating language used by Laravel is blade. When used with Laravel, very detailed error reports are produced which again, increases development speed as minimal time is spent debugging.

4.3 Admin Panel Frontend Research

From the start of the design phase, we knew that we wanted the Admin Panel to resemble the Nottinghamshire Showground's current website. The reason we decided on this was simply because the people that would be using the Admin Panel would be familiar with the layout and feel of the website. For this reason, we decided that we wouldn't need to research any Frontend Frameworks (as their site doesn't use any). However, the Nottinghamshire Showground's website uses the BootStrap CSS Framework. We were fairly certain that we would use Bootstrap as our choice when designing the layout, however we still wanted to explore our options. After looking at various options, we decided to narrow it down to three choices: BootStrap, Semantic UI and Foundation. The table below outlines the pros and cons of each option.

NAME OF TECHNOLOGY	**PROS**	**CONS**
BOOTSTRAP	- Very well documented and has a large community surrounding it - Very easy to use grid system	- BootStrap websites look alike (this is a general negative point of BootStrap – we want it to look like the Showgrounds website) - Is very heavy so loading times may be increased
SEMANTIC UI	- Easy to use from the start - Lightweight	- No experience with Semantic UI - Relatively small community
FOUNDATION	- Good gridsystem - Has easy-to-use widgets	- Steep learning curve - Smallest community out of the three

BootStrap is the option we decided on after carefully looking over the three choices we had narrowed down to. As mentioned above, it was the logical choice to make as we wanted to replicate the website of the showground as closely as possible.

One of the features that the admin panel will have is the ability to choose the exact coordinates of where you want the marker for each zone to be on a map. For this we knew we were going to use MapBoxJS from the start as we had had previous experience in using it making it the obvious choice.

4.4 The Database/ERD Design

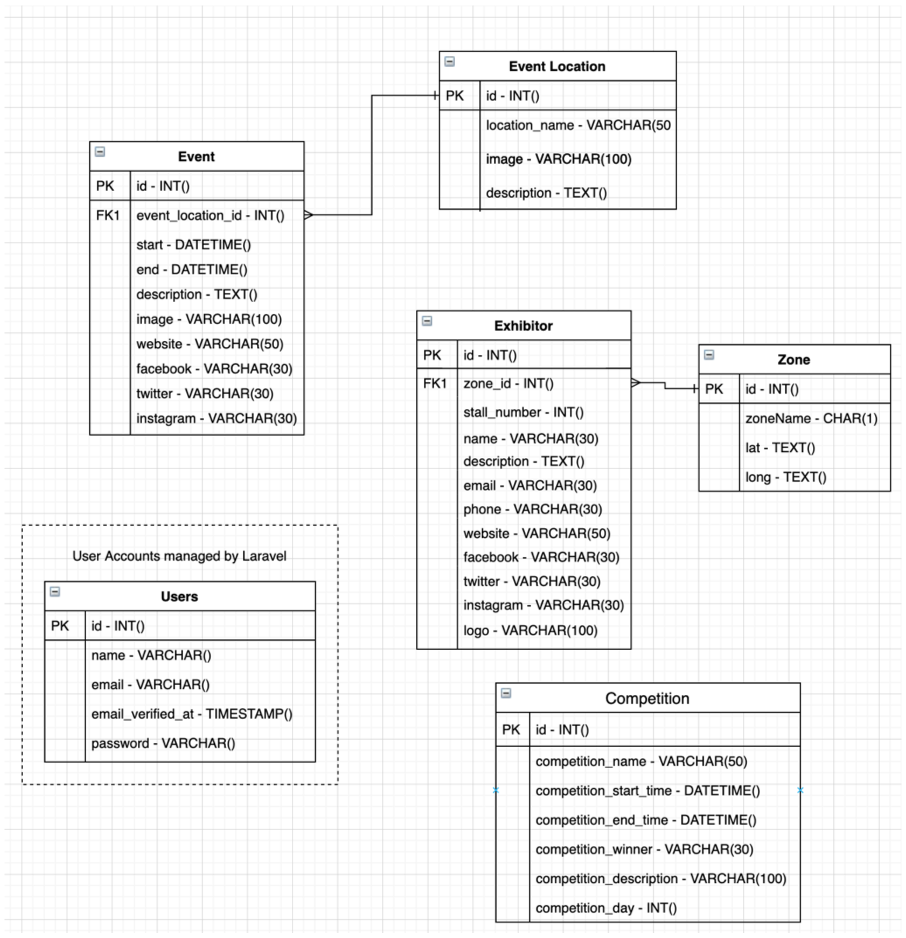


Figure 8: ERD Design

The first thing we designed was the database. We followed the rules of normalisation to ensure that our data stored as little duplicate data as possible. One of the requirements we needed to meet was user authentication before they can access the site as Laravel has this feature built in and automatically generates the controllers and the database tables for this, we didn't have to worry about it. After a couple iterations of what our database could look like we settled on four tables – Event Location, Event, Zone and Exhibitor. Every event has an Event Location, and every Exhibitor has a Zone. Below is the entity relationship diagram which describes our database. The one-to-many relationships can be seen between the Event Location table and the Event table.

Once the database was finalised, we next decided on what software we would use to create and run the database. The choices were narrowed down to two options: Oracle and MySQL. Oracle was a good choice as every team member had gained experience with Oracle over the course of this year as we used it in our Databases module. Ultimately however, we decided to use MySQL due to Laravel's "out of the box" support for it, along with two members of the backend having prior experience with MySQL. Having completed the Database design this mapped out what the API and the backend for the and the Admin Panel would look like. Event, Event Location, Zone and Exhibitor would each get two controllers – one for API requests, and one for Admin Panel requests. Whilst it would have been possible to process both types of requests through the same controller, splitting them up was done to improve the readability of the code.

4.5 Admin Panel Design and Development

As mentioned in the Admin Panel Frontend Research section, we knew that we would try to replicate the Nottinghamshire Showground's current website as close as possible. For this reason, most of the design phase was targeted towards what we wanted the user to be able to do from the web panel. The table below breaks down the actions a user can perform on each page.

PAGE NAME	ACTIONS
EVENT LOCATIONS	
VIEW EVENT LOCATIONS	• View Events in each Event Location • Go to each Event Location's page • Go to Event page • Go to the Create Event Location page
VIEW EVENT LOCATION	• Update all fields of the Event Location • Filter Events happening in the Event Location by keywords • Delete Event Location
CREATE EVENT LOCATION	• Create new Event Location
EVENTS	
VIEW EVENT	• Update all fields of the Event • Delete Event • Go to the create Event page
CREATE EVENT	• Create a new Event
ZONES	
VIEW ZONES	• View each Zone and how many exhibitors are in that zone • Go to each Zone's page • Search for Zones by keywords • Go to the create Zones page
VIEW ZONE	• Update all fields of the Zone • Search for Exhibitors in Zone by keywords • Delete the Zone
CREATE ZONE	• Create a new Zone
EXHIBITORS	
VIEW EXHIBITORS	• Search for Exhibitor by keywords • Go to the Exhibitors Zone's page • Go to the Exhibitors Page • Go to the Create Exhibitor Page
VIEW EXHIBITOR	• Update all fields of the Exhibitor • Delete the Exhibitor
CREATE EXHIBITOR	• Create a new Exhibitor
COMPETITIONS	
VIEW COMPETITIONS	• View all competitions • Search for competitions by name • Go to each competition
CREATE COMPETITION	• Create competition
VIEW COMPETITION	• Edit competition • Delete Competition
HOME PAGE	
HOME PAGE	• View all statistics on number of Events, Event Locations, Zones and Exhibitors

	• Go to the create pages of Events, Event Locations, Zones and Exhibitors • Go to each Zone's page • Go to each Events Location's page
LOGIN/SIGN UP PAGE	
LOGIN/SIGN UP PAGE	• Login with existing account • Create a new account

Table 1 – Features in the Admin Panel

Whilst we took a lot of the design from the Nottinghamshire Showgrounds current website, we still had to modify various elements, as well as make some new from scratch. Below are some comparisons to elements we have modified as well as explaining elements we made from scratch.

4.5.1 The navigation bar - Changed

The navigation bar on the Nottinghamshire Showground's website contains the drop-down menu within it that we needed to make the navigation bar used in our admin panel.

The original navigation bar can be seen here:

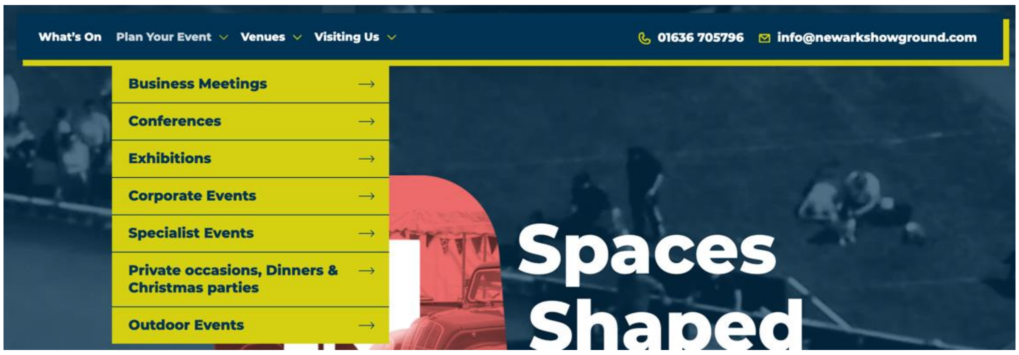


Figure 9: Navigation Bar

And the modified version can be seen here:

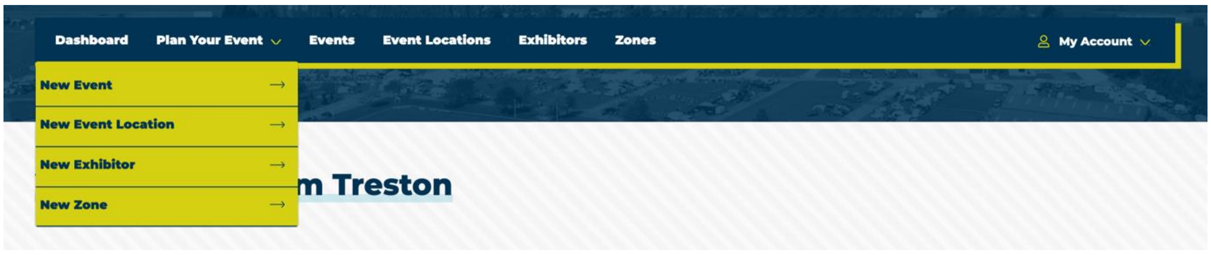
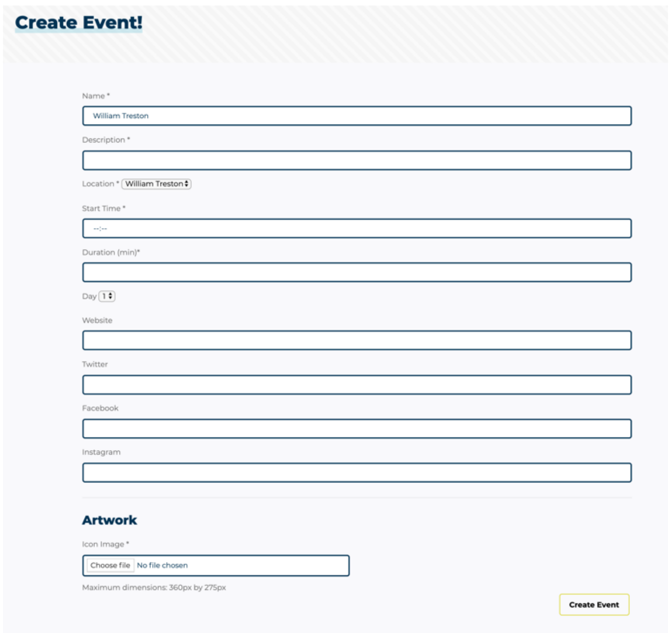


Figure 10: Modified Navigation Bar

As can be seen, the two are very similar.

4.5.2 Forms – New


Create Event!

Name *

William Treston

Description *

Location * William Treston

Start Time *

--:--

Duration (min)*

Day 1

Website

Twitter

Facebook

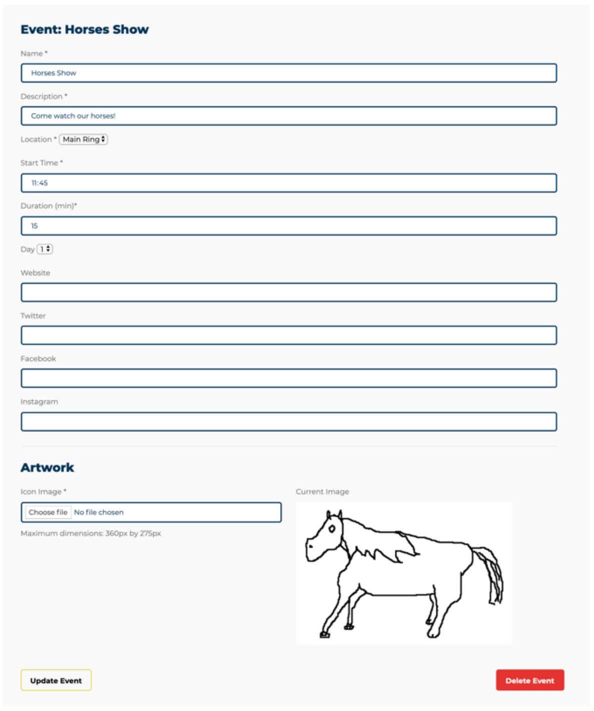
Instagram

Artwork

Icon Image *

Choose file No file chosen

Maximum dimensions: 360px by 275px


Event: Horses Show

Name *

Horses Show

Description *

Come watch our horses!

Location * Main Ring

Start Time *

11:45

Duration (min)*

15

Day 1

Website

Twitter

Facebook

Instagram

Artwork

Icon Image *

Choose file No file chosen

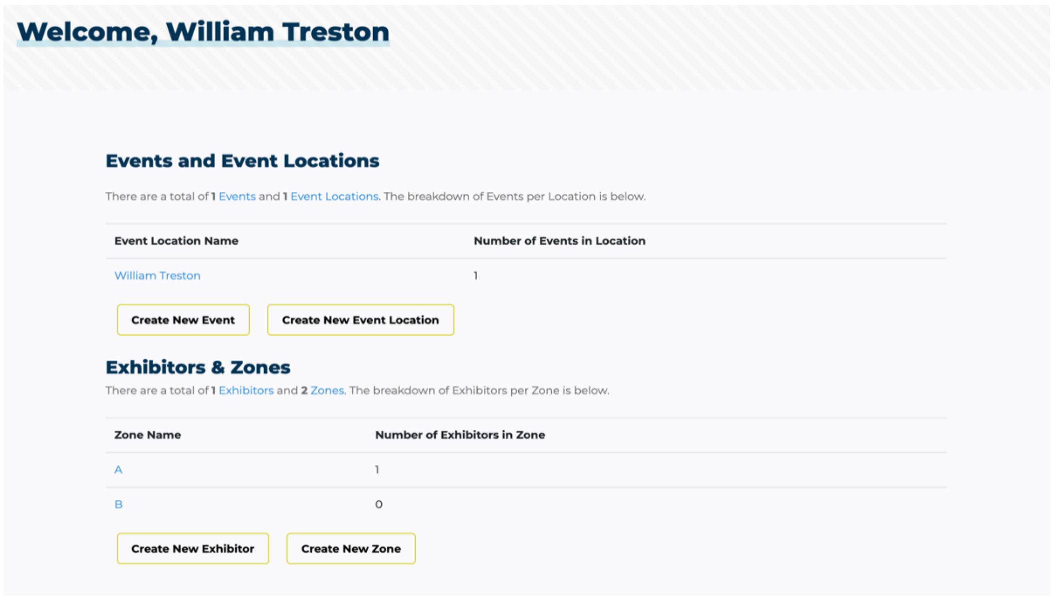
Maximum dimensions: 360px by 275px

Current Image

Figure 11: Forms

The forms used on each viewing or creation page had to be made from scratch, as the Nottinghamshire Showground's website had no forms on it. As we were aiming to have as close to a similar feel as possible to the original site, we kept the form fields very minimalistic and made them follow the colour schemes of the website. Below are two examples of the forms we made – one for creating a new event, and one for editing an event.

4.5.3 Home Page - New



Welcome, William Treston

Events and Event Locations

There are a total of **1** Events and **1** Event Locations. The breakdown of Events per Location is below.

Event Location Name	Number of Events in Location
William Treston	1

Exhibitors & Zones

There are a total of **1** Exhibitors and **2** Zones. The breakdown of Exhibitors per Zone is below.

Zone Name	Number of Exhibitors in Zone
A	1
B	0

Figure 12: Home Page

The homepage required no new elements to be made, however its layout was designed from scratch. We wanted this page to show some general statistics as well as show the Event Locations and Zones to make these pages easily accessible. Below is the final product.

4.5.4 Development

Each section of the Admin Panel contains similar functionalities and therefore they have been implemented in very similar ways. For this reason, in this report we have only explained the implementation of one view, create, delete, update and search actions.

Create New Event

This describes, in pseudocode how a new Event is created once the form has been submitted.

```
IF user is logged in:
        check that form data is valid

        create new event object

        event->name = request->name
        event->description = request->description
        event->location_id = request->location_id

        IF request->website NOT null
                event->website = request->website
        ENDIF

        IF request->Facebook NOT null
                event->Facebook = request->Facebook
        ENDIF
        IF request->instagram NOT null
                event->instagram = request->instagram
        ENDIF

        event->start_time = request->time
        event->day = request->day
        event->duration = request->duration

        IF request->image NOT null
                save image
                event->image = path to image
        ENDIF

        save event to database

        redirect to new event page
ELSE
        redirect to login page
ENDIF
```

View Event

This describes, in pseudocode how an Event is retrieved from the database and then displayed.

```
IF user is logged in:
    get event by ID from database
    return template with event info
ELSE
    redirect to login page
ENDIF
```

Edit Event

This describes, in pseudocode how an Event is edited.

```
IF user is logged in:
    get event by ID from database

    IF request->name different to event->name
        event->name = request->name
    ENDIF
    IF request->description different to event->description
        event->description = request->description
    ENDIF
    IF request->location_id different to event->location_id
        event->location_id = request->location_id
    ENDIF
    IF request->website different to event->website
        event->website = request->website
    ENDIF
    IF request->Facebook different to event->Facebook
        event->Facebook = request->Facebook
    ENDIF
    IF request->instagram different to event->instagram
        event->instagram = request->instagram
    ENDIF
    IF request->start_time different to event->start_time
        event->start_time = request->start_time
    ENDIF
    IF request->day different to event->day
        event->day = request->day
    ENDIF
    IF request->duration different to event->duration
        event->duration = request->duration
    ENDIF
    IF request->image different to event->image
        save image
        event->image = path to image
    ENDIF

    save event to database

    redirect to event's page with updated info
ELSE
    redirect to login page
ENDIF
```

Delete Event

This describes, in pseudocode how an Event is deleted.

```
IF user is logged in:
    get event by ID
    delete event
    redirect to list of events page

ELSE
    redirect to login page
ENDIF
```

The four pieces of pseudo code above are identical to the implementations for the four other objects in the database apart from attribute names. The dashboard implementation is the most complex, and for this reason, we have decided to also include its pseudocode below.

Dashboard

This describes, in pseudocode how the statistics on the dashboard are counted.

```
IF user is logged in:
    get all events
    count how many events there are
    get all event locations
    FOREACH event location
        count how many events are occurring in this location
    ENDFOREACH
    count how many event locations there are

    get all zones
    count how many zones there are

    FOREACH zone
        count how many exhibitors in this zone
    ENDFOREACH

    count total number of exhibitors
    get all competitions
    count number of competitions
    load information into template

    return dashboard with template
ELSE
    redirect to login page
ENDIF
```

4.6 API Design and Development

The API was designed in a way to allow the app to get all the information possible about a specific Zone, Exhibitor, Event Location or Event in a single request. This choice was made for one main reason. During the two-day event at the Nottinghamshire Showground there will be thousands of devices connected to the internet via cellular data which may lead to issues with connectivity and internet speeds. For this reason, we decided that getting all information in one go was a better idea than making the app make multiple requests.

Once we had decided that the API would return all possible information in one request, we created the API endpoints. A list of all endpoints is below, however the full documentation for the API can be seen in Appendix 13.

- Get information on all Event Locations
- Get information on a specific Event Location
- Get information on all Events in a specific Event Location
- Get information on all Events
- Get information on a single Event
- Get information on all Exhibitors
- Get information on a specific Exhibitor
- Get information on Exhibitors within a specific Zone
- Get information on all Zones
- Get information on a single Zone
- Get information on all Competitions
- Get information on a single Competition

4.6.1 Development

Like the Admin Panel, the various endpoints of the API provide similar information to one another. For this reason, the way they have been implemented in Laravel are very similar, and once again we will only describe each type of endpoint once in pseudocode.

Get All Events

This describes, in pseudocode how the all Events are retrieved from the database when called by the API.

```
get all events
FOREACH
        get event location for event
ENDFOREACH

return events->as JSON
```

Get Single Event

This describes, in pseudocode how an Event is retrieved from the database when called by the API.

```
get event by ID
get event location for event
return event->as JSON
```

Get all Events in an Event Location

This describes, in pseudocode how all Events in a given Event Location are retrieved when called by the API.

```
get event location by id
IF event location doesn't exist
    return empty json object
ENDIF
get all events that have an event location id = event location id retrieved above
FOREACH event
    add event location details to event
ENDFOREACH
return events->as JSON
```

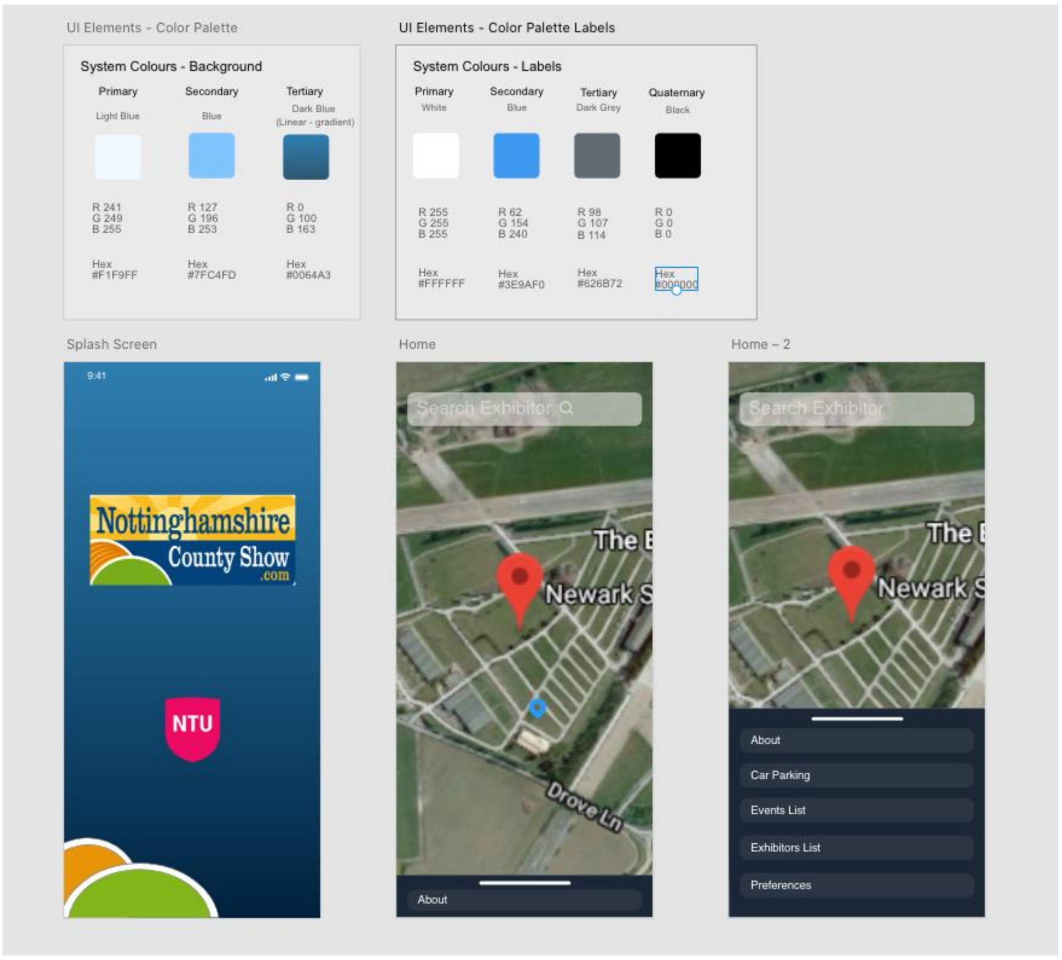
5 App Design & Development

5.1 App Design

Initial Design

Prior to the development of the mobile application the team created an initial design of the what the mobile application should look like. The purpose of the initial design was to represent the initial ideas, which the group had and start the discussion of the colour palette, pages which the application should include and approve the initial designs meet the client's expectations.

The initial design was created using Adobe XD; and included the following eight pages: Splash Screen, Home Page, About Page, Exhibitors List Page, Exhibitor View Page, Events List Page, Events View Page and the Car Parking Page. Figure shows the initial Splash screen and home page layout along with our initial colour palette. The full initial design is included appendix 1.



UI Elements - Color Palette

System Colours - Background

Primary	Secondary	Tertiary
Light Blue	Blue	Dark Blue (Linear - gradient)
R 241 G 249 B 255	R 127 G 196 B 253	R 0 G 100 B 163
Hex #F1F9FF	Hex #7FC4FD	Hex #0064A3

UI Elements - Color Palette Labels

System Colours - Labels

Primary	Secondary	Tertiary	Quaternary
White	Blue	Dark Grey	Black
R 255 G 255 B 255	R 62 G 154 B 240	R 98 G 107 B 114	R 0 G 0 B 0
Hex #FFFFFF	Hex #3E9AF0	Hex #626B72	Hex #000000

Figure 12: Initial Design

Our initial thoughts of going with a blue design was to imitate the Nottingham County Show's website they had, so we could possess some similarity between the mobile application we were creating and the website. This would also show some familiarity to the user as the website has been the main contact area users going to the show.

During our development the client modified their website and after a meeting with our client we were able to get a map of the area created by them which can be incorporated into our app. Along with this we created a new colour palette and redesigned the initial design to math their updated website. From this meeting we were given an extensive number of artwork and images which we could use to design the mobile application.

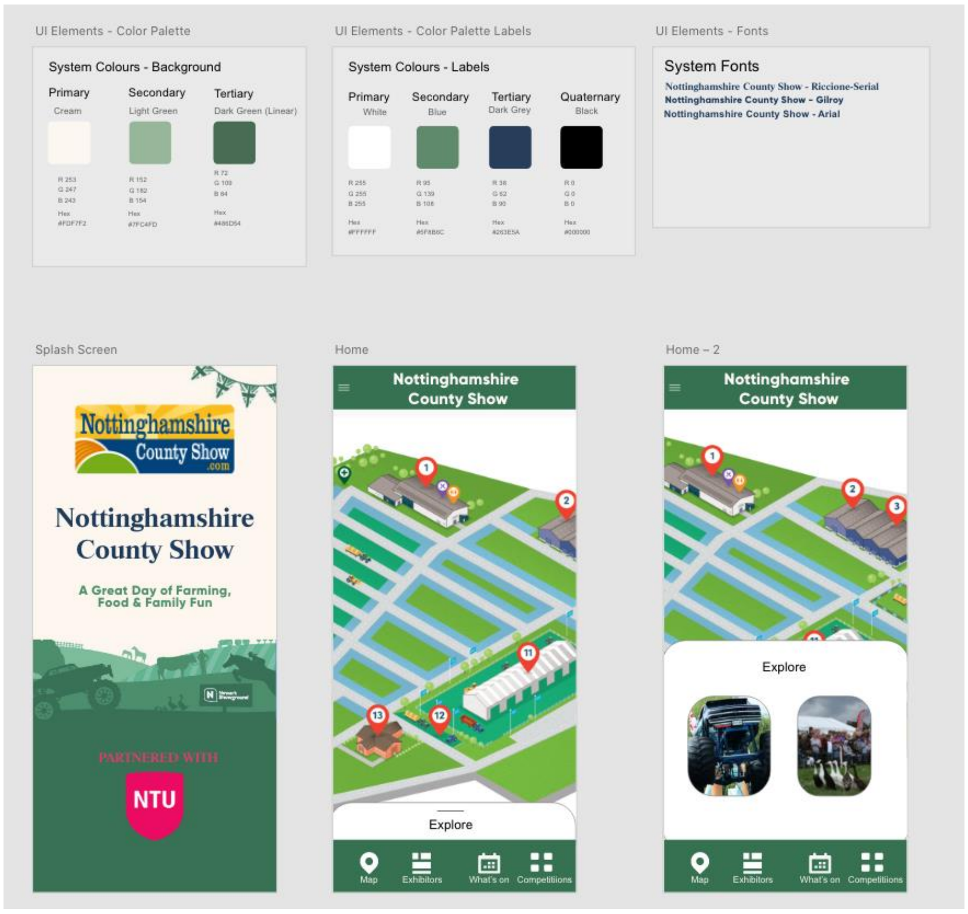


Figure 13: Modified Design

As shown in Figure 13, the Splash Screen is first page which the user sees when opening the application. We have included some of the design elements the Nottinghamshire County Show has given us such as the design in the background. The main screen of the application is the home page; this is where the map of the show ground is displayed. A navigation bar at the bottom of the screen is also present to navigate between different pages such as the exhibitor's pages and the competitions page. The remainder of the screens can be found in Appendix 25.

5.2 App Development

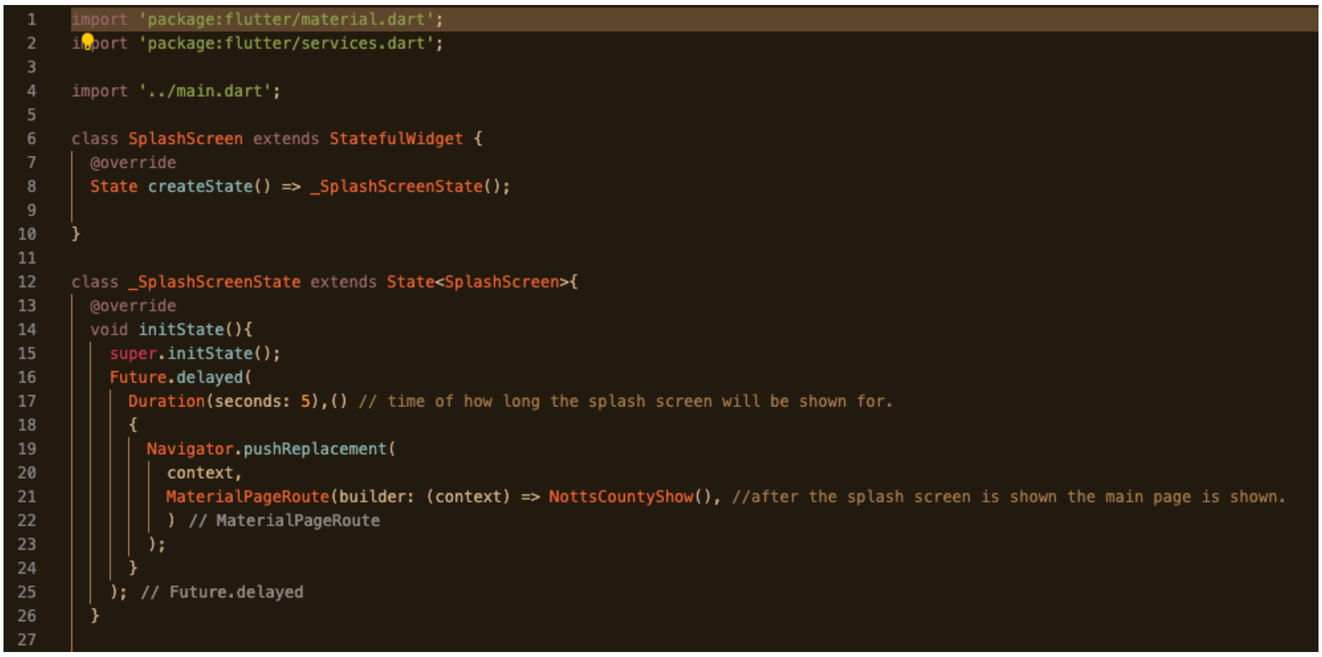
This section will describe the developmental stages of the project providing the structure and the thinking behind how the team were able to develop significant stages of the project. This section will also discuss some of the difficulties the group faced in the development stage and how they were resolved along with some of the key functionality that the team would like to highlight. The functionalities we would like to highlight in this section contain the Splash Screen, Main Screen, the map.

The approach taken in the development stages was to identify the screens that are required for the project which were done in the design stages and try to reflect the designs from section 5.1 in the app. Members on the backend development team were given specific pages to develop which then were integrated later together.

Due to the lack of experience in flutter by most members of the team it was very important the team understood key terminology in the code. Some of the key terminology required to start the project was understanding the concept of a 'widget' and more specifically understanding the difference between a 'stateful widget' and a 'stateless widget'. A Flutter widget is a tool which can be used to build a user interface; depending on the configurations and state widgets display what their view should look like. Some examples of widgets include Text, Row, Column; a text widget creates a run of a styled text within the application. The flutter documentation page describes a stateful widget as 'dynamic' compared to a stateless widget which is referred one that 'never changes' (Flutter.dev, 2020) or static. Team members were given specific tasks regarding the flutter development; one of the first tasks was to create a splash screen for the mobile application.

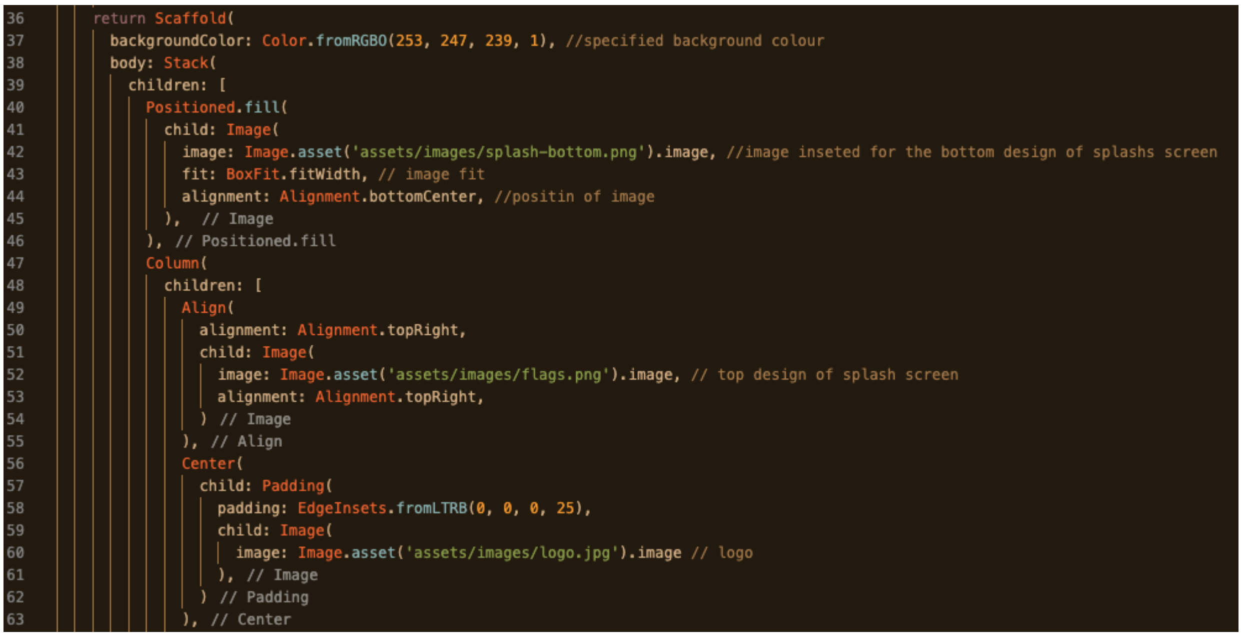
5.2.1 Code Design of the Splash Screen

Figure 14: Screenshot of Splash Screen code



```dart
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';

import '../main.dart';

class SplashScreen extends StatefulWidget {
  @override
  State createState() => _SplashScreenState();

}

class _SplashScreenState extends State<SplashScreen>{
  @override
  void initState(){
    super.initState();
    Future.delayed(
      Duration(seconds: 5),() // time of how long the splash screen will be shown for.
      {
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (context) => NottsCountyShow(), //after the splash screen is shown the main page is shown.
          ) // MaterialPageRoute
        );
      }
    ); // Future.delayed
  }

```

Splash Screen is a screen which opens on the launch of a program in this case when the mobile application is launched the first screen the user will see is the splash screen. In section *5.1 App Design* the design of the Splash Screen is shown. In this snippet of code, we have created a class and extended to a stateful widget; in the class we have created a mutable state for this widget at a given location using createState(), the framework can call the this method multiple times. The code below shows the duration of the splash screen along with the next screen which will be shown.



```
36        return Scaffold(
37          backgroundColor: Color.fromRGBO(253, 247, 239, 1), //specified background colour
38          body: Stack(
39            children: [
40              Positioned.fill(
41                child: Image(
42                  image: Image.asset('assets/images/splash-bottom.png').image, //image inseted for the bottom design of splashs screen
43                  fit: BoxFit.fitWidth, // image fit
44                  alignment: Alignment.bottomCenter, //positin of image
45                ),  // Image
46              ), // Positioned.fill
47              Column(
48                children: [
49                  Align(
50                    alignment: Alignment.topRight,
51                    child: Image(
52                      image: Image.asset('assets/images/flags.png').image, // top design of splash screen
53                      alignment: Alignment.topRight,
54                    ) // Image
55                  ), // Align
56                  Center(
57                    child: Padding(
58                      padding: EdgeInsets.fromLTRB(0, 0, 0, 25),
59                      child: Image(
60                        image: Image.asset('assets/images/logo.jpg').image // logo
61                      ), // Image
62                    ) // Padding
63                  ), // Center
```

Figure 15: Positioning of widgets

The code above is a small snippet of the structure the team has used to create the Splash screen from declaring the duration of the splash screen (5 seconds) to creating widgets to hold the images for the splash screen. The statement on line 16 Future.delayed() allows the splash screen to be shown for a specific amount of time which is seen on the next line. The next computation on line 21 in this case will run after 5 seconds; this changes the screen from the Splash Screen to the main screen of the application. The importance of structuring is very crucial in Flutter as there can be many errors shown if the correct statements are not included in the correct positions.

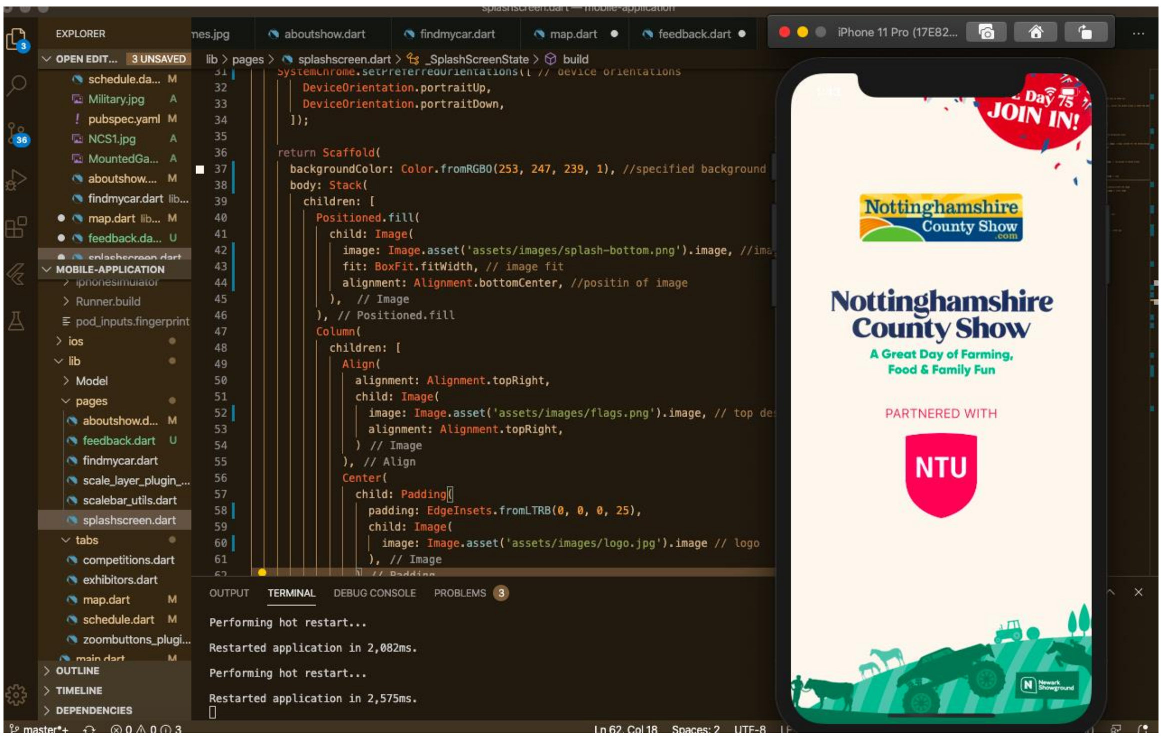
Some of the difficulties the team had with the implementation of this code was the correct positioning of the images on the screen. This was solved by implementing the correct structure of the code and understanding which widgets should follow one another. In Figure 16 to position the design we have used the widget Scaffold as shown on line 36. When the mobile application will be opened a scaffold is returned (a scaffold returns the basic structure of the visual design) which contains a Stack. The stack allows us to overlay widgets such as Positioned, Column and Center. This is the structure of the widgets we have used to create the splash screen and position all the design elements accurately. The Figure below displays the final look of the Splash Screen implemented on an iPhone 11 Pro.

To design this screen, the frontend implementation team has used the colour palette which is included in Figure 14. The design elements such as the logo and the other decorations were stored as image files (jpg or png) and a new folder is created in the asset folder called ‘image’. This folder only holds all the images which are to be used by the application. To display the decorations on the splash screen we have used the Image.asset() constructor which holds the file path of the image allows us to include these images to the Splash Screen. By being able to include the design images and positioning the widgets accurately we were able to produce the Splash Screen shown below in Figure 17.

5.2.2 Code Design of the Main screen.

The main feature of this mobile application is the unique map given to the team by the client. From the designs shown in *section 5.1* the map is located on the main page of the application which is the first thing the user will see after the Splash Screen. The team felt this was an important aspect of the application because the map is the main and key functionality of the application by putting the map on the main page it makes the map easily accessible for all users.

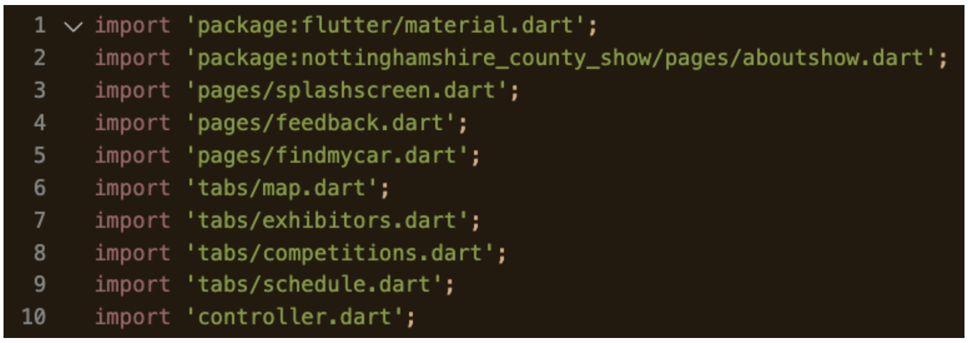
In this section we will be explaining and discussing the implementation of the map on the main screen. Below is a snippet of the code we developed which shows the packages we used in order to acquire the appropriate functionality of the code.



```
    SystemChrome.setPreferredOrientations([ // device orientations
      DeviceOrientation.portraitUp,
      DeviceOrientation.portraitDown,
    ]);

    return Scaffold(
      backgroundColor: Color.fromRGBO(253, 247, 239, 1), //specified background
      body: Stack(
        children: [
          Positioned.fill(
            child: Image(
              image: Image.asset('assets/images/splash-bottom.png').image, //ima
              fit: BoxFit.fitWidth, // image fit
              alignment: Alignment.bottomCenter, //positin of image
            ),  // Image
          ), // Positioned.fill
          Column(
            children: [
              Align(
                alignment: Alignment.topRight,
                child: Image(
                  image: Image.asset('assets/images/flags.png').image, // top de
                  alignment: Alignment.topRight,
                ) // Image
              ), // Align
              Center(
                child: Padding(
                  padding: EdgeInsets.fromLTRB(0, 0, 0, 25),
                  child: Image(
                    image: Image.asset('assets/images/logo.jpg').image // logo
                  ), // Image
```

```
Performing hot restart...
Restarted application in 2,082ms.
Performing hot restart...
Restarted application in 2,575ms.
```

Figure 16 - Result of Splash Screen after implementation

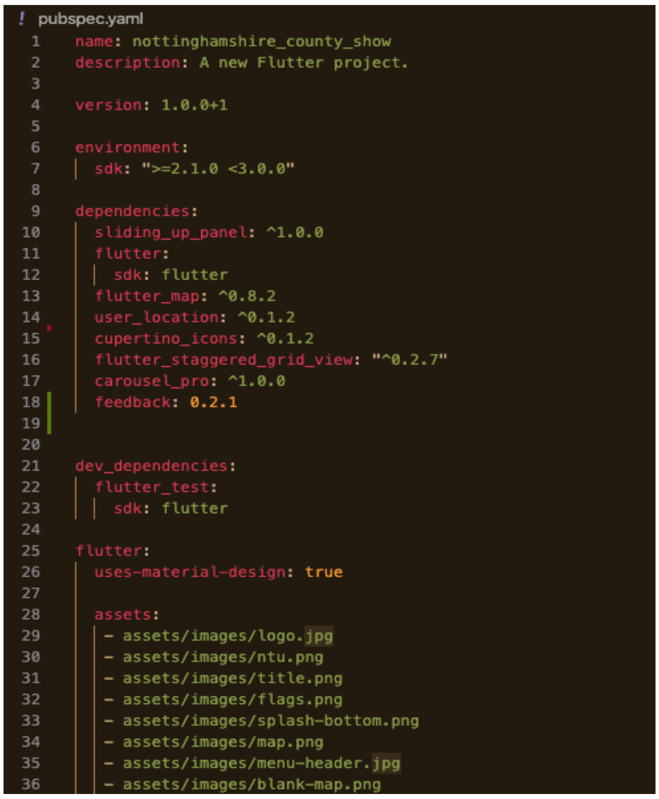


```dart
import 'package:flutter/material.dart';
import 'package:nottinghamshire_county_show/pages/aboutshow.dart';
import 'pages/splashscreen.dart';
import 'pages/feedback.dart';
import 'pages/findmycar.dart';
import 'tabs/map.dart';
import 'tabs/exhibitors.dart';
import 'tabs/competitions.dart';
import 'tabs/schedule.dart';
import 'controller.dart';
```

Figure 17: Import third party packages

To implement the map, we included a package called flutter_map which we have declared in the pubspec.yaml under the dependencies section as shown below. The pubspec.yaml is a key file that we used in the development of the project to import packages and include images in the final product.

The pubspec.yaml shows the name, description, version and environment of the project. To import any packages into the package the dependencies must be written in this file followed by the command ‘flutter pub get’ in the terminal to install any new packages.



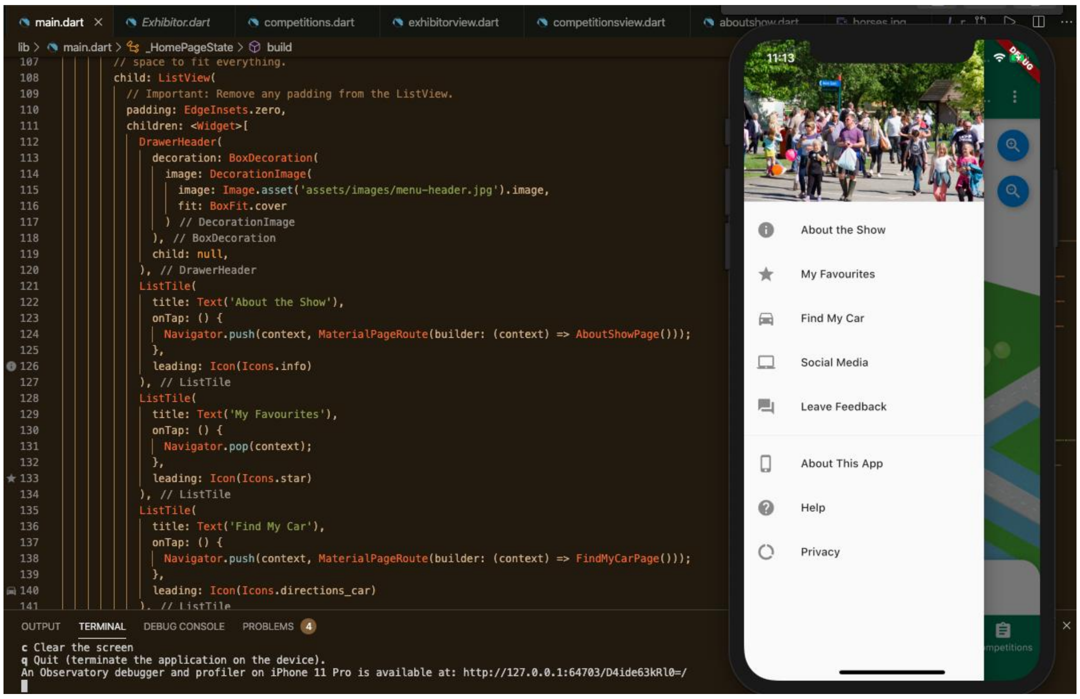
```yaml
! pubspec.yaml
name: nottinghamshire_county_show
description: A new Flutter project.

version: 1.0.0+1

environment:
  sdk: ">=2.1.0 <3.0.0"

dependencies:
  sliding_up_panel: ^1.0.0
  flutter:
    sdk: flutter
  flutter_map: ^0.8.2
  user_location: ^0.1.2
  cupertino_icons: ^0.1.2
  flutter_staggered_grid_view: "^0.2.7"
  carousel_pro: ^1.0.0
  feedback: 0.2.1


dev_dependencies:
  flutter_test:
    sdk: flutter

flutter:
  uses-material-design: true

  assets:
   - assets/images/logo.jpg
   - assets/images/ntu.png
   - assets/images/title.png
   - assets/images/flags.png
   - assets/images/splash-bottom.png
   - assets/images/map.png
   - assets/images/menu-header.jpg
   - assets/images/blank-map.png
```

Figure 18: Pubspec.yaml

As noted in Section 5.2.1 to include images in the application their file paths must be noted in the pubspec.yaml file for the images to be used. Another problem the team faced was when importing these dependencies, the team had a problem with flutter recognising the dependencies in the pubspec.yaml file this was due to some dependencies not being the latest versions.

The main screen contains an app bar with the title of the app. In the main file of this project the main screens widgets and visual is design is written. To retrieve the data of the information the Showground have given us we have used a DataContoroller; the data is retrieved when the splash screen is loading. The data that is received is mainly the exhibitor's details such as their name, description and image if provided which can be seen on the exhibitors view page.

Inside the body of the scaffold we have implemented a Drawer and ListView which are two widgets in flutter that have enabled the team to implement the side panel which shows the different pages in the app as seen in the figure below. To be able to use the Drawer widget it was required that a Scaffold was returned before. To the Drawer the team has had added a ListView which ensures the user can scroll down if there is limited vertical space to fit all the tiles. The ListView contains children widgets which in this case are ListTiles holding the title and navigation contents which direct the user to page they would like to access.



```
            // space to fit everything.
            child: ListView(
              // Important: Remove any padding from the ListView.
              padding: EdgeInsets.zero,
              children: <Widget>[
                DrawerHeader(
                  decoration: BoxDecoration(
                    image: DecorationImage(
                      image: Image.asset('assets/images/menu-header.jpg').image,
                      fit: BoxFit.cover
                    ) // DecorationImage
                  ), // BoxDecoration
                  child: null,
                ), // DrawerHeader
                ListTile(
                  title: Text('About the Show'),
                  onTap: () {
                    Navigator.push(context, MaterialPageRoute(builder: (context) => AboutShowPage()));
                  },
                  leading: Icon(Icons.info)
                ), // ListTile
                ListTile(
                  title: Text('My Favourites'),
                  onTap: () {
                    Navigator.pop(context);
                  },
                  leading: Icon(Icons.star)
                ), // ListTile
                ListTile(
                  title: Text('Find My Car'),
                  onTap: () {
                    Navigator.push(context, MaterialPageRoute(builder: (context) => FindMyCarPage()));
                  },
                  leading: Icon(Icons.directions_car)
                ), // ListTile
```

```
c Clear the screen
q Quit (terminate the application on the device).
An Observatory debugger and profiler on iPhone 11 Pro is available at: http://127.0.0.1:64703/D4ide63kRl0=/
```

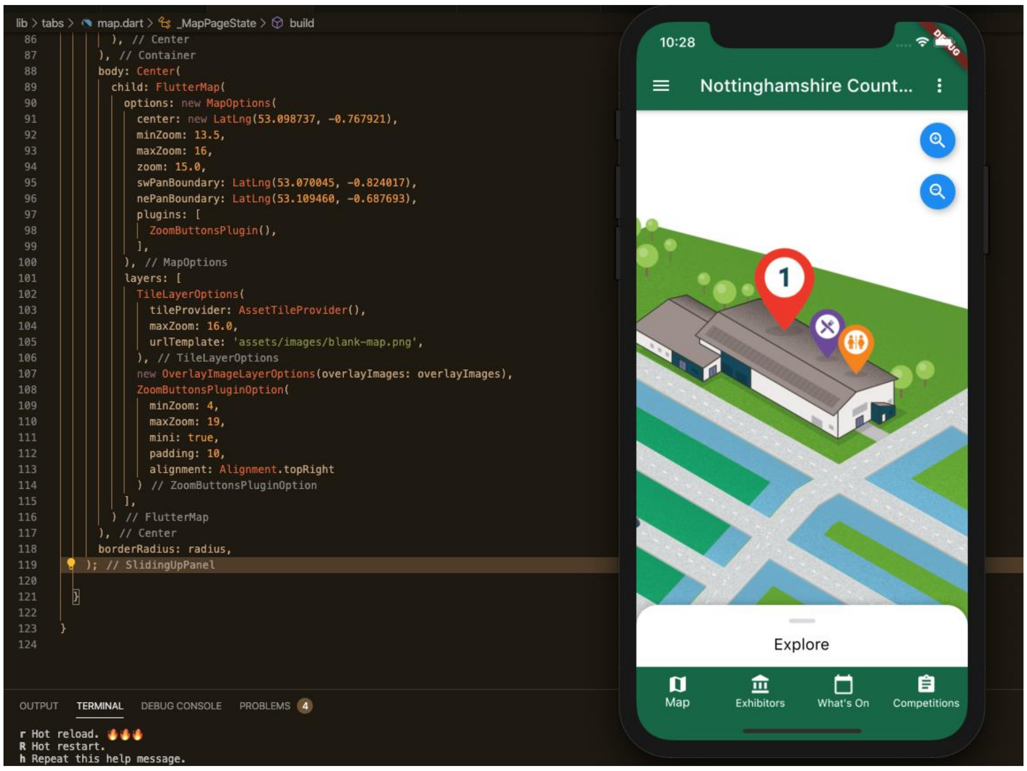
Figure 19: Development of the sidebar

5.2.3 The Map

The development of the map was one of the main developments the team worked on. It was important for the team to stay in communication with the Nottinghamshire County Show team and understand their needs. In our case the showground had a map that they wanted the team to implement.

The map has been implemented using flutter_map dependency which allows us to create an offline map. As shown in the figure below we have set the maps size and restraints allowing the user to move the map around. We have also allowed the user to zoom into the map using a zoom button plugin which is available with the flutter_map dependency. The zoom buttons are also given restraints to a certain amount allowing the use to zoom in and out to an extent.

The zoom buttons have a separate file for their implementation and in this file the minimimum and maximum zoom attributes defined along with the design of the buttons the bounds of the map are also defined in the zoombuttons_plugins file allowing the user to move the app when zoomed in or out. On this same screen we have also implemented a sliding panel which shows the events that are going on at the Nottinghamshire County Show; this aspect still requires more development which is highlighted later on in the document.


lib > tabs > map.dart > _MapPageState > build

```
          ), // Center
        ), // Container
        body: Center(
          child: FlutterMap(
            options: new MapOptions(
              center: new LatLng(53.098737, -0.767921),
              minZoom: 13.5,
              maxZoom: 16,
              zoom: 15.0,
              swPanBoundary: LatLng(53.070045, -0.824017),
              nePanBoundary: LatLng(53.109460, -0.687693),
              plugins: [
                ZoomButtonsPlugin(),
              ],
            ), // MapOptions
            layers: [
              TileLayerOptions(
                tileProvider: AssetTileProvider(),
                maxZoom: 16.0,
                urlTemplate: 'assets/images/blank-map.png',
              ), // TileLayerOptions
              new OverlayImageLayerOptions(overlayImages: overlayImages),
              ZoomButtonsPluginOption(
                minZoom: 4,
                maxZoom: 19,
                mini: true,
                padding: 10,
                alignment: Alignment.topRight
              ) // ZoomButtonsPluginOption
            ],
          ) // FlutterMap
        ), // Center
        borderRadius: radius,
      ); // SlidingUpPanel

    }

  }
```

```
r Hot reload. 🔥🔥🔥
R Hot restart.
h Repeat this help message.
```

Figure 20: Development of the map

There are more features which we were able to highlight such as the exhibitors and events page which both have viewing pages for each exhibitor/event. A similar coding style is used for these pages as well. We have also included a find my car feature which was implemented through the use of flutter_map and user_location dependencies similar to map feature shown above. Another feature we have implemented was the fetching of data from the backend for this we created a new controller file and were able to retrieve data and use it in the app. This section highlighted some of the important features that we have implemented in the development section the other features will be highlighted in the demonstration.

6 Testing

6.1 Testing Plan

The following is an explanation of how evaluation of the application would have been carried out; due to the COVID-19 pandemic and the resulting restrictions on movement put in place in the United Kingdom, the group was unable to visit the showground to perform these tests.

Internet connectivity is arguably the most important aspect of the application – there would need to be a strong enough signal for users to collect information from the backend while present at the showgrounds. The cached information also needs to be tested to ensure the product retains as much functionality as possible if the signal is lost.

To test the GPS functionality of the app, the team would need to visit the Newark showground in Coddington, Nottinghamshire. From here, a set of predetermined locations would be visited in person, and this would be compared to the location displayed in the app. For the test to succeed, the position displayed in the app must match the user's physical location. Similar testing would also be carried out for the 'Find My Car' feature.

As the app is intended for use by the general public, accessibility is of utmost important – the app should be easy to use and navigate and should be aesthetically appealing. For Android devices, the Accessibility Scanner is a tool that can help with this process. The app must also meet the requirements and guidelines set in place by Apple and Google for publication on the App Store and Google Play, respectively.

6.1.1 Admin Panel Functional Testing

The functionality testing of the was done by going through each one of the features listed in table 1. Whilst the majority of these passed first time, there was a few issues with the "Search for Exhibitors in Zone by keywords" on the view Zone page. The reason for this error was a simple logical error in an if statement that was "==" rather than "!=". The full list of Functionality Tests for the Admin Panel can be seen in Appendix 22.

6.1.2 API Functionality testing

The API was also tested using functionality testing. The tests we performed were directly related to the endpoints that the API has. Each endpoint had two tests performed on it – one with parameters that would return a successful response, and one that would cause a 404 error. All tests passed first time and the full list of functionality tests for the API can be seen in Appendix 23.

6.2 Integration Testing

The integration testing phase consisted of ensuring that the API, admin panel and mobile app could work together. The tests consisted of calling each endpoint, parsing the data returned and then displaying it on the app. The table showing these tests is in Appendix 24.

7 Evaluation

The aim behind this application was to design and implement an application which will be suitable for the Nottinghamshire County Show and enhance the user's experience. It was vital that we followed the design principles to develop this app, and in my opinion, I think we did it. The application has a stable interface with little to no lag at all, the information on the application was easy to navigate. In addition to that, the map was intuitive and clear to the user so anyone can understand it with no ease. It is also important to note that the application is available on Apple and Android too.

The Brecon Agriculture County Show application was an app created by another developer. As analysed above (see Fig. 2.1) The application was quite underwhelming compared to their website, despite this, there are some parts in the application that they did better than us. For example, their application has multi language support, so you can switch languages for the application and their application can switch different metric systems through the settings page.

However, despite The Brecon Agriculture County Show application has a better language and metric system support, I believe our application has reign superior as our overall design and stability is better.

8 Conclusion

Going forward, the next possible opportunity for this project to be used for its purpose would be for the next Nottinghamshire County show in May 2021 or the Midlands Machinery Show in October 2020 (if this goes ahead). For this to happen, we would have to address the issues we have discovered during this project.

The first issue would be data and data entry specifically. For the mobile app to display any data (such as exhibitors and events) these need to be inputted somehow. An attempt was made to contact EventHalo – their 3rd party which handles data entry – in order to easily access the data via an API of some sort. However, they were not compliant and led us to other avenues. Our second attempt was to use spreadsheet data which was exported from EventHalo. However, this was lacking in detail and required manual work from a member of the showground. The final attempt was to have the showground team enter the data manually into our website so that we have full control over the data. This brought on an issue of time and technical ability by the staff. The solution to all these problems with data entry would be to have our system completely replace Event Halo for data entry so that entry of data only occurs once, and the app has full access to the data. This would require ensured security of personal information and reliability from the system.

Throughout the project, the team felt somewhat “in the dark” with information, branding and resources. Although the client is a charity, little was done to ensure that we had the correct branding material and information about the show during development. Furthermore, this branding was available as it was publicly released as a poster which is how we obtained the initial colour schemes and images. To ensure continuity between the website, the event and the app, the team should have been given direct access to the graphics team and provided with materials as soon as they became available to allow us to progress with development as soon as possible.

9 Professional, Social, Ethical and Legal Issues

When working for governing bodies, there are a variety of professional, social, ethical and legal issues that must be identified to maintain integrity throughout development. From the start it was clear that the app must stick to the guidelines set out by the British Computer Society, and IEEE-CE (Institute of Electrical and Electronics Engineers Computer Society), as all professionals working in the computer science industry should work to them. Therefore, a few examples of the guidelines the team will follow were found, to ensure that our final product is of the highest possible standard and to avoid Professional, Social, Ethical and Legal Issues.

9.1 Professional

The app will have university branding on it, any flaws, or imperfections with it will not only reflect poorly on the team, but also on NTU. So, it is important to follow this guideline as it states in the BCS Code of Conduct, "carry out your professional responsibilities with due care and diligence in accordance with the relevant authority's requirements while exercising your professional judgement at all times." Otherwise it may lead to less opportunities like the one the group is fortunate enough to have and take advantage of. To ensure that the app runs smoothly and is what the university and the agricultural society want, we will closely work with representatives from either party.

Our coding style will also follow good programming practises to ensure that the code is readable and reusable throughout the project and for the future too. This is important as in case of a change in development team in the future, a readable and understandable codebase will be of utmost importance. Lastly, to ensure approval on submitting to the Apple and Android app stores, strict adherence to their respective design guidelines will need to be followed. Without a successful application the app may not be able to be put out to the store by the targeted date of the Nottingham County Show. In this way it would be a failure of professional duty in the production of this app.

9.2 Social

One of the key areas of the BCS Code of Conduct pertains to public interest. Professionals "have due regard for public health, privacy, security and wellbeing of others and the environment" (British Computer Society, 2019). To make the app accessible to as many people as possible, when designing the user interface, consideration will be taken into the various forms of disabilities that people may have. As an example, to ensure that colour-blind people can use the app, the design will avoid colour combinations that are known to cause problems.

9.3 Ethical

Throughout the course of the development period it may be necessary to use various pieces of code and libraries found on the internet. To avoid any legal issues and to ensure that this project remains ethical, the group will firstly ensure that anything found can be used in our type of project, and if it is, will be correctly referenced. In addition, it is important professionals "have due regard for the legitimate right of third parties", indicating that users of the app need to know their information is being used ethically and solely for the purpose stated.

9.4 Legal

As the UK is currently a part of the EU, any data that is stored and/or processed needs to follow the General Data Protection Regulation (UK Government, 2018). This means that careful consideration will need to be taken with how information is stored, most importantly personal information such as names, addresses, phone numbers etc. To ensure the GDPR and other laws are followed, an arrangement of a meeting with NTU's legal team will be made in the hopes that they will be able to aid us in our goal of meeting and exceeding all set regulations. Finally, the smallest amount of information possible will be stored about the users and keep processing of user data to a minimum.

(British Computer Society, 2019)

References

Android, 2019. *Design for Android.* [Online]
Available at: https://developer.android.com/design
[Accessed 28 April 2020].

Anon., n.d. *IEEE Code of Ethics.* [Online]
Available at: https://www.ieee.org/about/corporate/governance/p7-8.html
[Accessed 2020].

Apple, 2020. *App Store Review Guidelines.* [Online]
Available at: https://developer.apple.com/app-store/review/guidelines/
[Accessed 30 April 2020].

Blair, I., 2019. *Mobile App Download and Usage Statistics (2019).* [Online]
Available at: https://buildfire.com/app-statistics/
[Accessed 15 April 2020].

British Computer Society, 2019. *British Computer Society.* [Online]
Available at: https://cdn.bcs.org/bcs-org-media/2211/bcs-code-of-conduct.pdf

Coding Dojo, n.d. *Choosing Python Web Frameworks: Django and Flask.* [Online]
Available at: https://www.codingdojo.com/blog/choosing-python-web-frameworks
[Accessed 2020].

Django, 2020. *Django Documentation.* [Online]
Available at: https://www.djangoproject.com/
[Accessed 2020].

Express Documentation, 2020. *Express Documentation.* [Online]
Available at: https://expressjs.com/
[Accessed 2020].

Flask, 2020. *Flask Documentation.* [Online]
Available at: https://flask.palletsprojects.com/
[Accessed 2020].

Google, 2020. *Flutter.* [Online]
Available at: https://flutter.dev/
[Accessed 09 January 2020].

Laravel, 2020. *Laravel.* [Online]
Available at: https://laravel.com/
[Accessed 01 May 2020].

Laravel, 2020. *Lumen.* [Online]
Available at: https://lumen.laravel.com/
[Accessed 09 January 2020].

Mindsea, 2019. *25 Mobile App Usage Statistics To Know In 2019.* [Online]
Available at: https://mindsea.com/app-stats/
[Accessed 15 April 2020].

MySQL, 2020. *MySQL.* [Online]
Available at: https://www.mysql.com/
[Accessed 09 January 2020].

Newak and Nottinghamshire Agricultural Society, 2019. *About Our Society.* [Online]
Available at: https://www.newarkshowground.com/our-charity/
[Accessed 12 January 2020].

Newark Showground, 2020. *Nottinghamshire County Show.* [Online]
Available at: https://www.nottinghamshirecountyshow.com/
[Accessed 4 December 2019].

NodeJS, 2020. *NodeJS.* [Online]
Available at: https://nodejs.org/en/
[Accessed 2020].

Nottinghamshire & Newark Agricultural Society, 2019. *Newark Showground.* [Online]
Available at: https://www.newarkshowground.com/
[Accessed 18 February 2020].

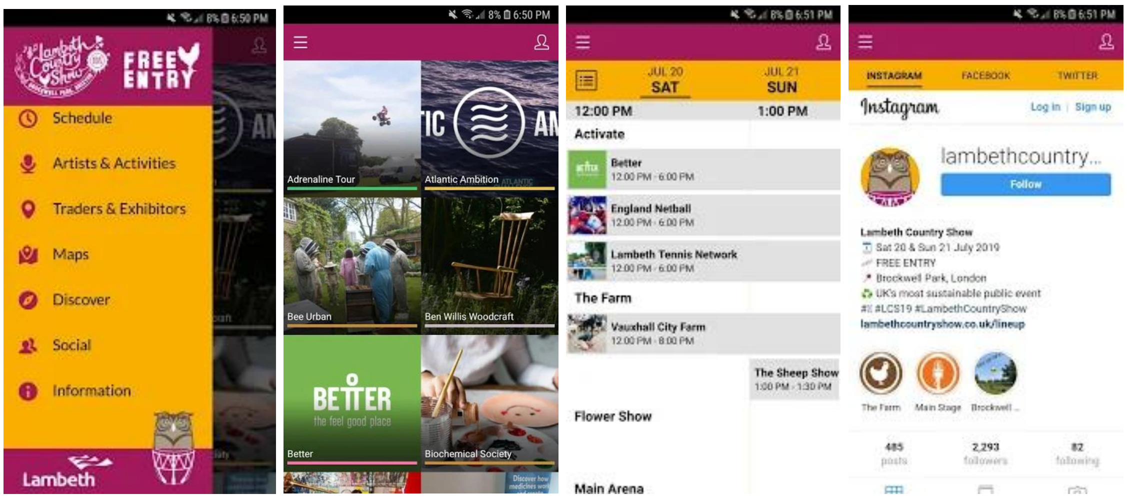
Pivotal, 2020. *Spring Boot.* [Online]
Available at: https://spring.io/projects/spring-boot
[Accessed 09 January 2020].

Skywell Software, 2019. *JAVA SPRING FRAMEWORK – PROS, CONS, COMMON MISTAKES.* [Online]
Available at: https://skywell.software/blog/java-spring-framework-pros-cons-mistakes/
[Accessed 2020].

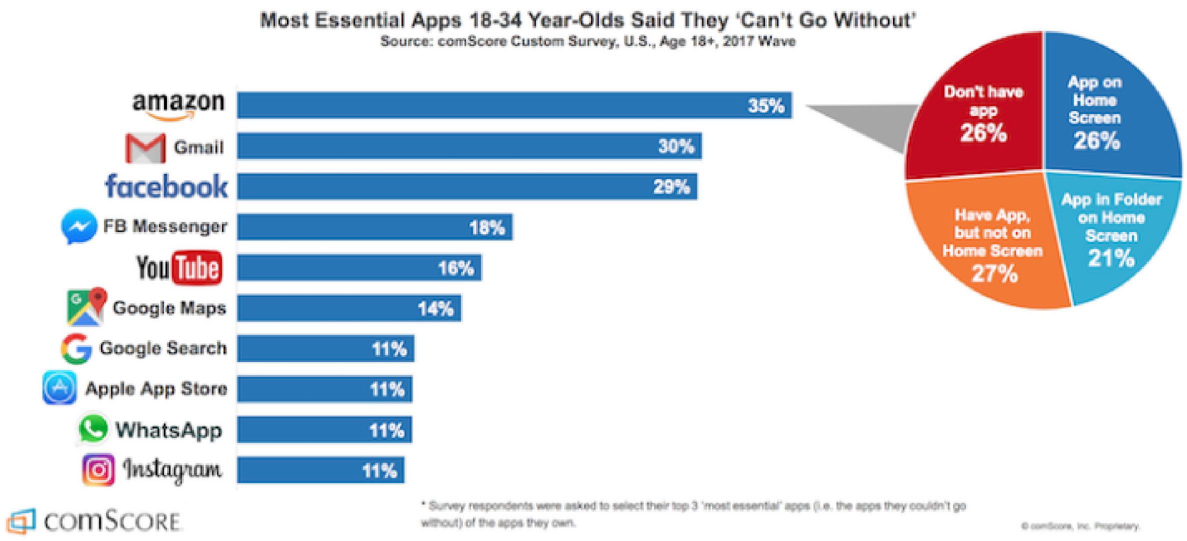
UK Government, 2018. *Data protection.* [Online]
Available at: https://www.gov.uk/data-protection
[Accessed 2020].

Appendix

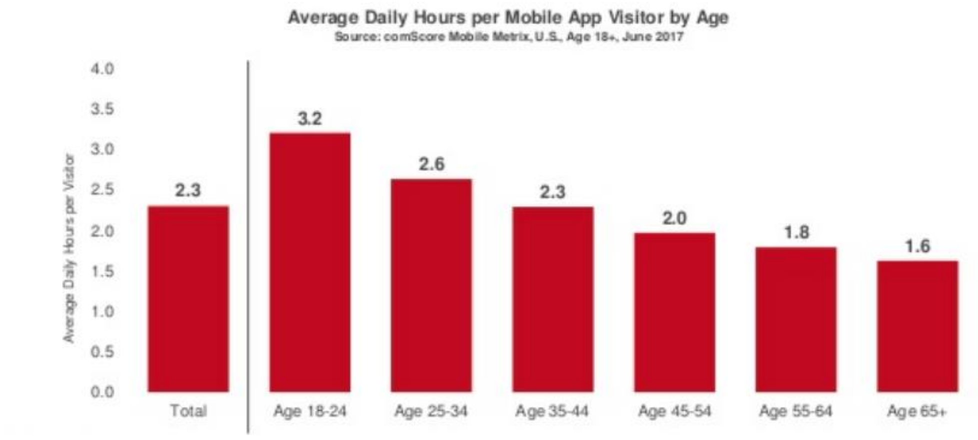
Appendix – 1



Appendix - 2

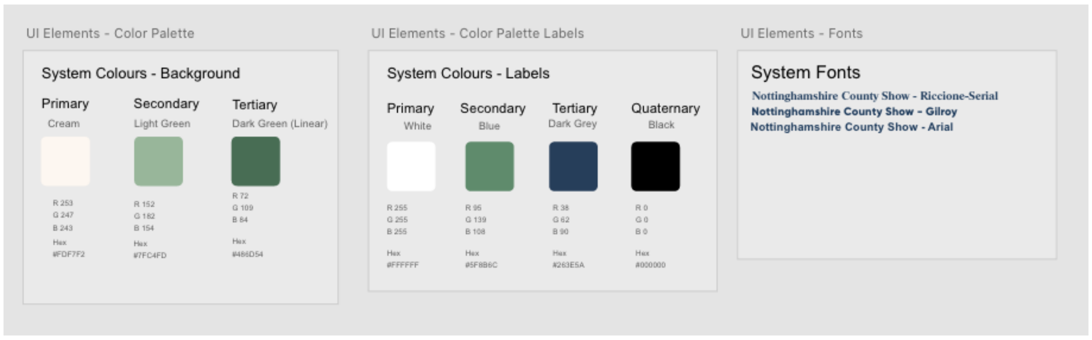


Appendix - 3



Average Daily Hours per Mobile App Visitor by Age
Source: comScore Mobile Metrix, U.S., Age 18+, June 2017
Average Daily Hours per Visitor
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
2.3
3.2
2.6
2.3
2.0
1.8
1.6
Total
Age 18-24
Age 25-34
Age 35-44
Age 45-54
Age 55-64
Age 65+

Appendix - 4



UI Elements - Color Palette

System Colours - Background

Primary	Secondary	Tertiary
Cream	Light Green	Dark Green (Linear)
R 253	R 152	R 72
G 247	G 182	G 109
B 243	B 154	B 84
Hex	Hex	Hex
#FDF7F2	#7FC4FD	#486D54

UI Elements - Color Palette Labels

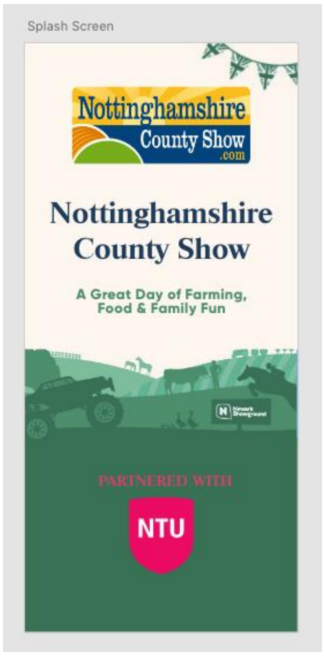
System Colours - Labels

Primary	Secondary	Tertiary	Quaternary
White	Blue	Dark Grey	Black
R 255	R 95	R 38	R 0
G 255	G 139	G 62	G 0
B 255	B 108	B 90	B 0
Hex	Hex	Hex	Hex
#FFFFFF	#5F8B6C	#263E5A	#000000

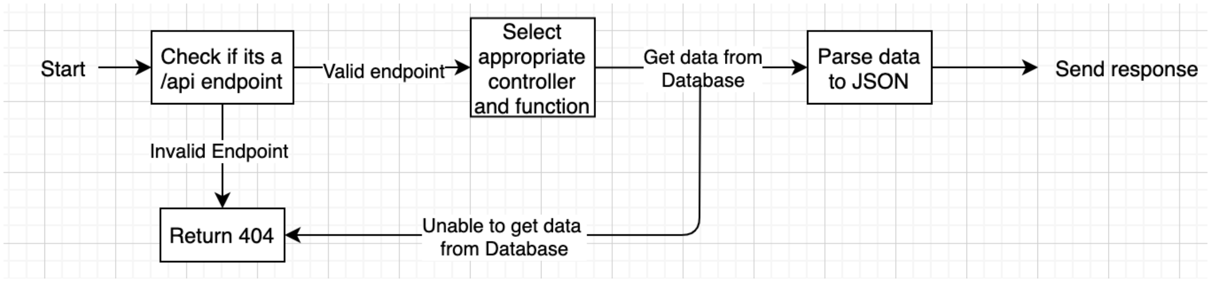
UI Elements - Fonts

System Fonts

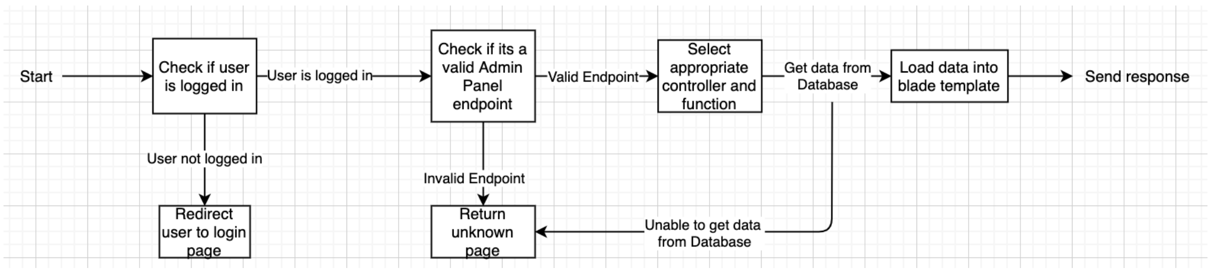
Nottinghamshire County Show - Riccione-Serial
Nottinghamshire County Show - Gilroy
Nottinghamshire County Show - Arial



Appendix - 5

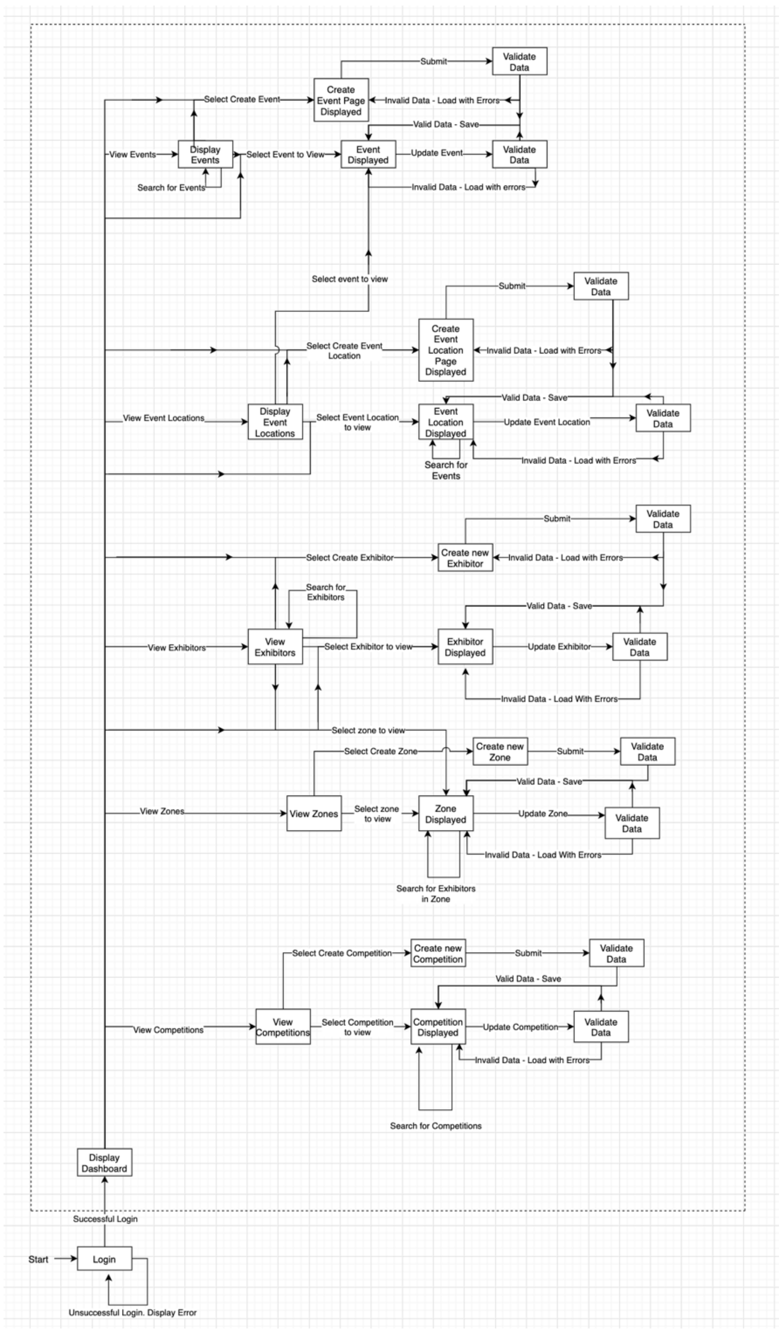

Start
Check if its a /api endpoint
Valid endpoint
Select appropriate controller and function
Get data from Database
Parse data to JSON
Send response
Invalid Endpoint
Return 404
Unable to get data from Database

Appendix - 6

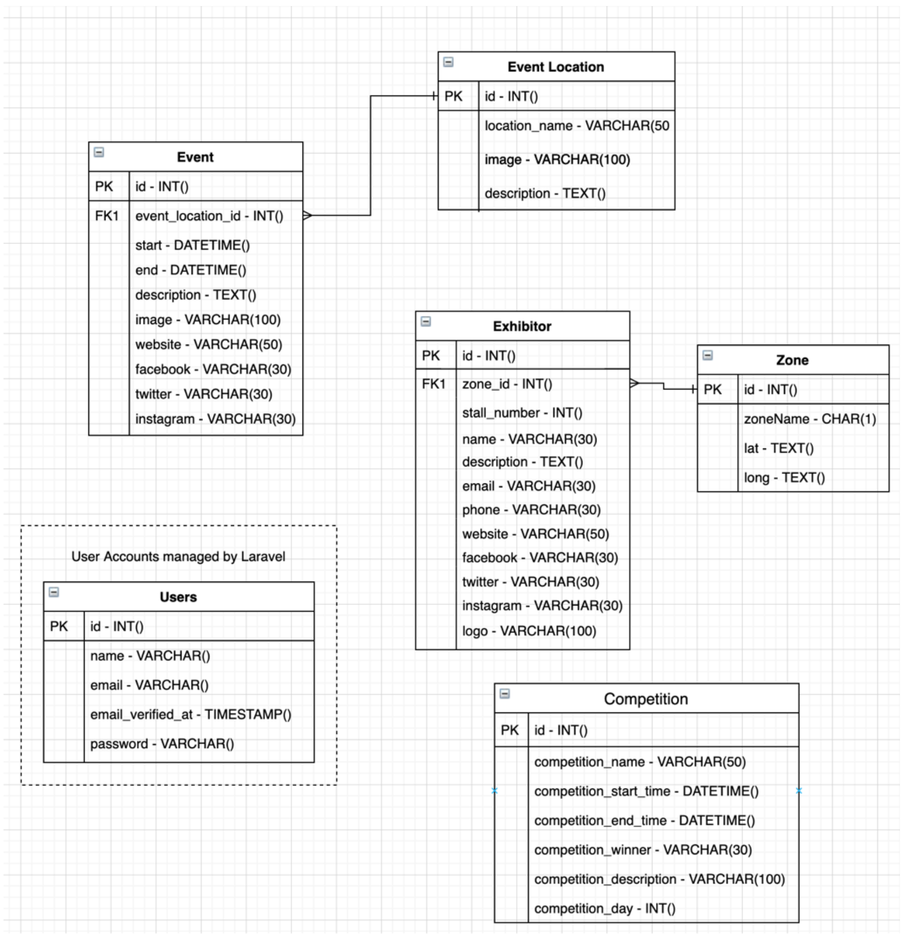


Start
Check if user is logged in
User is logged in
Check if its a valid Admin Panel endpoint
Valid Endpoint
Select appropriate controller and function
Get data from Database
Load data into blade template
Send response
User not logged in
Redirect user to login page
Invalid Endpoint
Return unknown page
Unable to get data from Database

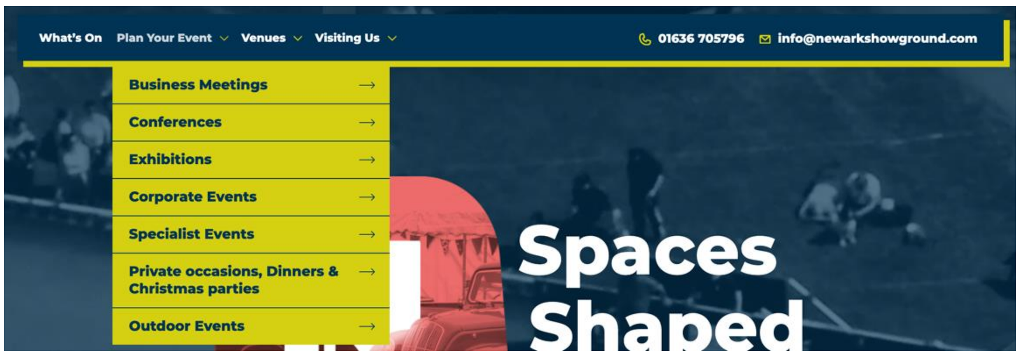
Appendix - 7



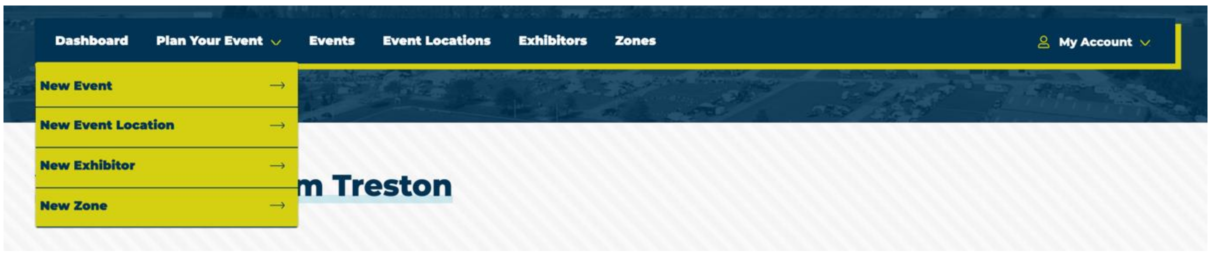
Appendix – 8



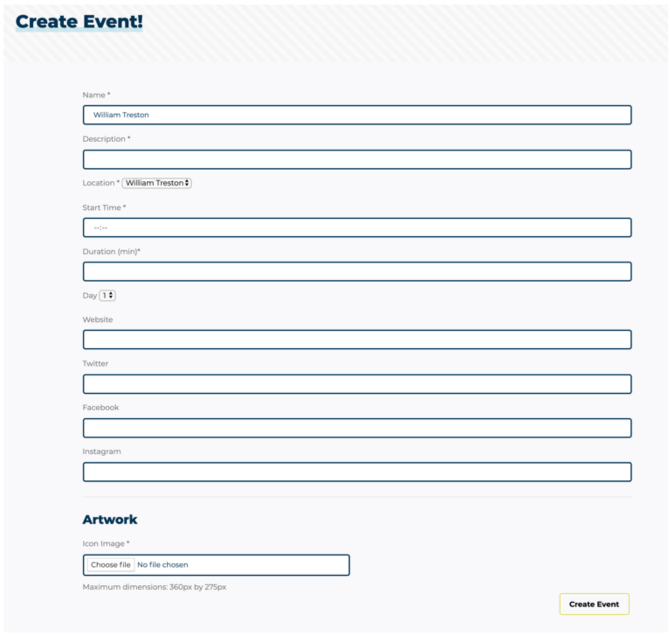
Appendix - 9



Appendix - 10



Appendix – 11



Create Event!

Name * William Treston

Description *

Location * William Treston

Start Time * --:--

Duration (min)*

Day 1

Website

Twitter

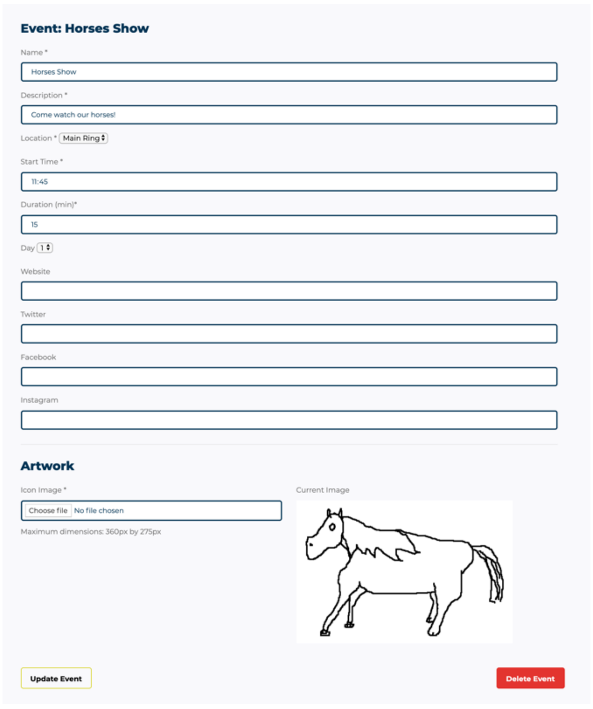
Facebook

Instagram

Artwork

Icon Image * Choose file No file chosen

Maximum dimensions: 360px by 275px



Event: Horses Show

Name * Horses Show

Description * Come watch our horses!

Location * Main Ring

Start Time * 11:45

Duration (min)* 15

Day 1

Website

Twitter

Facebook

Instagram

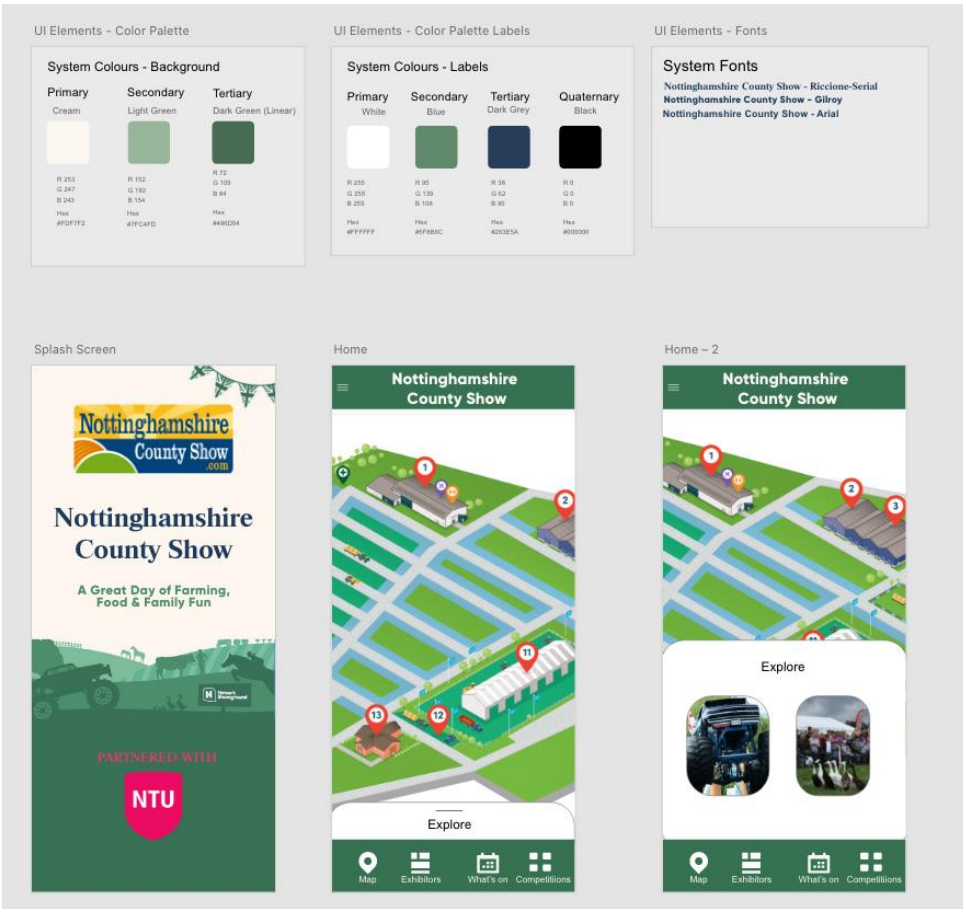
Artwork

Icon Image * Choose file No file chosen

Maximum dimensions: 360px by 275px

Current Image

Appendix - 12



UI Elements - Color Palette

System Colours - Background

Primary	Secondary	Tertiary
Cream	Light Green	Dark Green (Linear)
R 253	R 152	R 72
G 247	G 182	G 109
B 243	B 154	B 84
Hex #FDF7F2	Hex #7FC4FD	Hex #486D54

UI Elements - Color Palette Labels

System Colours - Labels

Primary	Secondary	Tertiary	Quaternary
White	Blue	Dark Grey	Black
R 255	R 95	R 38	R 0
G 255	G 139	G 62	G 0
B 255	B 108	B 90	B 0
Hex #FFFFFF	Hex #5F8B6C	Hex #263E5A	Hex #000000

UI Elements - Fonts

System Fonts

Nottinghamshire County Show - Riccione-Serial
Nottinghamshire County Show - Gilroy
Nottinghamshire County Show - Arial

Splash Screen
Nottinghamshire County Show .com
Nottinghamshire County Show
A Great Day of Farming, Food & Family Fun
PARTNERED WITH
NTU
Home
Nottinghamshire County Show
Explore
Map
Exhibitors
What's on
Competitions
Home – 2
Nottinghamshire County Show
Explore
Map
Exhibitors
What's on
Competitions

Appendix 13 – API Endpoint Documentation

Get All Event Locations - Working

Endpoint: /api/v1/events/event_locations/all

Request Type: GET

Description: Return a list of every location an event can take place at.

Returns:

```
[
    {
        "location_id": 2,
        "location_name": "location 1",
        "location_description": "123",
        "location_image": "rbumGDCRvJAI2Y3WDLedMc5XGAf3U3dkUlRYjWU7.jpeg"
    },
    {
        "location_id": 3,
        "location_name": "location 2",
        "location_description": "123",
        "location_image": "zhgAsEpSIYvzo2r7wMDqoa92BWkv85B1whwRScUb.jpeg"
    }
]
```

Get all Info about a specific Event Location - Working

Endpoint: /api/v1/events/event_locations/{location_id}

Request Type: GET
Description: Returns info about a specific Event Location

Returns:

```
{
    "location_id": 2,
    "location_name": "location 1",
    "location_description": "123",
    "location_image": "rbumGDCRvJAI2Y3WDLedMc5XGAf3U3dkUlRYjWU7.jpeg"
}
```

Get Event By Location - Working

Endpoint: /api/v1/events/events/events_by_location/{location_id}

Request Type: GET

Description: Returns a list of events specific to a location ID

Returns:

```
[
    {
        "event_id": 2,
        "event_name": "Horses show 1",
        "event_start_time": "11:34:00",
        "event_day": 1,
        "event_duration": 45,
        "event_description": "horses show",
        "event_image": "RBbTzVN7Ftk9C2YPiBkkF9ziEWgnCIatcjOtl6FH.gif",
        "event_website": null,
        "event_facebook": null,
        "event_twitter": null,
        "event_instagram": null,
        "event_location_id": 2,
        "event_location_info": {
            "location_id": 2,
            "location_name": "location 1",
            "location_description": "123",
            "location_image": "rbumGDCRvJAI2Y3WDLedMc5XGAf3U3dkUlRYjWU7.jpeg"
        }
    },
    {
        "event_id": 3,
        "event_name": "Horses show 2",
        "event_start_time": "11:34:00",
        "event_day": 1,
        "event_duration": 45,
        "event_description": "horses show",
        "event_image": "vSebZrGOkH1n0pb25ZkbwqYsMSoxKqYLqjZvHvI0.gif",
        "event_website": null,
        "event_facebook": null,
        "event_twitter": null,
        "event_instagram": null,
        "event_location_id": 2,
        "event_location_info": {
            "location_id": 2,
            "location_name": "location 1",
            "location_description": "123",
            "location_image": "rbumGDCRvJAI2Y3WDLedMc5XGAf3U3dkUlRYjWU7.jpeg"
        }
    }
]
```

Get a list of all events - Working

Endpoint: /api/v1/events/events/all

Request Type: GET

Description: Get a list of every event.

Returns:

```
[
    {
        "event_id": 2,
        "event_name": "Horses show 1",
        "event_start_time": "11:34:00",
        "event_day": 1,
        "event_duration": 45,
        "event_description": "horses show",
        "event_image": "RBbTzVN7Ftk9C2YPiBkkF9ziEWgnCIatcjOtl6FH.gif",
        "event_website": null,
        "event_facebook": null,
        "event_twitter": null,
        "event_instagram": null,
        "event_location_id": 2,
        "event_location_info": {
            "location_id": 2,
            "location_name": "location 1",
            "location_description": "123",
            "location_image": "rbumGDCRvJAI2Y3WDLedMc5XGAf3U3dkUlRYjWU7.jpeg"
        }
    },
    {
        "event_id": 3,
        "event_name": "Horses show 2",
        "event_start_time": "11:34:00",
        "event_day": 1,
        "event_duration": 45,
        "event_description": "horses show",
        "event_image": "vSebZrGOkH1n0pb25ZkbwqYsMSoxKqYLqjZvHvI0.gif",
        "event_website": null,
        "event_facebook": null,
        "event_twitter": null,
        "event_instagram": null,
        "event_location_id": 2,
        "event_location_info": {
            "location_id": 2,
            "location_name": "location 1",
            "location_description": "123",
            "location_image": "rbumGDCRvJAI2Y3WDLedMc5XGAf3U3dkUlRYjWU7.jpeg"
        }
    }
]
```

Get an events - Working

Endpoint: /api/v1/events/events /{event_id}

Request Type: GET

Description: Get an event

Returns:

```
{
    "event_id": 2,
    "event_name": "Horses",
    "event_start_time": "11:34:00",
    "event_day": 1,
    "event_duration": 45,
    "event_description": "horses show",
    "event_image": "RBbTzVN7Ftk9C2YPiBkkF9ziEWgnCIatcjOtl6FH.gif",
    "event_website": null,
    "event_facebook": null,
    "event_twitter": null,
    "event_instagram": null,
    "event_location_id": 2,
    "event_location_info": {
        "location_id": 2,
        "location_name": "location 1",
        "location_description": "123",
        "location_image": "rbumGDCRvJAI2Y3WDLedMc5XGAf3U3dkUlRYjWU7.jpeg"
    }
}
```

Get all exhibitors - Working

Endpoint: /api/v1/exhibitors/all

Request Type: GET

Description: Returns a list of every exhibitor

Returns:

```
[
    {
        "exhibitor_id": 1,
        "zone_id": 1,
        "exhibitor_stall_number": 123,
        "exhibitor_name": "William Treston",
        "exhibitor_description": "123",
        "exhibitor_email": "badger1661@gmail.com",
        "exhibitor_phone": "07460151118",
        "exhibitor_website": null,
        "exhibitor_facebook": null,
        "exhibitor_twitter": null,
        "exhibitor_instagram": null,
        "exhibitor_logo_url": "4zCiIvrnJJa0tvy6FeGR9Tc8E601yIniMvQBiYzX.gif",
        "zone_info": [
            {
                "id": 1,
                "zone_name": "A",
                "lat": "53.09987774367113",
                "long": "-0.769100710862773"
            }
        ]
    },
    {
        "exhibitor_id": 2,
        "zone_id": 1,
        "exhibitor_stall_number": 13,
        "exhibitor_name": "pizza",
        "exhibitor_description": "123123",
        "exhibitor_email": "badger1661@gmail.com",
        "exhibitor_phone": "07460151118",
        "exhibitor_website": null,
        "exhibitor_facebook": null,
        "exhibitor_twitter": null,
        "exhibitor_instagram": null,
        "exhibitor_logo_url": "v0zcttHzHyTVMEEITyMTw1Li6awITqTaqCdlABSH.png",
        "zone_info": [
            {
                "id": 1,
                "zone_name": "A",
                "lat": "53.09987774367113",
                "long": "-0.769100710862773"
            }
        ]
    }
]
```

Get an Exhibitor - Working

Endpoint: /api/v1/exhibitors/exhibitor/{exhibitor_id}

Request Type: GET

Description: Returns a single exhibitor

Returns:

```
{
    "exhibitor_id": 1,
    "zone_id": 1,
    "exhibitor_stall_number": 123,
    "exhibitor_name": "William Treston",
    "exhibitor_description": "123",
    "exhibitor_email": "badger1661@gmail.com",
    "exhibitor_phone": "07460151118",
    "exhibitor_website": null,
    "exhibitor_facebook": null,
    "exhibitor_twitter": null,
    "exhibitor_instagram": null,
    "exhibitor_logo_url": "4zCiIvrnJJa0tvy6FeGR9Tc8E601yIniMvQBiYzX.gif",
    "zone_info": [
        {
            "id": 1,
            "zone_name": "A",
            "lat": "53.09987774367113",
            "long": "-0.769100710862773"
        }
    ]
}
```

<u>Get Exhibitors by Zone ID - Working</u>

Endpoint: /api/v1/exhibitors/by_location/{location_id }

Request Type: GET

Description: Returns all exhibitors in a given location

Returns:

```
[
    {
        "exhibitor_id": 1,
        "zone_id": 1,
        "exhibitor_stall_number": 123,
        "exhibitor_name": "William Treston",
        "exhibitor_description": "123",
        "exhibitor_email": "badger1661@gmail.com",
        "exhibitor_phone": "07460151118",
        "exhibitor_website": null,
        "exhibitor_facebook": null,
        "exhibitor_twitter": null,
        "exhibitor_instagram": null,
        "exhibitor_logo_url": "4zCiIvrnJJa0tvy6FeGR9Tc8E601yIniMvQBiYzX.gif",
        "zone_info": {
            "id": 1,
            "zone_name": "A",
            "lat": "53.09987774367113",
            "long": "-0.769100710862773"
        }
    },
    {
        "exhibitor_id": 2,
        "zone_id": 1,
        "exhibitor_stall_number": 13,
        "exhibitor_name": "pizza",
        "exhibitor_description": "123123",
        "exhibitor_email": "badger1661@gmail.com",
        "exhibitor_phone": "07460151118",
        "exhibitor_website": null,
        "exhibitor_facebook": null,
        "exhibitor_twitter": null,
        "exhibitor_instagram": null,
        "exhibitor_logo_url": "v0zcttHzHyTVMEEITyMTw1Li6awITqTaqCdlABSH.png",
        "zone_info": {
            "id": 1,
            "zone_name": "A",
            "lat": "53.09987774367113",
            "long": "-0.769100710862773"
        }
    }
]
```

<u>Get Zones - Working</u>

Endpoint: /api/v1/zones/all

Request Type: GET

Description: Returns all zones

Returns:

```
[
    {
        "id": 1,
        "zone_name": "A",
        "lat": "53.09987774367113",
        "long": "-0.769100710862773"
    },
    {
        "id": 2,
        "zone_name": "B",
        "lat": "53.097929789015666",
        "long": "-0.7696050734585071"
    }
]
```

Get Zone by ID - Working

Endpoint: /api/v1/zones/by_zone_id/{zone_id}

Request Type: GET

Description: Returns zone info about the specified ID

Returns:

```
{
    "id": 1,
    "zone_name": "A",
    "lat": "53.09987774367113",
    "long": "-0.769100710862773"
}
```

Get all Competitions - Working

Endpoint: /api/v1/competitions/all

Request Type: GET

Description: Returns info about all competitions

Returns:

```
[
    {
        "competition_id": 4,
        "competition_name": "William Treston",
        "competition_day": 1,
        "competition_start_time": "11:11:00",
        "competition_end_time": "12:45:00",
        "competition_winner": null,
        "competition_description": "123"
    }
]
```

Get Competitions - Working

Endpoint: /api/v1/competitions/view_competition/{competition_id}

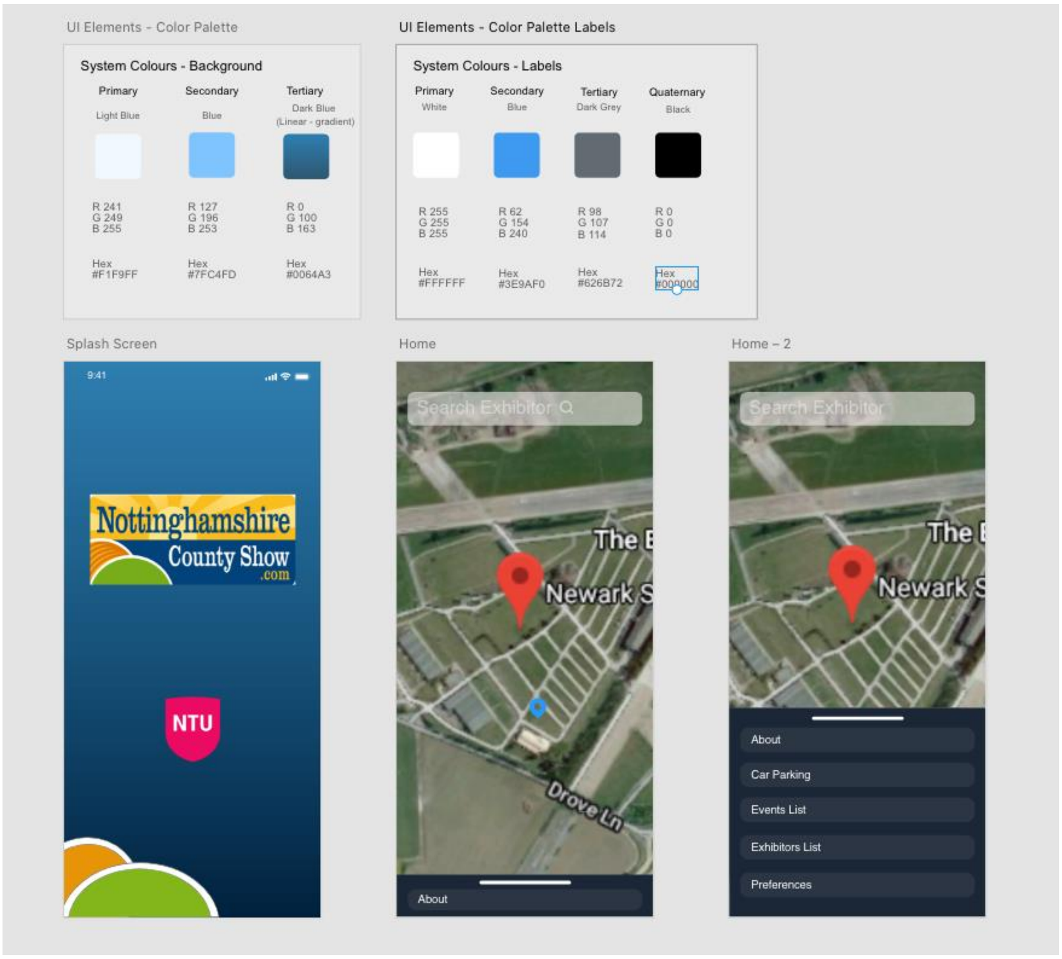
Request Type: GET

Description: Returns info about all competitions

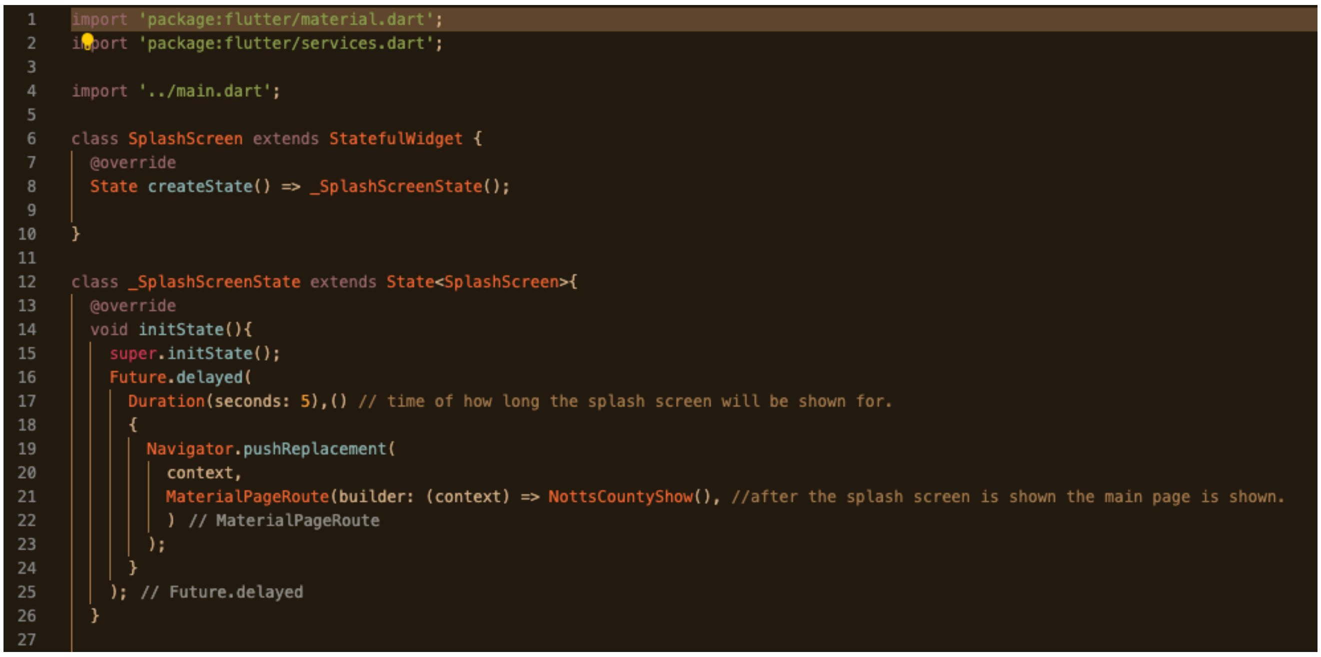
Returns:

```
{
    "competition_id": 4,
    "competition_name": "William Treston",
    "competition_day": 1,
    "competition_start_time": "11:11:00",
    "competition_end_time": "12:45:00",
    "competition_winner": null,
    "competition_description": "123"
}
```

Appendix - 14

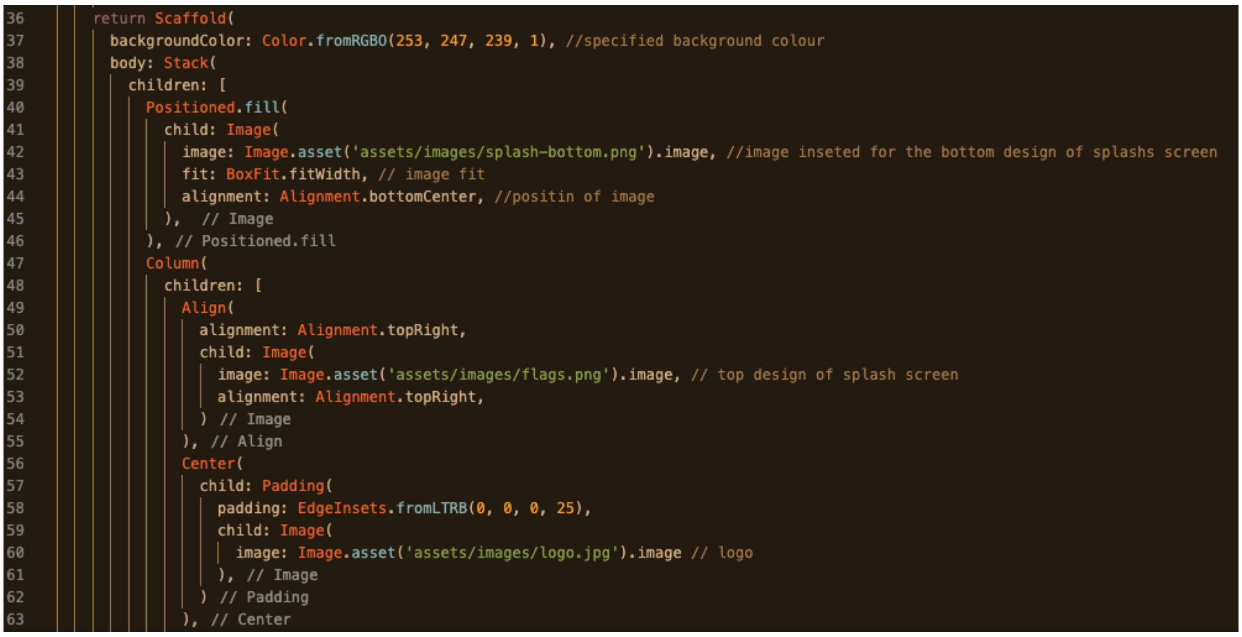


Appendix - 15



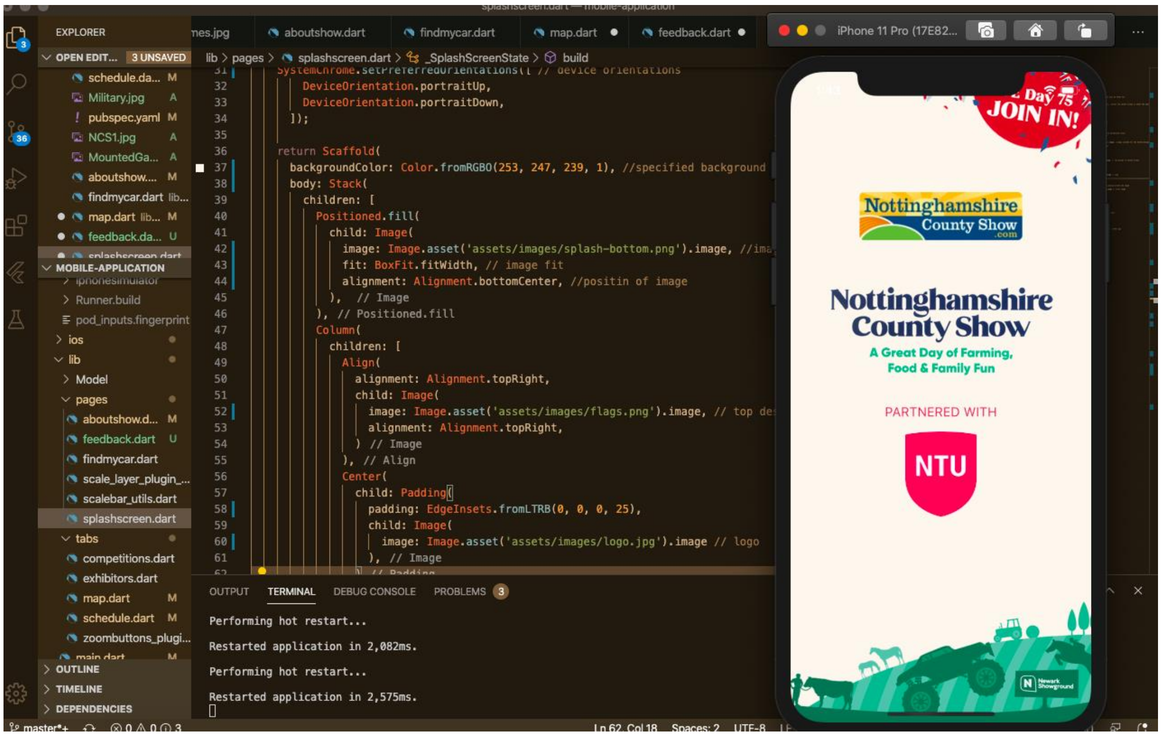
```dart
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';

import '../main.dart';

class SplashScreen extends StatefulWidget {
  @override
  State createState() => _SplashScreenState();

}

class _SplashScreenState extends State<SplashScreen>{
  @override
  void initState(){
    super.initState();
    Future.delayed(
      Duration(seconds: 5),() // time of how long the splash screen will be shown for.
      {
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (context) => NottsCountyShow(), //after the splash screen is shown the main page is shown.
          ) // MaterialPageRoute
        );
      }
    ); // Future.delayed
  }

```

Appendix - 16



```
    return Scaffold(
      backgroundColor: Color.fromRGBO(253, 247, 239, 1), //specified background colour
      body: Stack(
        children: [
          Positioned.fill(
            child: Image(
              image: Image.asset('assets/images/splash-bottom.png').image, //image inseted for the bottom design of splashs screen
              fit: BoxFit.fitWidth, // image fit
              alignment: Alignment.bottomCenter, //positin of image
            ),  // Image
          ), // Positioned.fill
          Column(
            children: [
              Align(
                alignment: Alignment.topRight,
                child: Image(
                  image: Image.asset('assets/images/flags.png').image, // top design of splash screen
                  alignment: Alignment.topRight,
                ) // Image
              ), // Align
              Center(
                child: Padding(
                  padding: EdgeInsets.fromLTRB(0, 0, 0, 25),
                  child: Image(
                    image: Image.asset('assets/images/logo.jpg').image // logo
                  ), // Image
                ) // Padding
              ), // Center
```

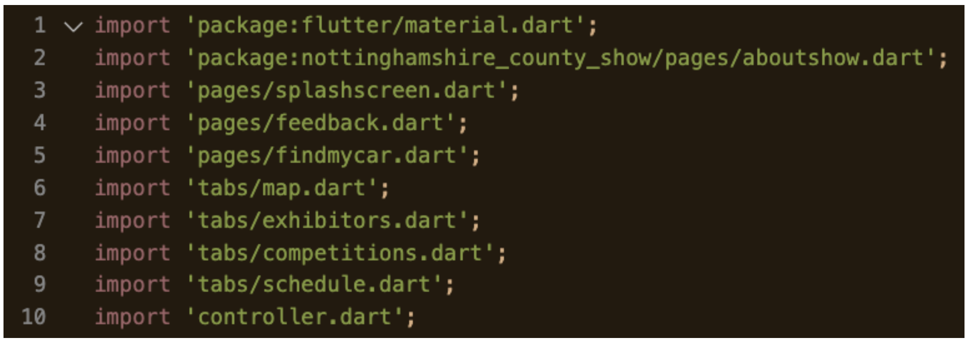
Appendix - 17



```
    SystemChrome.setPreferredOrientations([ // device orientations
      DeviceOrientation.portraitUp,
      DeviceOrientation.portraitDown,
    ]);

    return Scaffold(
      backgroundColor: Color.fromRGBO(253, 247, 239, 1), //specified background
      body: Stack(
        children: [
          Positioned.fill(
            child: Image(
              image: Image.asset('assets/images/splash-bottom.png').image, //ima
              fit: BoxFit.fitWidth, // image fit
              alignment: Alignment.bottomCenter, //positin of image
            ),  // Image
          ), // Positioned.fill
          Column(
            children: [
              Align(
                alignment: Alignment.topRight,
                child: Image(
                  image: Image.asset('assets/images/flags.png').image, // top de
                  alignment: Alignment.topRight,
                ) // Image
              ), // Align
              Center(
                child: Padding(
                  padding: EdgeInsets.fromLTRB(0, 0, 0, 25),
                  child: Image(
                    image: Image.asset('assets/images/logo.jpg').image // logo
                  ), // Image
```

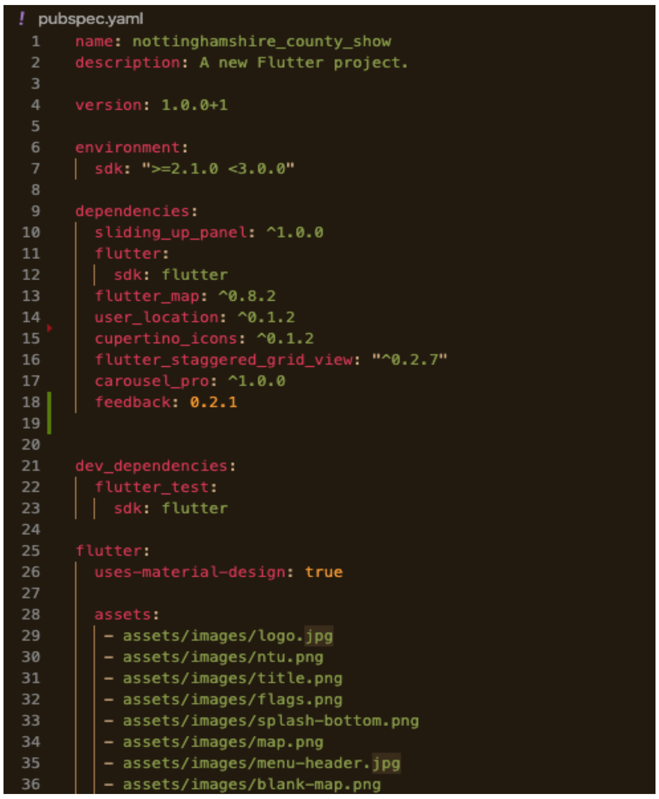
```
Performing hot restart...

Restarted application in 2,082ms.

Performing hot restart...

Restarted application in 2,575ms.
```

Appendix - 18



```dart
import 'package:flutter/material.dart';
import 'package:nottinghamshire_county_show/pages/aboutshow.dart';
import 'pages/splashscreen.dart';
import 'pages/feedback.dart';
import 'pages/findmycar.dart';
import 'tabs/map.dart';
import 'tabs/exhibitors.dart';
import 'tabs/competitions.dart';
import 'tabs/schedule.dart';
import 'controller.dart';
```

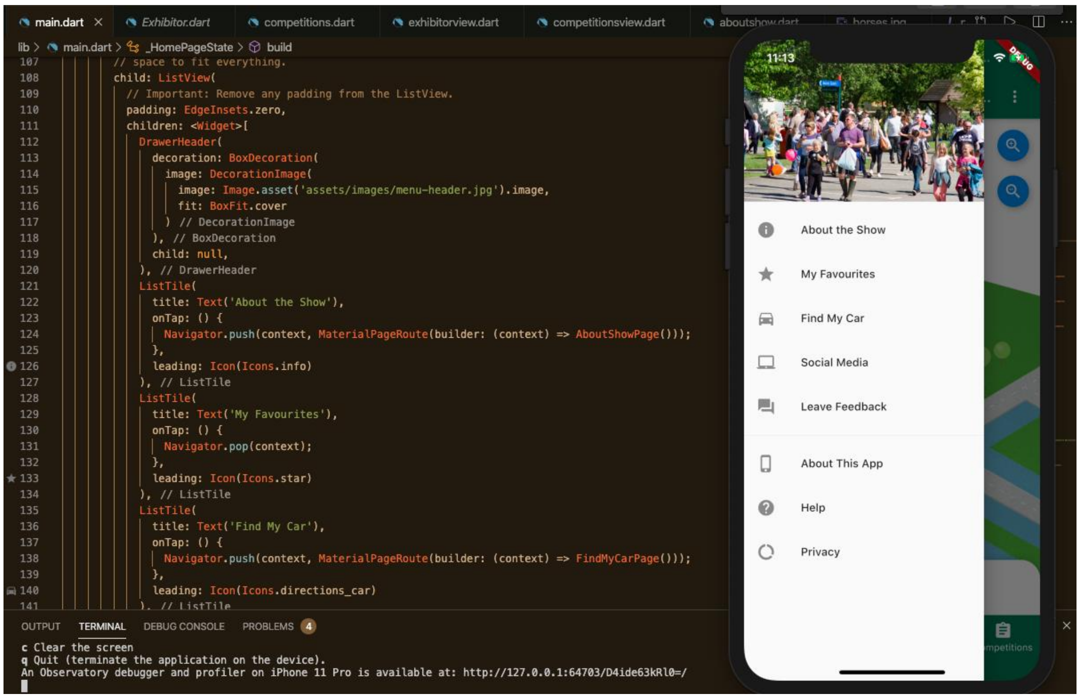
Appendix - 19



pubspec.yaml

```yaml
name: nottinghamshire_county_show
description: A new Flutter project.

version: 1.0.0+1

environment:
  sdk: ">=2.1.0 <3.0.0"

dependencies:
  sliding_up_panel: ^1.0.0
  flutter:
    sdk: flutter
  flutter_map: ^0.8.2
  user_location: ^0.1.2
  cupertino_icons: ^0.1.2
  flutter_staggered_grid_view: "^0.2.7"
  carousel_pro: ^1.0.0
  feedback: 0.2.1


dev_dependencies:
  flutter_test:
    sdk: flutter

flutter:
  uses-material-design: true

  assets:
    - assets/images/logo.jpg
    - assets/images/ntu.png
    - assets/images/title.png
    - assets/images/flags.png
    - assets/images/splash-bottom.png
    - assets/images/map.png
    - assets/images/menu-header.jpg
    - assets/images/blank-map.png
```

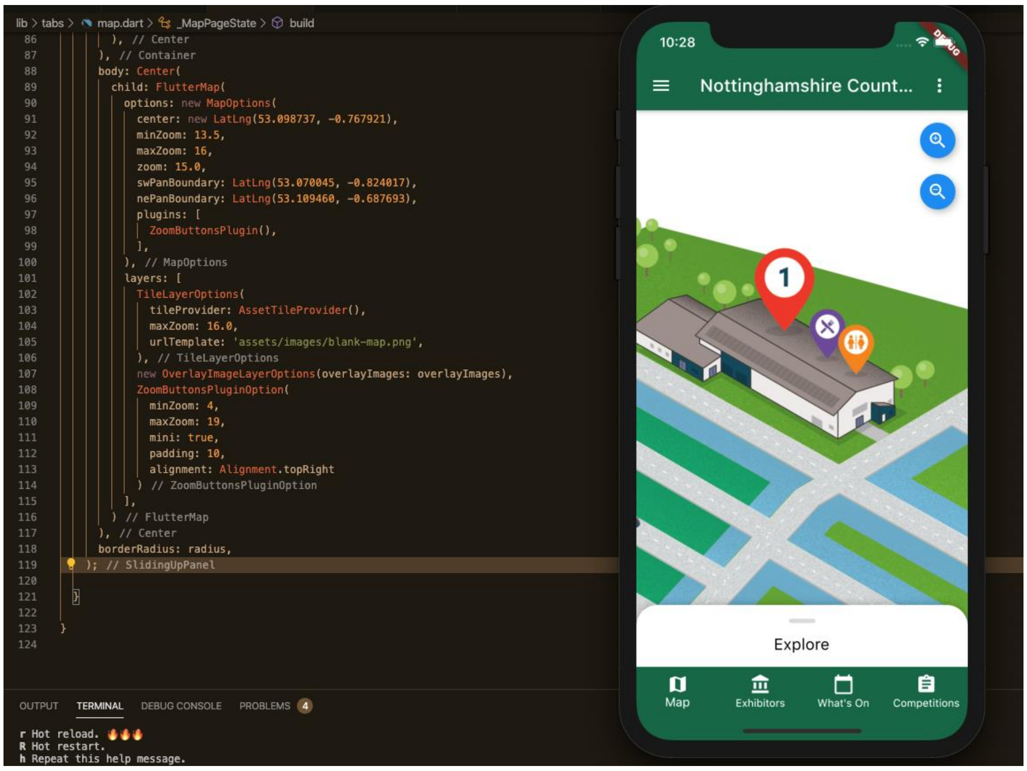
Appendix - 20



```dart
          // space to fit everything.
          child: ListView(
            // Important: Remove any padding from the ListView.
            padding: EdgeInsets.zero,
            children: <Widget>[
              DrawerHeader(
                decoration: BoxDecoration(
                  image: DecorationImage(
                    image: Image.asset('assets/images/menu-header.jpg').image,
                    fit: BoxFit.cover
                  ) // DecorationImage
                ), // BoxDecoration
                child: null,
              ), // DrawerHeader
              ListTile(
                title: Text('About the Show'),
                onTap: () {
                  Navigator.push(context, MaterialPageRoute(builder: (context) => AboutShowPage()));
                },
                leading: Icon(Icons.info)
              ), // ListTile
              ListTile(
                title: Text('My Favourites'),
                onTap: () {
                  Navigator.pop(context);
                },
                leading: Icon(Icons.star)
              ), // ListTile
              ListTile(
                title: Text('Find My Car'),
                onTap: () {
                  Navigator.push(context, MaterialPageRoute(builder: (context) => FindMyCarPage()));
                },
                leading: Icon(Icons.directions_car)
              ), // ListTile
```

```
c Clear the screen
q Quit (terminate the application on the device).
An Observatory debugger and profiler on iPhone 11 Pro is available at: http://127.0.0.1:64703/D4ide63kRl0=/
```

Appendix - 21



```dart
          ), // Center
        ), // Container
        body: Center(
          child: FlutterMap(
            options: new MapOptions(
              center: new LatLng(53.098737, -0.767921),
              minZoom: 13.5,
              maxZoom: 16,
              zoom: 15.0,
              swPanBoundary: LatLng(53.070045, -0.824017),
              nePanBoundary: LatLng(53.109460, -0.687693),
              plugins: [
                ZoomButtonsPlugin(),
              ],
            ), // MapOptions
            layers: [
              TileLayerOptions(
                tileProvider: AssetTileProvider(),
                maxZoom: 16.0,
                urlTemplate: 'assets/images/blank-map.png',
              ), // TileLayerOptions
              new OverlayImageLayerOptions(overlayImages: overlayImages),
              ZoomButtonsPluginOption(
                minZoom: 4,
                maxZoom: 19,
                mini: true,
                padding: 10,
                alignment: Alignment.topRight
              ) // ZoomButtonsPluginOption
            ],
          ) // FlutterMap
        ), // Center
        borderRadius: radius,
      ); // SlidingUpPanel

    }

  }
```

```
r Hot reload. 🔥🔥🔥
R Hot restart.
h Repeat this help message.
```

Appendix 22 – Functionality Test table for Admin Panel

TEST NUMBER	TEST DESCRIPTION	EXPECTED OUTCOME	ACTUAL OUTCOME	PASS/FAIL	FIX
EVENT LOCATIONS					
1	Create a new Event Location	Data is validated and new Event Location object created, saved to the database and user redirected to the Event Locations page	Data was validated, a new object was saved to the database and user was redirected to the Event Locations page	PASS	N/A
2	Update an Event location	New data is validated, and stored in the database and user is displayed the new data	Data was validated, updated in the database and shown the new data	PASS	N/A
3	Go to Event Locations Page	User is redirected to Event Locations Page	User was redirected to Event Locations Page	PASS	N/A
4	Filter Events in Event location	Events happening in the Event Location are filtered by keywords and the Events are displayed	Events were correctly filtered and displayed to the user	PASS	N/A
5	Delete Event Location	Event Location and all Events happening at the Event Location are deleted and user redirected to Event Locations page	Event Location and Events happening at that Event Location are deleted and user was redirected to Event Locations page	PASS	N/A
EVENTS					

6	Filter Events by name	Events are filtered by keywords and displayed	Events were filtered by keywords and displayed	PASS	N/A
7	Create new event	Data is validated and new Event object created, saved to the database and user redirected to the Event’s page	Data was validated, a new object was saved to the database and user was redirected to the Event’s page	PASS	N/A
8	Update Event	New data is validated, and stored in the database and user is displayed the new data	Data was validated, updated in the database and shown the new data	PASS	N/A
9	Delete Event	Event is deleted and user redirected to events page	Event was deleted and user was redirected to events page	PASS	N/A
			EXHIBITORS		
10	Create a new Exhibitor	Data is validated and new Exhibitor object created, saved to the database and user redirected to the Exhibitor’s page	Data was validated, a new object was saved to the database and user was redirected to the Exhibitor’s page	PASS	N/A
11	Update an Exhibitor	New data is validated, and stored in the database and user is displayed the new data	Data was validated, updated in the database and shown the new data	PASS	N/A
12	Go to Exhibitor’s Page	User is redirected to Exhibitor’s Page	User was redirected to Exhibitor’s Page	PASS	N/A
13	Delete Exhibitor	Exhibitor is deleted	Exhibitor was deleted	PASS	N/A
14	Filter Exhibitors by name	Exhibitors are filtered by keywords and displayed	Exhibitors were filtered by keywords and displayed	PASS	N/A

			ZONES		
15	Create a new Zone	Data is validated and new Zone object created, saved to the database and user redirected to the Zone page	Data was validated, a new object was saved to the database and user was redirected to the Zone page	PASS	N/A
16	Update a Zone	New data is validated, and stored in the database and user is displayed the new data	Data was validated, updated in the database and shown the new data	PASS	N/A
17	Go to Zone Page	User is redirected to Zone Page	User was redirected to Zone Page	PASS	N/A
18	Delete Zone	Zone is deleted	Zone was deleted	PASS	N/A
19	Filter Zones	Zones are filtered by keywords and displayed	No Zones were displayed	FAIL	Logical error in an if statement being "==" instead of "!=".
			HOMEPAGE		
20	Display the home page	Displays accurate data about Events, Event Locations, Exhibitors and Zones	Displayed accurate data about Events, Event Locations, Exhibitors and Zones	PASS	N/A
			LOGIN/SIGNUP PAGE		
21	Create new account	Data is verified and user is signed in and redirected to '/home'	Data was verified and user is signed in and redirected to '/home'	PASS	N/A
22	Log in	User is signed in and redirected to '/home'	User was signed in and redirected to '/home'	PASS	N/A
			MISCELLANEOUS TESTS		
23	Try accessing a protected page	User redirected to '/login'	User allowed to access he page	FAIL	Ensure all routes are in the right section
24	Update Event/Event Location, Exhibitor,	No new data saved to database and user is	No new data saved to database and user was	PASS	N/A

	Zone without changing details	shown the Event/Event Location, Exhibitor, Zone's page	shown the Event/Event Location, Exhibitor, Zone's page		
25	Try to submit a form that hasn't had all its mandatory fields filled in	User is unable to submit the form and is prompted to provide all required information	User was unable to submit the form and was prompted to provide all required information	PASS	N/A
			COMPETITIONS		
26	Create a new Competition	Data is validated and new Competition object created, saved to the database and user redirected to the Competition page	Data was validated, a new object was saved to the database and user was redirected to the Competition page	PASS	N/A
27	Update a Competition	New data is validated, and stored in the database and user is displayed the new data	Data was validated, updated in the database and shown the new data	PASS	N/A
28	Go to Competition Page	User is redirected to Competition Page	User was redirected to Competition Page	PASS	N/A
29	Delete Competition	Zone is deleted	Competition was deleted	PASS	N/A
	Filter Competitions	Competition are filtered by keywords and displayed	Competition were filtered by keywords and displayed	PASS	N/A

Appendix 23 – API Functionality testing

Note that for the “Complete Endpoints” column, each endpoint has a prefix of “/api/v1”

TEST NUMBER	TEST DESCRIPTION	COMPLETE ENDPOINT	EXPECTED OUTCOME	ACTUAL OUTCOME	PASS/FAIL	FIX
		EVEENT LOCATIONS				
1	Get information on all Event Locations with valid data	/events/event_locations/all	JSON object returned with all information on every Event Location	JSON object returned with all information on every Event Location	PASS	N/A
2	Get Information on an Event Location with valid data	/events/event_locations/1	JSON object returned with information about specific Event Location	JSON object returned with information about specific Event Location	PASS	N/A
3	Get Information on an Event Location with invalid data	/events/event_locations/1000	404 error	404 error	PASS	N/A
4	Get Events in a specific Location with valid data	/events/events/events_by_location/1	JSON object with information about Events occurring in the specified Event Location	JSON object with information about Events occurring in the specified Event Location	PASS	N/A
		EVENTS				
5	Get Events in a specific Location with invalid data	/events/events/events_by_location/asd	Returns 404 error	Returns 404 error	PASS	N/A
6	Get All events	/events/events/all	Returns JSON object with	Returned JSON object with	PASS	N/A

			information on every Event	information on every Event		
7	Get a specific Event with valid data	/events/events/1	Returns a JSON object with the information	Returned a JSON object with the information	PASS	N/A
8	Get a specific Event with invalid data	/events/events/123	Returns 404 error	Returns 404 error	PASS	N/A
			EXHIBITORS			
9	Get all exhibitors information	/exhibitors/all	Returns JSON object with information about every exhibitor	Returned JSON object with information about every exhibitor	PASS	N/A
10	Get an Exhibitors information with valid data	/exhibitors/exhibitor/12	Returns JSON object with Exhibitors information	Didn't return the Exhibitors Zone data	FAIL	Wasn't querying the database for the Zone after retrieving the Exhibitor
11	Get an Exhibitors information with invalid data	/exhibitors/exhibitor/aaa	Returns 404 error	Returns 404 error	PASS	N/A
12	Get Exhibitors within a Zone with valid data	/exhibitors/by_location/1	Returns a JSON object with the data about each Exhibitor in a specific Zone	Return a JSON object with the data about each Exhibitor in a specific Zone	PASS	N/A
13	Get Exhibitors within a Zone with invalid data	/exhibitors/by_location/asd	Returns 404 error	Returns 404 error	PASS	N/A
			ZONES			

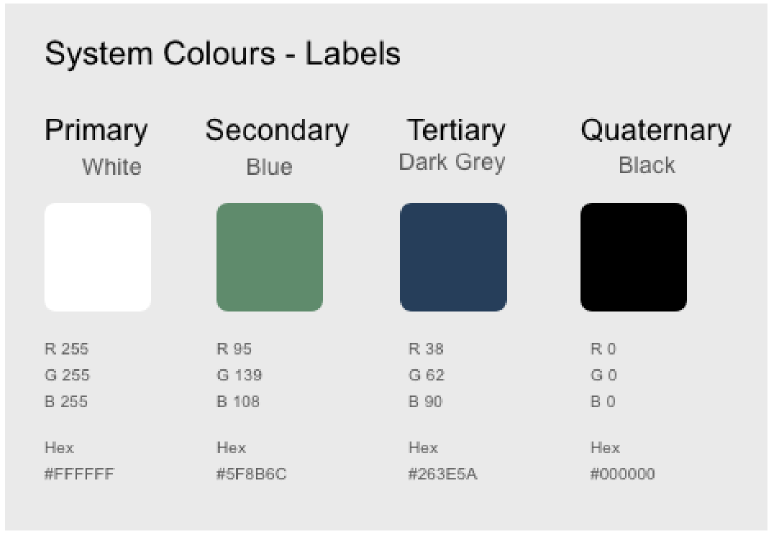
14	Get information on all Zones	/zones/all	Returns JSON object with information on every Zone	Returned JSON object with information on every Zone	PASS	N/A
15	Get information on a specific Zone with valid data	/zones/by_zone_id/1	Returns JSON object with information on a specific Zone	Returned JSON object with information on a specific Zone	PASS	N/A
16	Get information on a specific Zone with invalid data	/zones/by_zone_id/asd	Returns 404 error	Returns 404 error	PASS	N/A
			COMPETITIONS			
17	Get all information on competitions	/competitions/all	Returns JSON object with information about each competition	Returned JSON object with information about each competition	PASS	N/A
18	Get information on a single competition with valid data	/competitions/view_ competition/1	Returns JSON object with information about specific competition	Returned JSON object with information about specific competition	PASS	N/A
19	Get information on a single competition with invalid data	/competitions/view_ competition/asd	Returns 404 error	Returned 404 error	PASS	N/A

Appendix 24 – Integration Testing

TEST NUMBER	TEST DESCRIPTION	COMPLETE ENDPOINT	EXPECTED OUTCOME	ACTUAL OUTCOME	PASS/FAIL	FIX
			EVEENT LOCATIONS			
1	Get all events with API and display on app	/events/event_locations/all	Events are displayed on App	Events were displayed on App	PASS	N/A
2	Get info on 1 event and display on app	/events/event_locations/1	Event is displayed on app	Event was displayed on app	PASS	N/A
3	Get Events in a specific Location and display on app	/events/events/events_by_location/1	Events are displayed on App	Events were displayed on App	PASS	N/A
			EVENTS			
4	Get All events and display on app	/events/events/all	Events are displayed on app	Events were displayed on app	PASS	N/A
5	Get a specific Event display on app	/events/events/1	Event is displayed on app	Event was displayed on app	PASS	N/A
			EXHIBITORS			
6	Get all exhibitors information and display on app	/exhibitors/all	Exhibitors are displayed on the app	Exhibitors were displayed on the app	PASS	N/A
7	Get an Exhibitors information and display on app	/exhibitors/exhibitor/12	Exhibitor is displayed on the app	Exhibitor was displayed on the app	PASS	N/A
8	Get Exhibitors within a Zone and display on app	/exhibitors/by_location/1	Exhibitors are displayed on the app	Exhibitors were displayed on the app	PASS	N/A

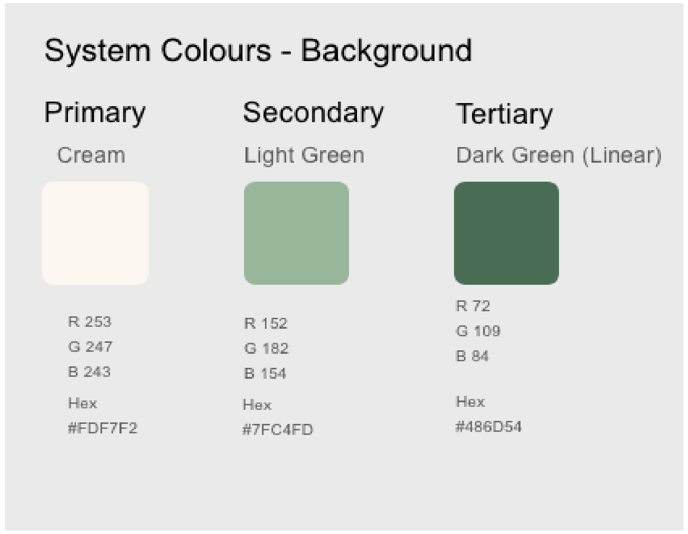
			ZONES			
9	Get information on all Zones and and display on app	/zones/all	Zones are displayed on app	Zones were displayed on app	PASS	N/A
10	Get information on a specific Zone and display on app	/zones/by_zone_id/1	Zone is displayed on app	Zone was displayed on app	PASS	N/A
			COMPETITIONS			
11	Get all information on competitions and display on app	/competitions/all	Competitions are displayed on app	Competitions were displayed in app	PASS	N/A
12	Get information on a single competition and display on app	/competitions/view_ competition/1	Competition is displayed on app.	Competitions was displayed on app.	PASS	N/A

Appendix 25 – User Interface Designs



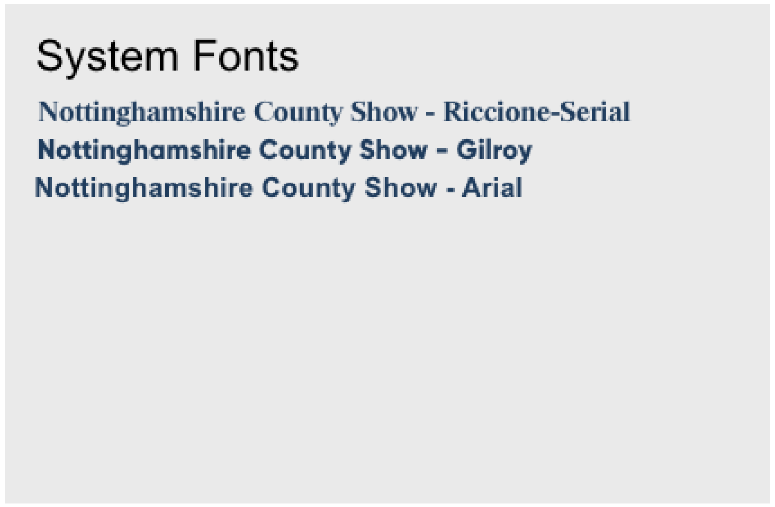
System Colours - Labels

Primary	Secondary	Tertiary	Quaternary
White	Blue	Dark Grey	Black
⬜	🟩	🟦	⬛
R 255	R 95	R 38	R 0
G 255	G 139	G 62	G 0
B 255	B 108	B 90	B 0
Hex	Hex	Hex	Hex
#FFFFFF	#5F8B6C	#263E5A	#000000



System Colours - Background

Primary	Secondary	Tertiary
Cream	Light Green	Dark Green (Linear)
⬜	🟩	🟩
R 253	R 152	R 72
G 247	G 182	G 109
B 243	B 154	B 84
Hex	Hex	Hex
#FDF7F2	#7FC4FD	#486D54

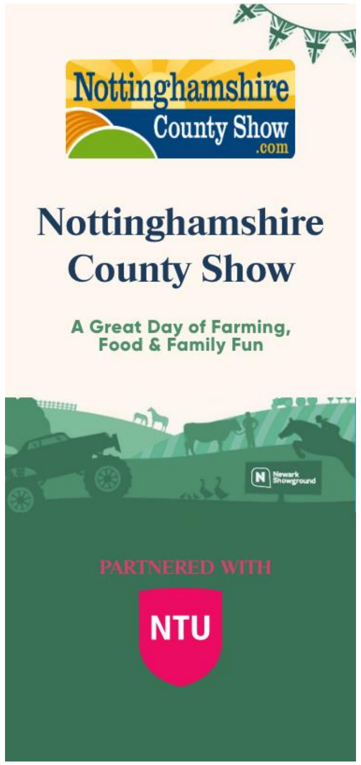


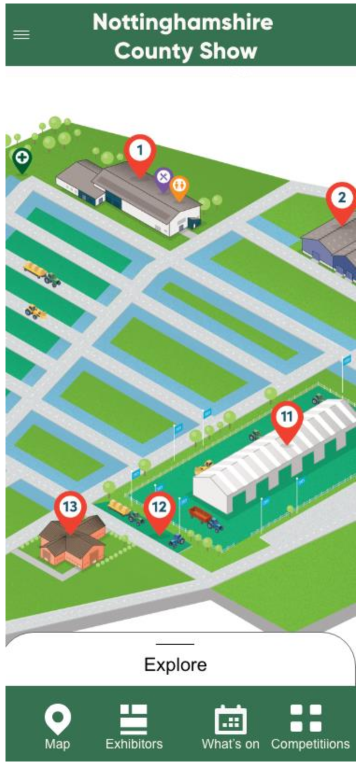
System Fonts

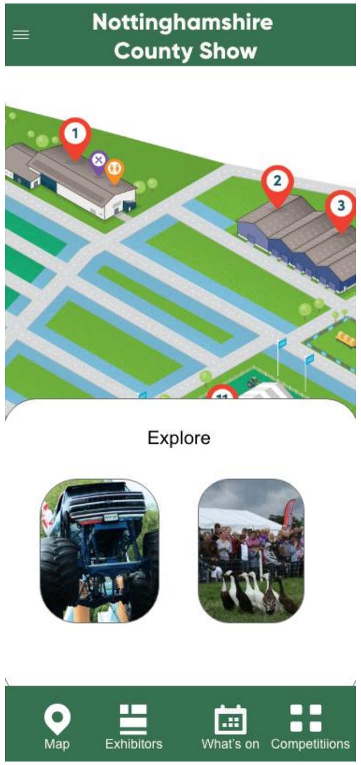
Nottinghamshire County Show - Riccione-Serial

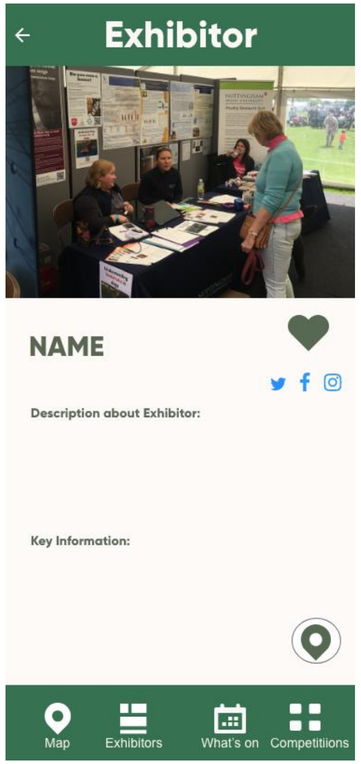
Nottinghamshire County Show - Gilroy

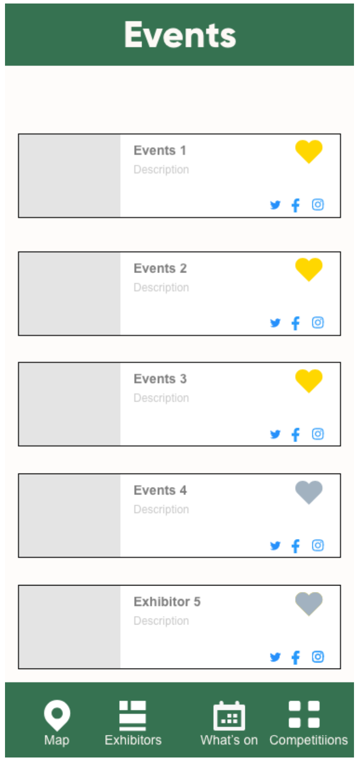
Nottinghamshire County Show - Arial

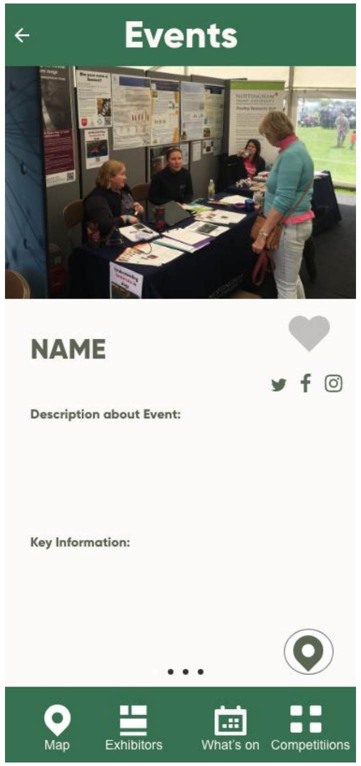

Nottinghamshire County Show .com
Nottinghamshire County Show
A Great Day of Farming, Food & Family Fun
Newark Showground
PARTNERED WITH
NTU

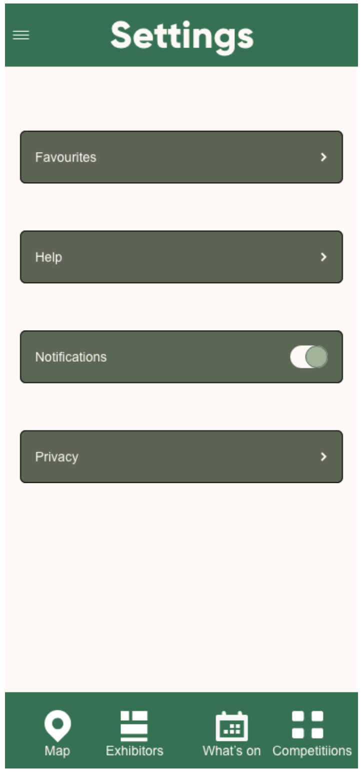

Nottinghamshire County Show
Explore
Map
Exhibitors
What's on
Competitions

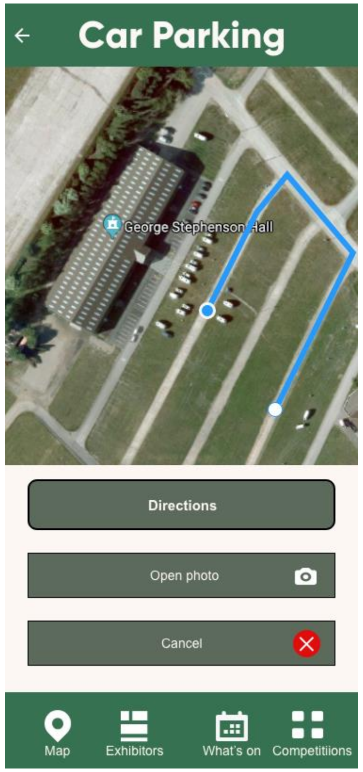

Nottinghamshire County Show
Explore
Map
Exhibitors
What's on
Competitions

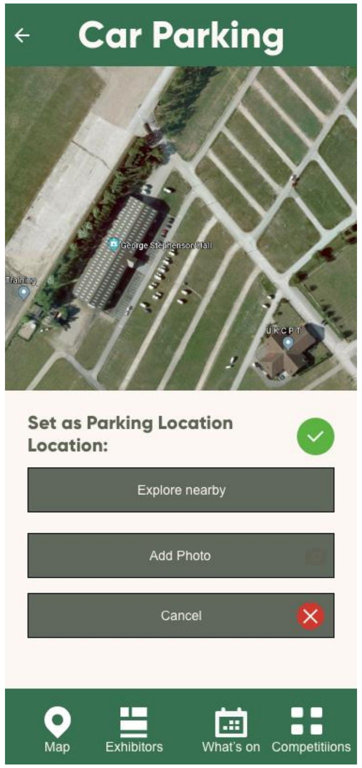

Exhibitor
NAME
Description about Exhibitor:
Key Information:
Map
Exhibitors
What's on
Competitions


Events
Events 1
Description
Events 2
Description
Events 3
Description
Events 4
Description
Exhibitor 5
Description
Map
Exhibitors
What's on
Competitions


Events
NAME
Description about Event:
Key Information:
Map
Exhibitors
What's on
Competitions


Settings
Favourites
Help
Notifications
Privacy
Map
Exhibitors
What's on
Competitions


Car Parking
George Stephenson Hall
Directions
Open photo
Cancel
Map
Exhibitors
What's on
Competitions


Car Parking
George Stephenson Hall
Set as Parking Location Location:
Explore nearby
Add Photo
Cancel
Map
Exhibitors
What's on
Competitions