Anglia Ruskin University

Developing Web Application Report

Holiday Booking Application

SID: 1726592

MOD002701

Submitted: May 2019

Table of Contents

1 Introduction

This report gives an overview of the processes that occur when building a simple web application for booking holidays. It describes the tools and web technologies used in the planning, design and implementation of the application.

The objectives are achieved by users being able to:

- browse all upcoming holidays
- register a username and password in order to access online services
- book, change or cancel a holiday
- view the locations where they have been.

Additionally, an admin user can:

- add new holidays
- modify or remove existing holidays
- add new features that can be linked to holidays.

Before starting the development, the requirements were refined, sorted and split into smaller tasks, following the agile methodology. The available PHP frameworks were researched and assessed. During development, the web application was made more secure by finding vulnerabilities in the code and fixing them. Security principles were adopted to avoid introducing new vulnerabilities. The application was tested after each code update for multiple scenarios where it was expected to pass or fail.

1.1 Literature Review

During the initial planning phase for the application, other holiday websites were researched to discover common features that would be beneficial to include in the holiday bookings application.

1.1.1 Explora Expeditions

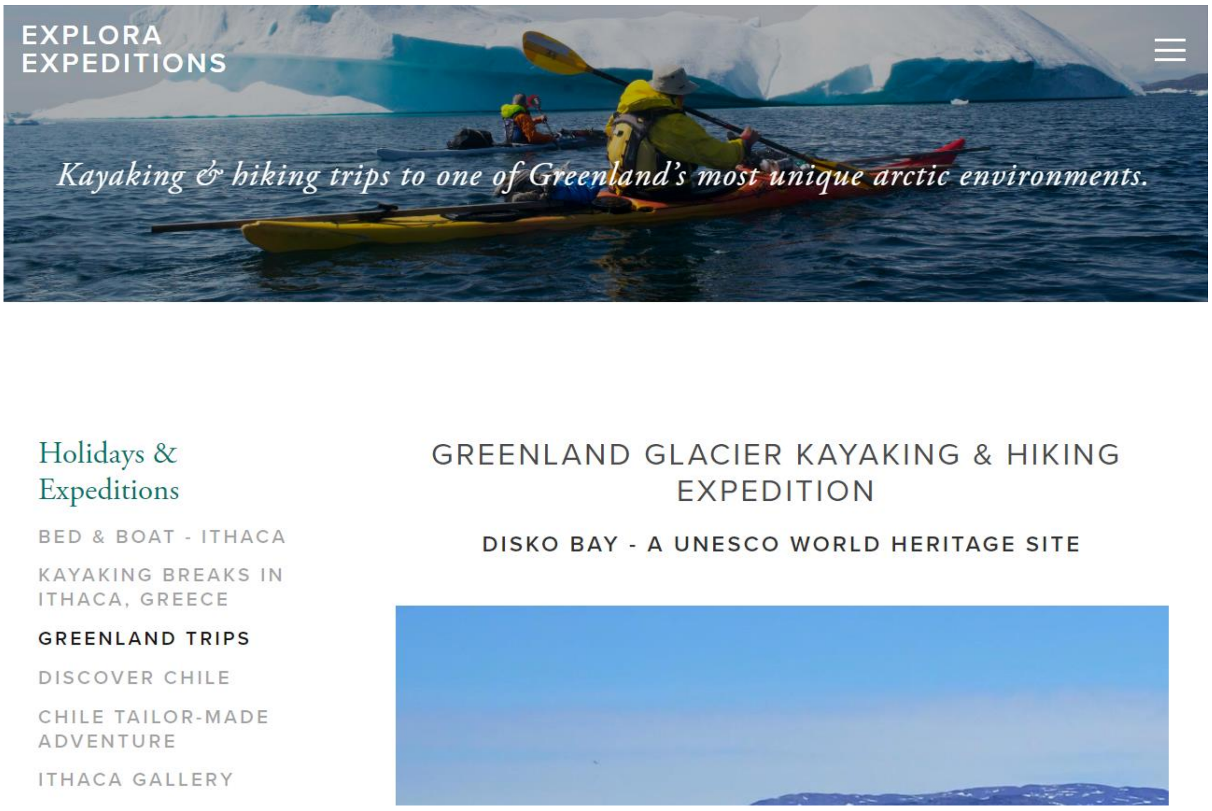

EXPLORA EXPEDITIONS
Kayaking & hiking trips to one of Greenland's most unique arctic environments.
Holidays & Expeditions
BED & BOAT - ITHACA
KAYAKING BREAKS IN ITHACA, GREECE
GREENLAND TRIPS
DISCOVER CHILE
CHILE TAILOR-MADE ADVENTURE
ITHACA GALLERY
GREENLAND GLACIER KAYAKING & HIKING EXPEDITION
DISKO BAY - A UNESCO WORLD HERITAGE SITE

Figure 1: Explora Expeditions holiday list

Explora Expeditions offer a select list of adventure holidays. Each holiday has a static HTML page. In order to book a holiday, the user is taken to the “contact” webpage where they can submit an enquiry. This submits an email to the site administrator who deals with each user on a one-to-one basis. The holiday can only be booked and paid for once this correspondence has taken place. There is no official booking and payment process on the website itself.

1.1.2 Thomas Cook

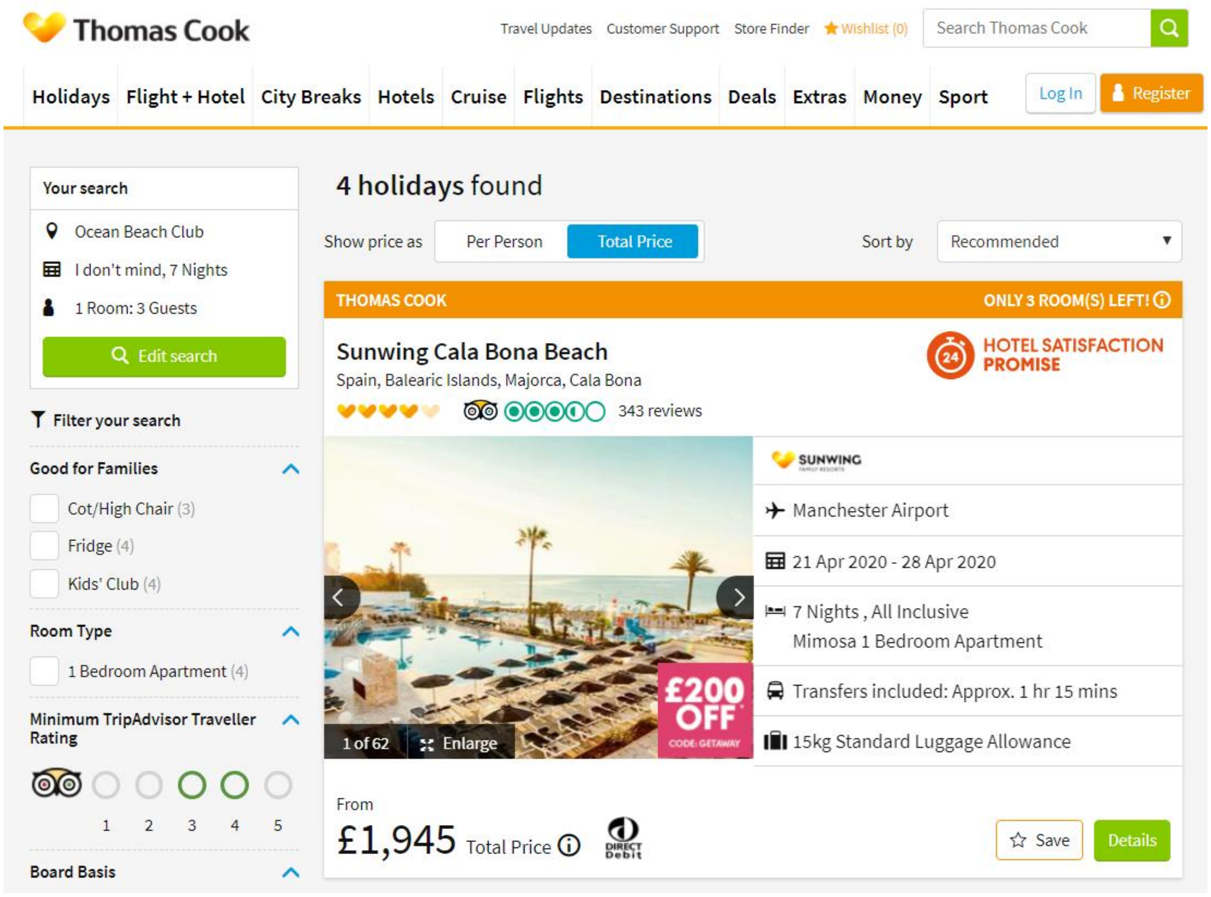
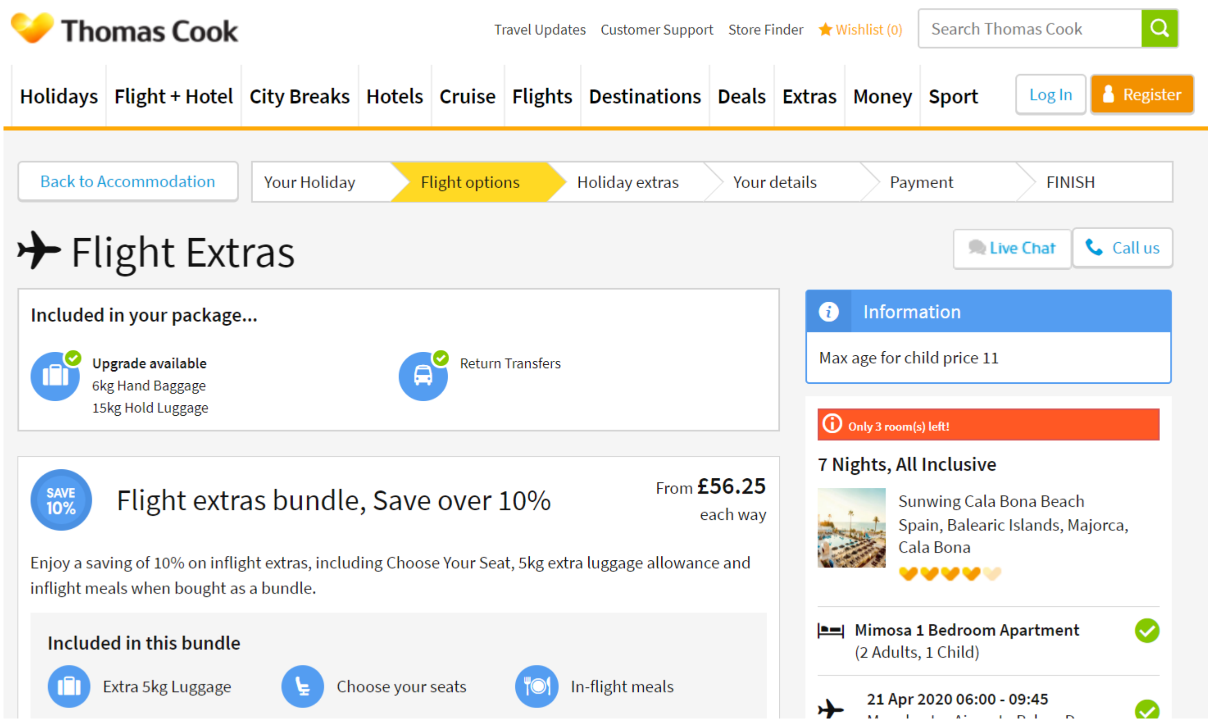


Figure 2: Thomas Cook holiday search results

Thomas Cook's website allows the user to browse holidays, flights, hotels etc with the option to filter search results according to the facilities on offer. When searching for holidays, the user is shown a dynamic list of holiday cards. Each card contains a snapshot of information and has "save" and "details" buttons. Clicking on the details button takes the user to a webpage for that holiday containing further information and an estimated price. The user can then choose from a range of accommodation and flights.


Thomas Cook

Travel Updates Customer Support Store Finder ★ Wishlist (0) Search Thomas Cook

Holidays | Flight + Hotel | City Breaks | Hotels | Cruise | Flights | Destinations | Deals | Extras | Money | Sport | Log In | Register

Back to Accommodation | Your Holiday | Flight options | Holiday extras | Your details | Payment | FINISH

Flight Extras

Live Chat | Call us

Included in your package...

Upgrade available
6kg Hand Baggage
15kg Hold Luggage

Return Transfers

SAVE 10%

Flight extras bundle, Save over 10%

From £56.25 each way

Enjoy a saving of 10% on inflight extras, including Choose Your Seat, 5kg extra luggage allowance and inflight meals when bought as a bundle.

Included in this bundle

Extra 5kg Luggage | Choose your seats | In-flight meals

Information

Max age for child price 11

Only 3 room(s) left!

7 Nights, All Inclusive

Sunwing Cala Bona Beach
Spain, Balearic Islands, Majorca, Cala Bona

Mimosa 1 Bedroom Apartment
(2 Adults, 1 Child)

21 Apr 2020 06:00 - 09:45

Figure 3: Thomas Cook booking flow

Once these are chosen, a final price is shown, which will change if the user picks flight extras. The user then navigates through a page where they can be upsold with holiday extras. The next page asks the user to enter personal details to create an account, or login to their existing account. This is followed by payment details and a confirmation page to finalise booking the holiday. During this stage, the user has the option to live chat or call an agent, which is the best communication platform for customer satisfaction and retention (Suthar, 2017).

1.1.3 Evaluation

Explora Expeditions has a simple website which is more informative than functional. This is an example of a traditional website. User interaction is limited to navigating links and entering data in forms. The thin-client architecture is simple and universal, but severely limits the quality of the application that could be delivered (Fraternali, 2010).

Thomas Cook has a complex web application with a wide range of offerings. The application is highly functional, with users having the ability to create a profile, browse and book holidays/ flights/ hotels and chat to an agent. It is a dynamic, scalable application. Holidays are displayed in a consistent format. This is an example of a Rich Internet Application (RIA) that can leverage the rendering and communication powers of the underlying scripting engine to request the data needed and update the user interface to display it. This permits significant enhancement of the user interface and facilitates improved application intuitiveness (Farrell, 2007).

The typography across the Thomas Cook site is consistent and easy to read, whereas Explora Expeditions uses a number of contrasting fonts. Sans serif typefaces are traditionally used for shorter passages of text and for text that is meant to be scanned quickly (Saffer, 2007), so the Thomas Cook site was found to be more user friendly.

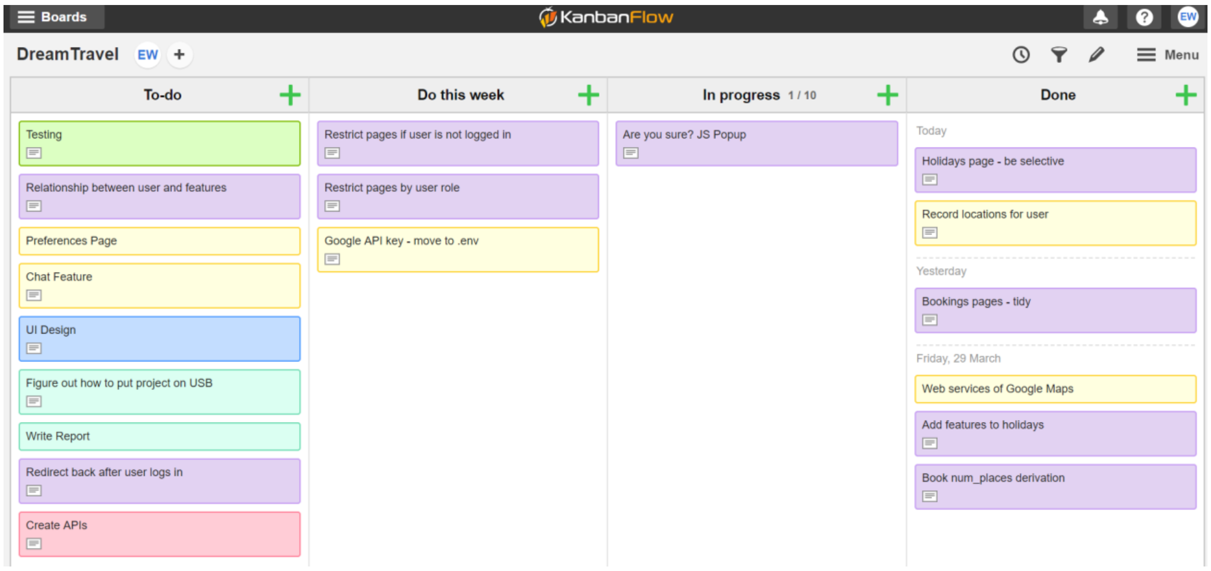
Both websites have responsive UI design.

The ability to see information about multiple holidays on a single page using Bootstrap cards for communicating quick stories (Adams, 2016) was a common feature of all the major holiday websites. This was incorporated into the holiday bookings UI design along with the top navigation bar across the top of the page as it is easy to find and click through to other pages.

2 Methodologies

2.1 Planning

The assignment was broken up so that each stage or piece of functionality was a separate task. KanbanFlow was used to record the tasks and status of each one.



DreamTravel EW

To-do
- Testing
- Relationship between user and features
- Preferences Page
- Chat Feature
- UI Design
- Figure out how to put project on USB
- Write Report
- Redirect back after user logs in
- Create APIs

Do this week
- Restrict pages if user is not logged in
- Restrict pages by user role
- Google API key - move to .env

In progress 1 / 10
- Are you sure? JS Popup

Done

Today
- Holidays page - be selective
- Record locations for user

Yesterday
- Bookings pages - tidy

Friday, 29 March
- Web services of Google Maps
- Add features to holidays
- Book num_places derivation

Figure 4: Kanban Flow task board

Git was used to version control the application. Code was pushed to a GitHub repository so that it was backed up on the cloud. Each time a new task was started, a branch was created for that task so that the code could be edited as much as needed. Once the new functionality was implemented on that branch, tests were run to make sure the existing functionality had not been affected. It was then compared with and merged to the master branch.

The decision was made to develop the application using a PHP framework. Web application frameworks are collections of libraries, aimed at promoting high code reusability, that can lead to faster development of web applications (Lancor, 2013). Web application frameworks contain code for common functionalities like database access, session management and security. They help reduce the development and testing overhead of web applications. They also support folder organisation and structure, thereby promoting easier code maintainability (Lancor, 2013).

Among the highly popular PHP frameworks, Laravel was chosen because of its claim to produce a development process that is agreeable for the developer without losing the application's functionality (Armel, 2014). It tries to ease the development process by simplifying repetitive tasks commonly used in today's web applications, including but not limited to routing, authentication, caching and sessions (Armel, 2014). The abundance of tutorial videos and user forums for Laravel further support new development.

3 Design

The Laravel framework uses the MVC (Model, View Controller) design pattern for development. MVC imposes a separation of behaviour between the actual model of the application domain, the views used for displaying the state of the model, and the editing or control of the model and views. It is a way to design and implement interactive application software that takes advantage of modularity. This supports the conceptual development of the application and allows pieces already developed for one application to be reused in a new application (Pope, 1998).

3.1 UI Design

The first stage of the UI Design was to consider the pages needed for each bit of functionality, and the user journey from one page to the next.

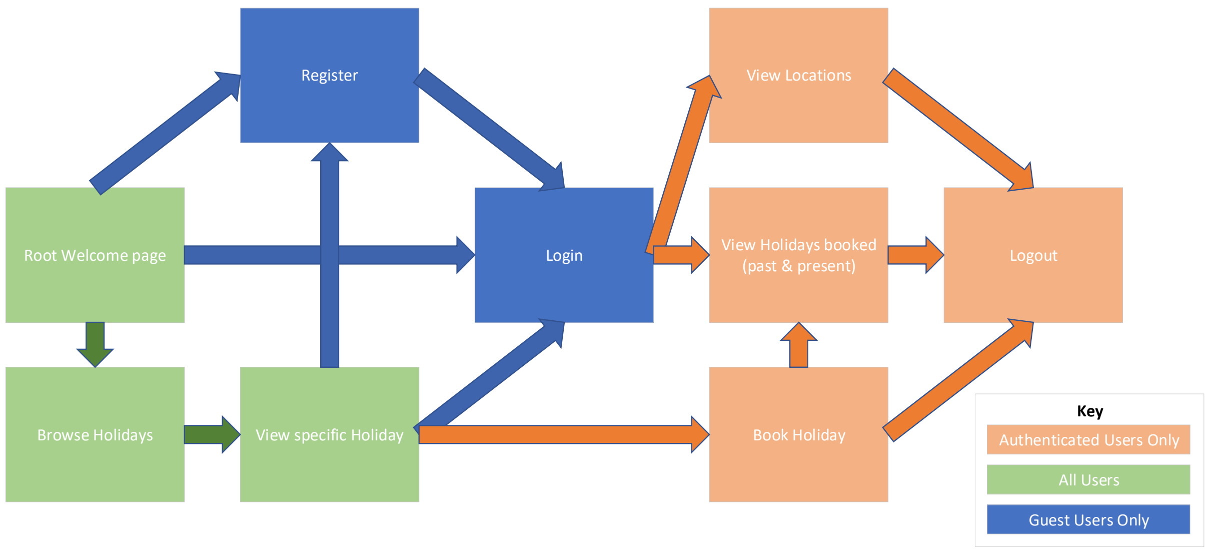


Figure 5: User Flow diagram

The front end should be easily navigable and follow the User Interface design guidelines of consistency, users feeling in control, usability, error prevention and minimalist design (Johnson, 2014).

Initially, each page was mocked up using draw.io to visualise them and keep the look and feel consistent throughout the application. There was a home page, holidays page, bookings page, preferences page, login page and register page.

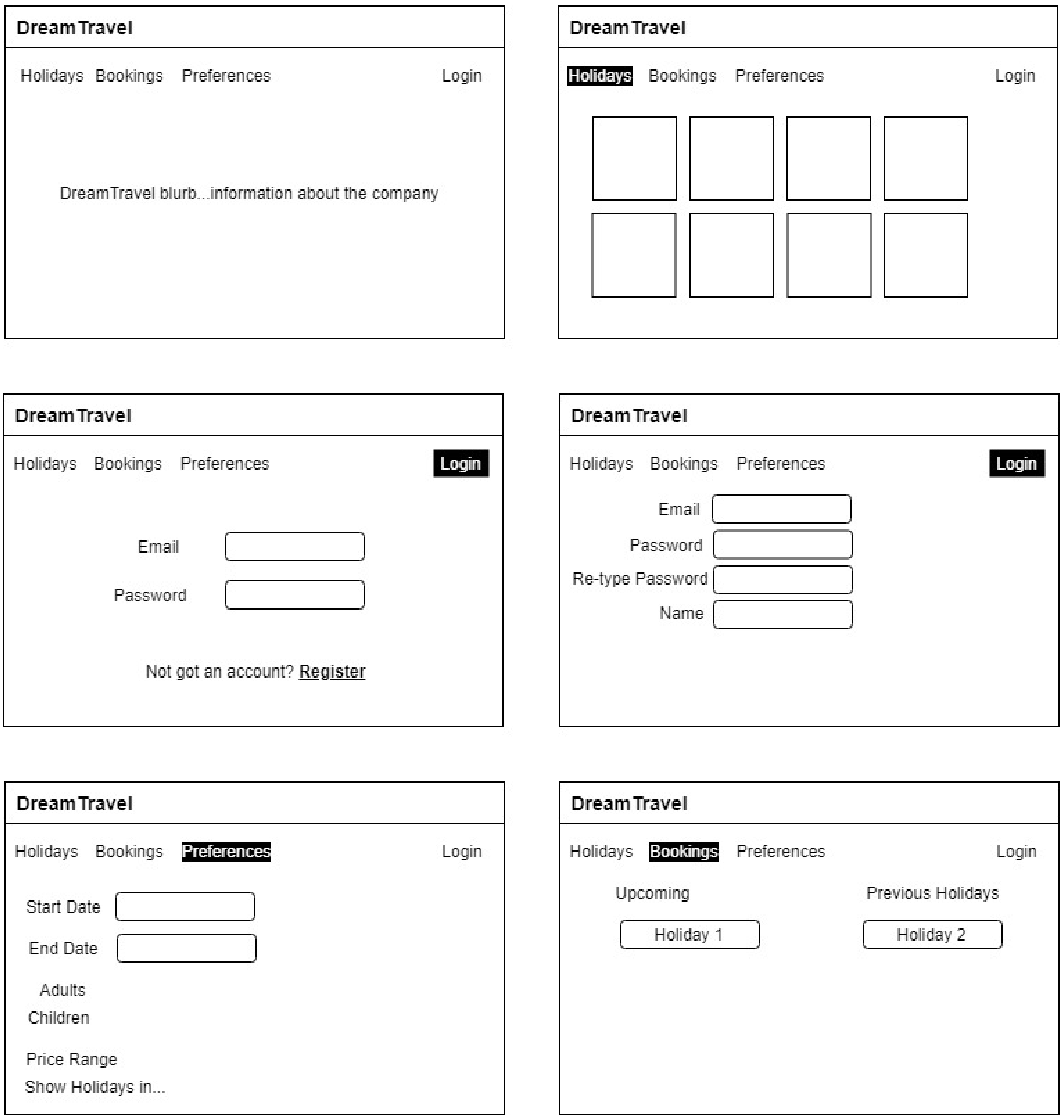


Figure 6: User Interface initial design

The initial design had a banner with the holiday company logo and a navigation bar across the top of every screen. Different options would be visible in the navigation bar depending upon whether the user was logged in, logged in as an admin, or not logged in. This would allow users to swap between application pages with ease.

The final application interface is broadly the same as the initial sketches, with a handful of modifications.

The navigation bar and company name/logo are incorporated into one. This reduces the amount of space taken up at the top of the page.

DreamTravel
Holidays
Login
Register

Figure 7: Navigation bar for guest access

The navigational buttons that lead to user specific pages (register/login/logout, my bookings, my locations) are placed on the right-hand side of the navbar, and those which are generic are placed on the left-hand side.

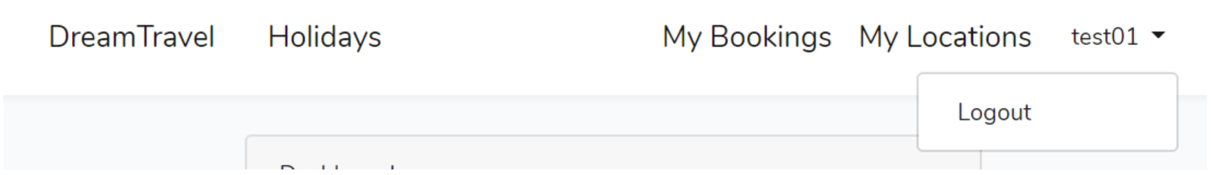


Figure 8: Navigation bar for a logged in user

The username appears instead of Login or Register when a user has logged in. Selecting this dropdown provides the user with the option to Logout. This could be extended to let the user change their account information or reset a password with more development work.

The default Laravel theme was used. This included the Bootstrap CSS framework which supports the popular web browsers and is good for pre-styled elements, responsive design and modularity (Jain, 2014).

The buttons were styled so that they were blue if clicking led to another page, green if clicking added or updated information, and red if clicking removed information.

3.2 Database Design

An object-oriented database was designed.

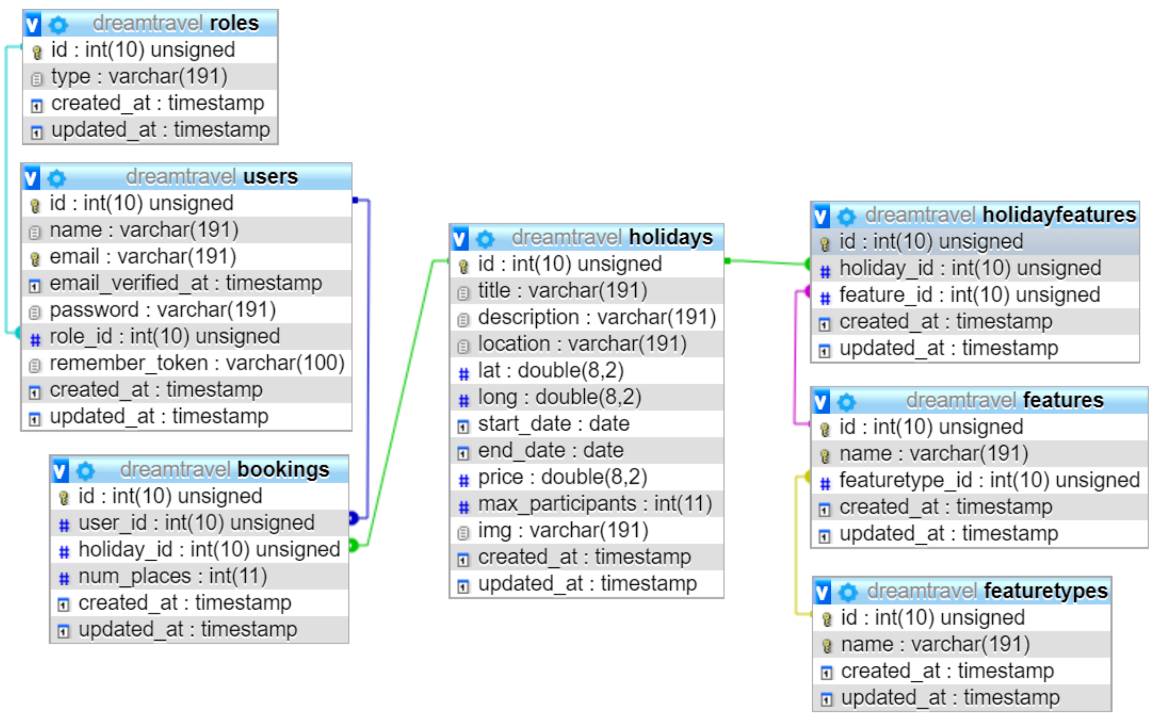


Figure 9: Database entity relationship diagram

Each table in the database has a corresponding Eloquent ORM model which is used to interact with that table. This means database queries can be written in PHP rather than writing SQL code.

The files containing the code for the Eloquent ORM models are:

- dreamtravel\app\booking.php
- dreamtravel\app\feature.php
- dreamtravel\app\featuretype.php
- dreamtravel\app\holiday.php
- dreamtravel\app\holidayfeature.php
- dreamtravel\app\Role.php
- dreamtravel\app\User.php

The Roles table is for storing role types. “user” and “admin” roles were created.

The Users table stores information about each user. By default, they are given the “user” role. There is a one-to-many relationship between the Roles and Users tables.

The FeatureTypes table stores categories for features. “Board Basis”, “Transport” and “Facilities” feature types were created.

The Features table stores features for each FeatureType. There is a one-to-many relationship between the FeatureTypes and Features tables. For example, the features “Full board”, “Half board” and “Self-catered” all have the feature type “Board Basis”.

The Holidays table stores holiday information such as location and price for each holiday.

The Bookings table links a holiday with a user and stores the number of places the user has booked for the holiday. A user can have many bookings, and a holiday can be booked by multiple users (many-to-many relationship).

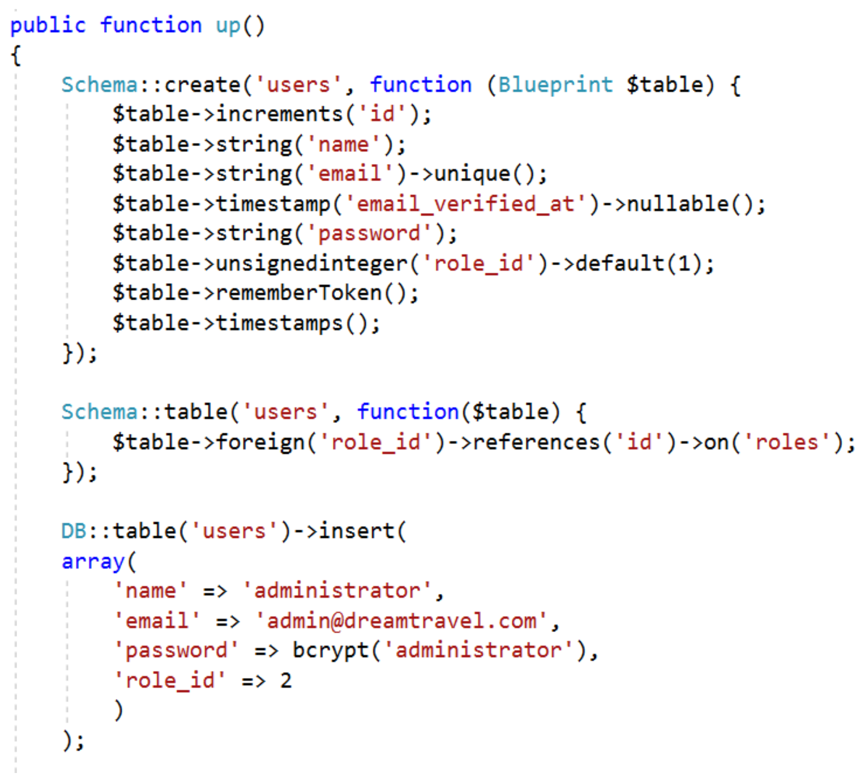
The HolidayFeatures table links features to a holiday. A holiday can have many features, and a feature can be applicable to multiple holidays (many-to-many relationship).

3.2.1 Database Migration and Population

The files containing the code for the database migration and population are found in the folder:

- dreamtravel\database\migrations

The database migration files contain the code to create each database table and any foreign keys. In addition, they also contain code to populate the database with a small amount of data including roles, users, feature types, features, holidays and holiday bookings. This means the database could be created and migrated from afresh if required, without needing to insert dummy data manually before the application would be usable.



```php
public function up()
{
    Schema::create('users', function (Blueprint $table) {
        $table->increments('id');
        $table->string('name');
        $table->string('email')->unique();
        $table->timestamp('email_verified_at')->nullable();
        $table->string('password');
        $table->unsignedinteger('role_id')->default(1);
        $table->rememberToken();
        $table->timestamps();
    });

    Schema::table('users', function($table) {
        $table->foreign('role_id')->references('id')->on('roles');
    });

    DB::table('users')->insert(
    array(
        'name' => 'administrator',
        'email' => 'admin@dreamtravel.com',
        'password' => bcrypt('administrator'),
        'role_id' => 2
        )
    );
```

Figure 10: Code from the Users table migration file

3.3 Software Design

The functional requirements are identified by the use case diagram.

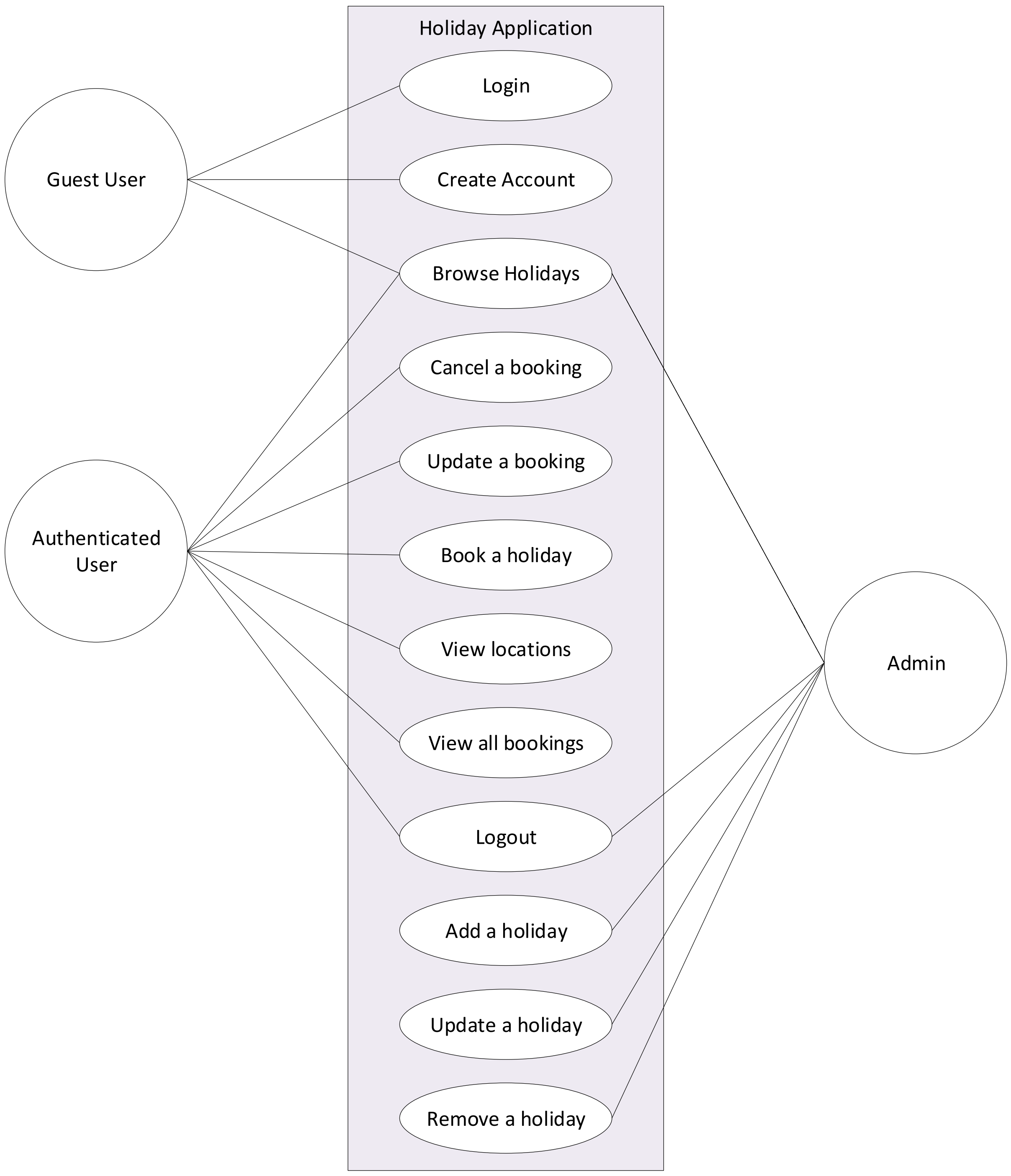


Figure 11: Use Case diagram

A guest user can browse all the available holidays, create an account and login. In addition, an authenticated user can make a booking, update or cancel the booking, view all their bookings, view the locations where they have been and logout.

An admin user can add new holidays, update existing holidays and remove holidays from the system.

Controllers were used to define the user actions.

4 Implementation

In addition to Laravel being used for software development, JavaScript and jQuery were used where dynamic updates were needed in the User Interface. An XAMPP server was used to host a MariaDB instance.

Laravel comes with the Blade template language, which has been used to create reusable and maintainable layouts and template components (Bean, 2015).

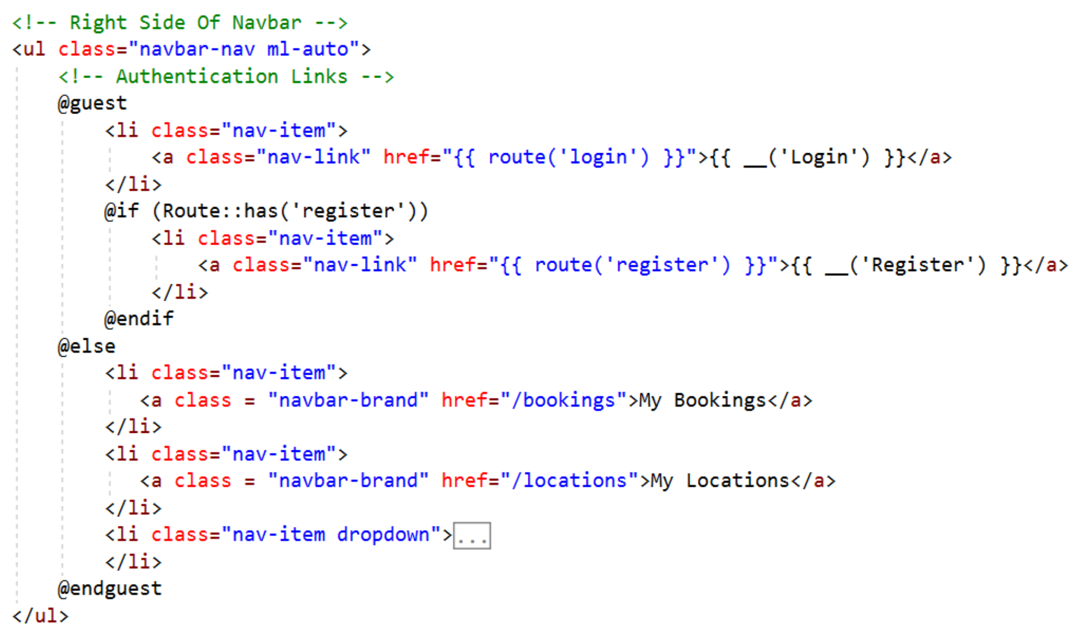
This section details the functionality of each page of the application.

4.1 Layout

The file containing the code for the layout is:

- dreamtravel\resources\views\layouts\app.blade.php

Each page of the application has the same navigation bar across the top of the page. This is achieved by having an application layout file containing the code for navigation as well as scripts for using Bootstrap and jQuery. The other pages all extend the layout file.



```
<!-- Right Side Of Navbar -->
<ul class="navbar-nav ml-auto">
    <!-- Authentication Links -->
    @guest
        <li class="nav-item">
            <a class="nav-link" href="{{ route('login') }}">{{ __('Login') }}</a>
        </li>
        @if (Route::has('register'))
            <li class="nav-item">
                <a class="nav-link" href="{{ route('register') }}">{{ __('Register') }}</a>
            </li>
        @endif
    @else
        <li class="nav-item">
           <a class = "navbar-brand" href="/bookings">My Bookings</a>
        </li>
        <li class="nav-item">
           <a class = "navbar-brand" href="/locations">My Locations</a>
        </li>
        <li class="nav-item dropdown">...
        </li>
    @endguest
</ul>
```

Figure 12: Layout file - navigation options

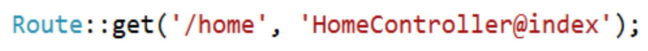
The underlying code for the navigation bar allows guest users to see the Login and Register options and shows authenticated users the options for My Bookings and My Locations. It also hides the Features option from all but admin users.

4.2 Routes

The file containing the code for the routes is:

- dreamtravel\routes\web.php

The routes file registers web routes for the application. Application requests are routed to the appropriate controller.



```
Route::get('/home', 'HomeController@index');
```

Figure 13: Web route

For example, whenever a user routes to /home, the index function in the HomeController will be run.

4.3 Welcome Page

The file containing code for the welcome page is:

- dreamtravel\resources\views\welcome.blade.php

The Welcome page is visible to all users. It provides the user with information about DreamTravel and its core offering. This is a simple HTML page with static content.

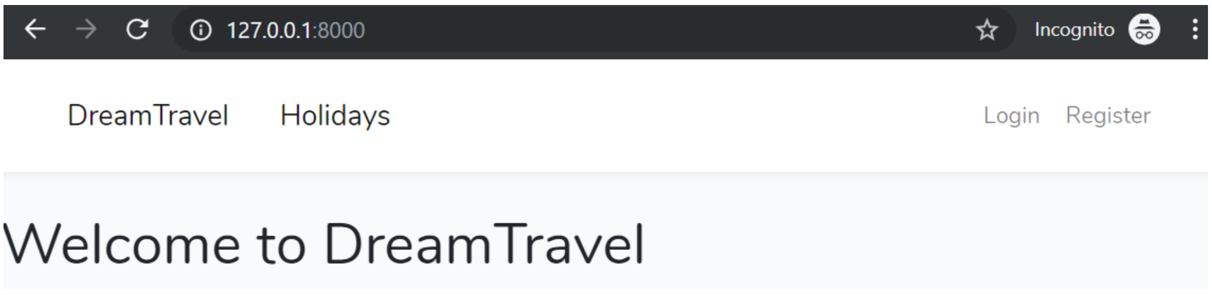


Figure 14: Welcome Page

4.4 Login and Register Pages

The folders containing files with the code for the Login and Register pages are:

- dreamtravel\resources\views\auth
- dreamtravel\app\Http\Controllers\Auth

The default Laravel Login and Register pages have been used for the application.

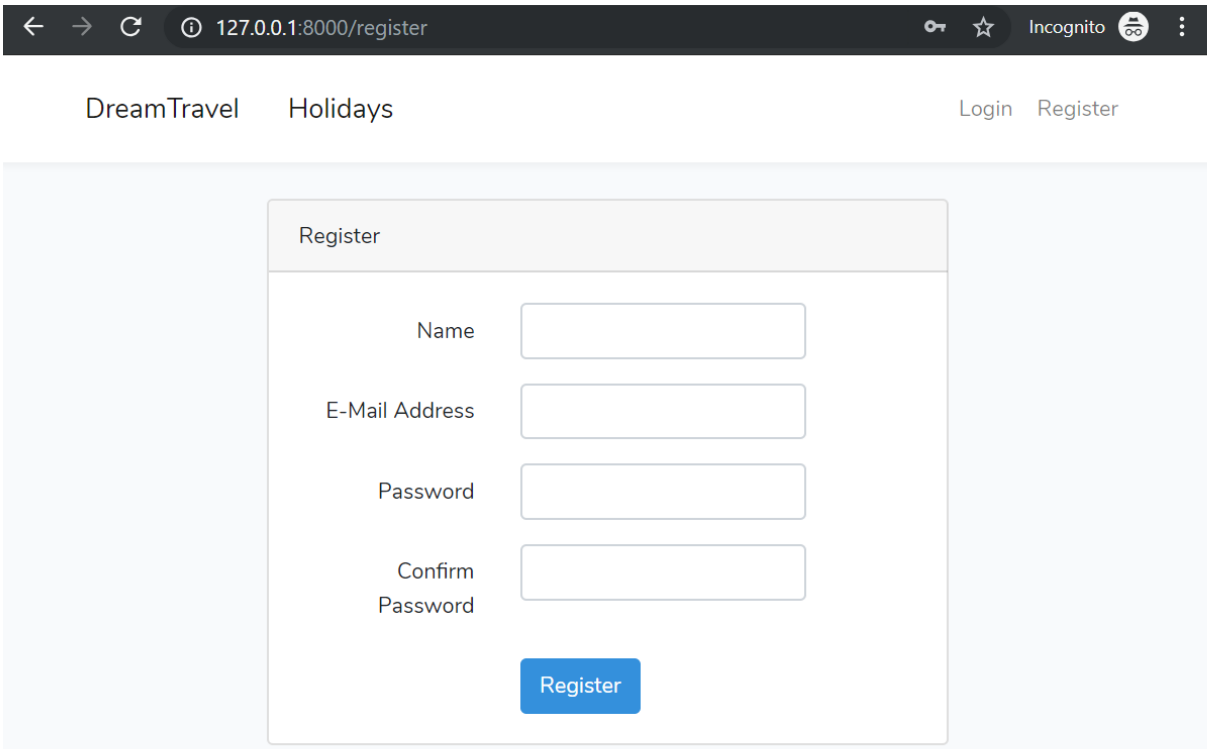
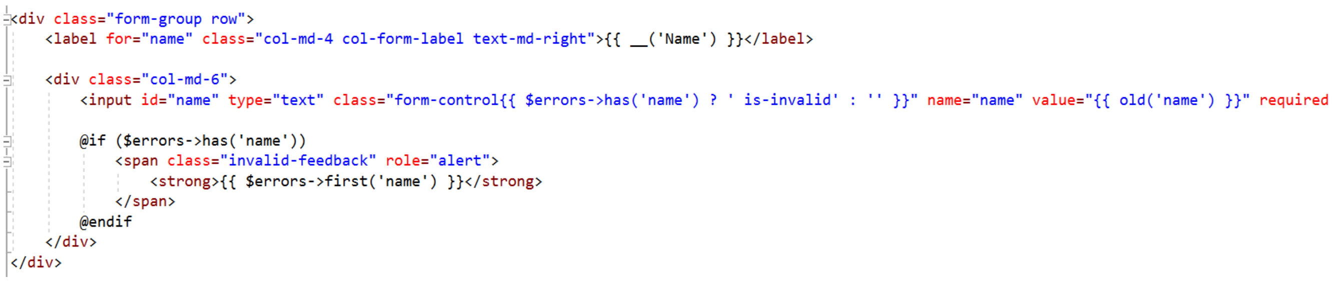


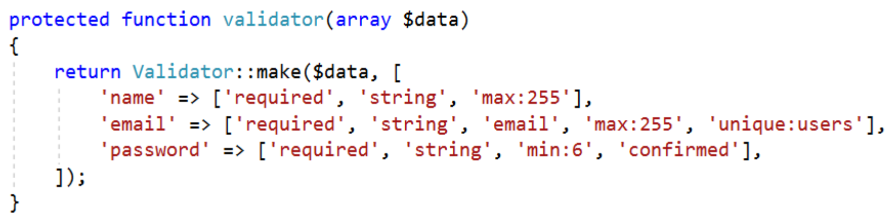
Figure 15: Register page

The register HTML form captures information about the user. Each field must be filled in before the form can be submitted. Validations are run on the fields on both the client and server to check for a unique email and that the passwords match.



```
<div class="form-group row">
    <label for="name" class="col-md-4 col-form-label text-md-right">{{ __('Name') }}</label>

    <div class="col-md-6">
        <input id="name" type="text" class="form-control{{ $errors->has('name') ? ' is-invalid' : '' }}" name="name" value="{{ old('name') }}" required

        @if ($errors->has('name'))
            <span class="invalid-feedback" role="alert">
                <strong>{{ $errors->first('name') }}</strong>
            </span>
        @endif
    </div>
</div>
```

Figure 16: Register - client side validates field is filled out and checks for errors



```
protected function validator(array $data)
{
    return Validator::make($data, [
        'name' => ['required', 'string', 'max:255'],
        'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
        'password' => ['required', 'string', 'min:6', 'confirmed'],
    ]);
}
```

Figure 17: Register - server validation for each field

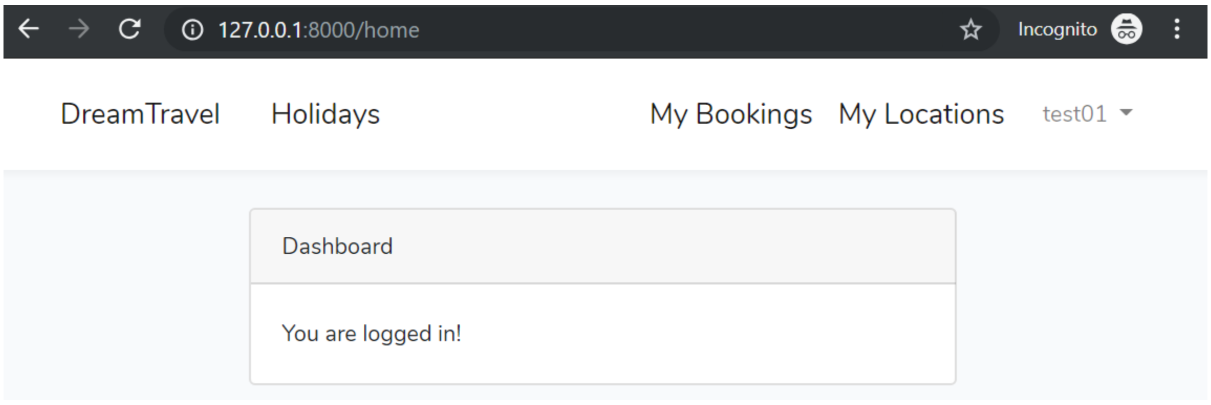
The password is hashed using the Bcrypt function, which is a cryptographic resource intensive algorithm that's almost impossible to crack (Ertaul, n.d.), before it is stored in the database. After registering or logging in, Laravel uses token authentication so the user is then able to access the application as an authenticated user instead of a guest.

4.5 Home Page

The file containing the code for the home page is:

- dreamtravel\resources\views\home.blade.php

The home page is another default Laravel page which is displayed after the user has logged in or registered. If an authenticated user tries to access the login or register pages, they are redirected to the home page.



Dashboard

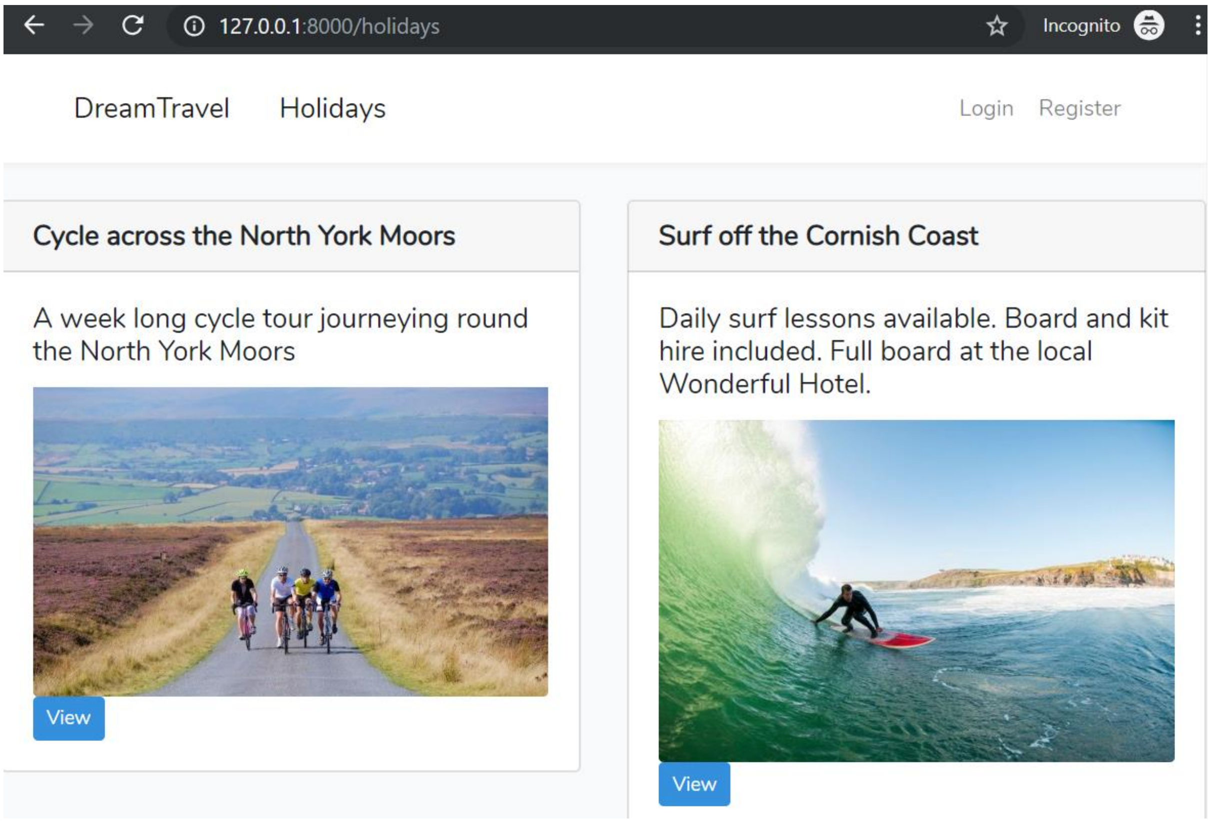
You are logged in!

Figure 18: Home page

4.6 Holidays Page

The files containing the code for the holidays page are:

- dreamtravel\resources\views\holidays\index.blade.php
- dreamtravel\app\Http\Controllers\HolidaysController.php



Cycle across the North York Moors

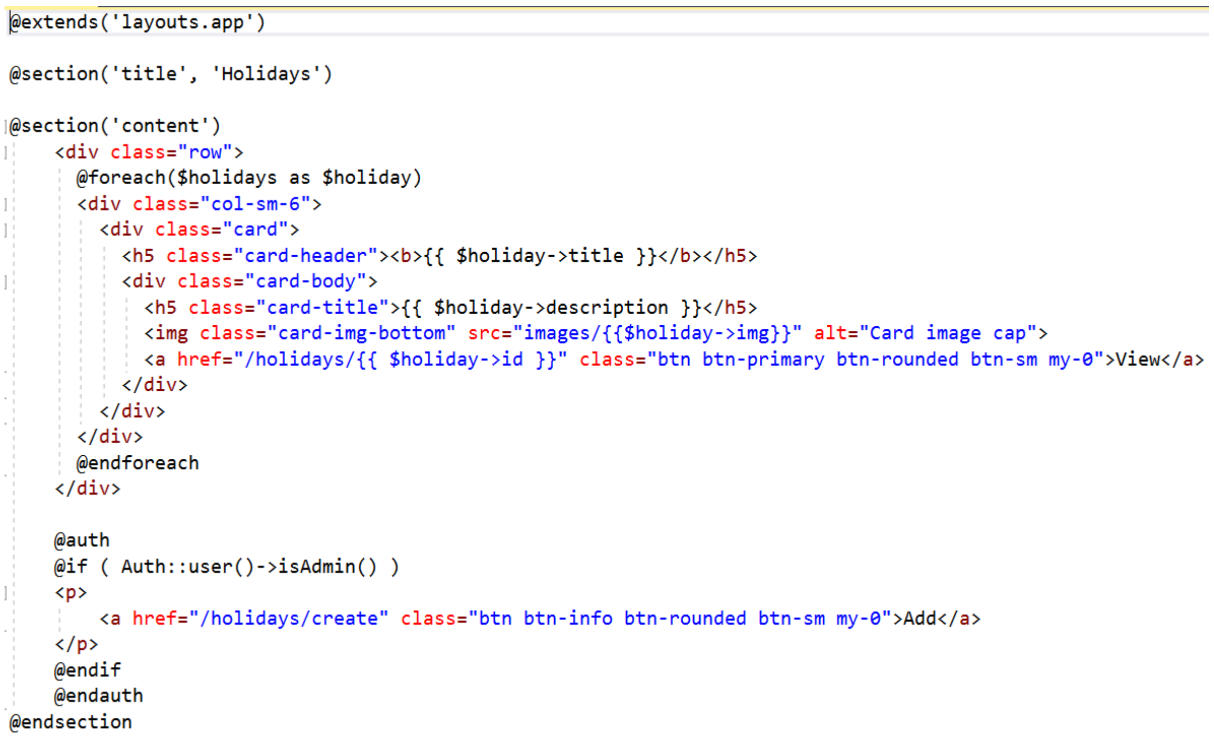
A week long cycle tour journeying round the North York Moors

Surf off the Cornish Coast

Daily surf lessons available. Board and kit hire included. Full board at the local Wonderful Hotel.

Figure 19: Holidays User Interface, Pictures from (Anon., 2019) (Myerscough, 2019) (Farm, 2019) (Anon., 2019)

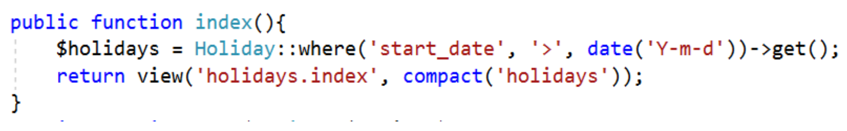
The holidays page is visible to all users. Snippets of information for each holiday are displayed using Bootstrap cards so that a user can directly compare multiple offerings. When the View button is clicked, they are directed to a page for that holiday which shows more detailed information.



```
@extends('layouts.app')

@section('title', 'Holidays')

@section('content')
    <div class="row">
      @foreach($holidays as $holiday)
      <div class="col-sm-6">
        <div class="card">
          <h5 class="card-header"><b>{{ $holiday->title }}</b></h5>
          <div class="card-body">
            <h5 class="card-title">{{ $holiday->description }}</h5>
            <img class="card-img-bottom" src="images/{{$holiday->img}}" alt="Card image cap">
            <a href="/holidays/{{ $holiday->id }}" class="btn btn-primary btn-rounded btn-sm my-0">View</a>
          </div>
        </div>
      </div>
      @endforeach
    </div>

    @auth
    @if ( Auth::user()->isAdmin() )
    <p>
        <a href="/holidays/create" class="btn btn-info btn-rounded btn-sm my-0">Add</a>
    </p>
    @endif
    @endauth
@endsection
```

Figure 20: Holidays view

The controller passes all holidays which have a start date later than the current date to the view for it to display.



```
public function index(){
    $holidays = Holiday::where('start_date', '>', date('Y-m-d'))->get();
    return view('holidays.index', compact('holidays'));
}
```

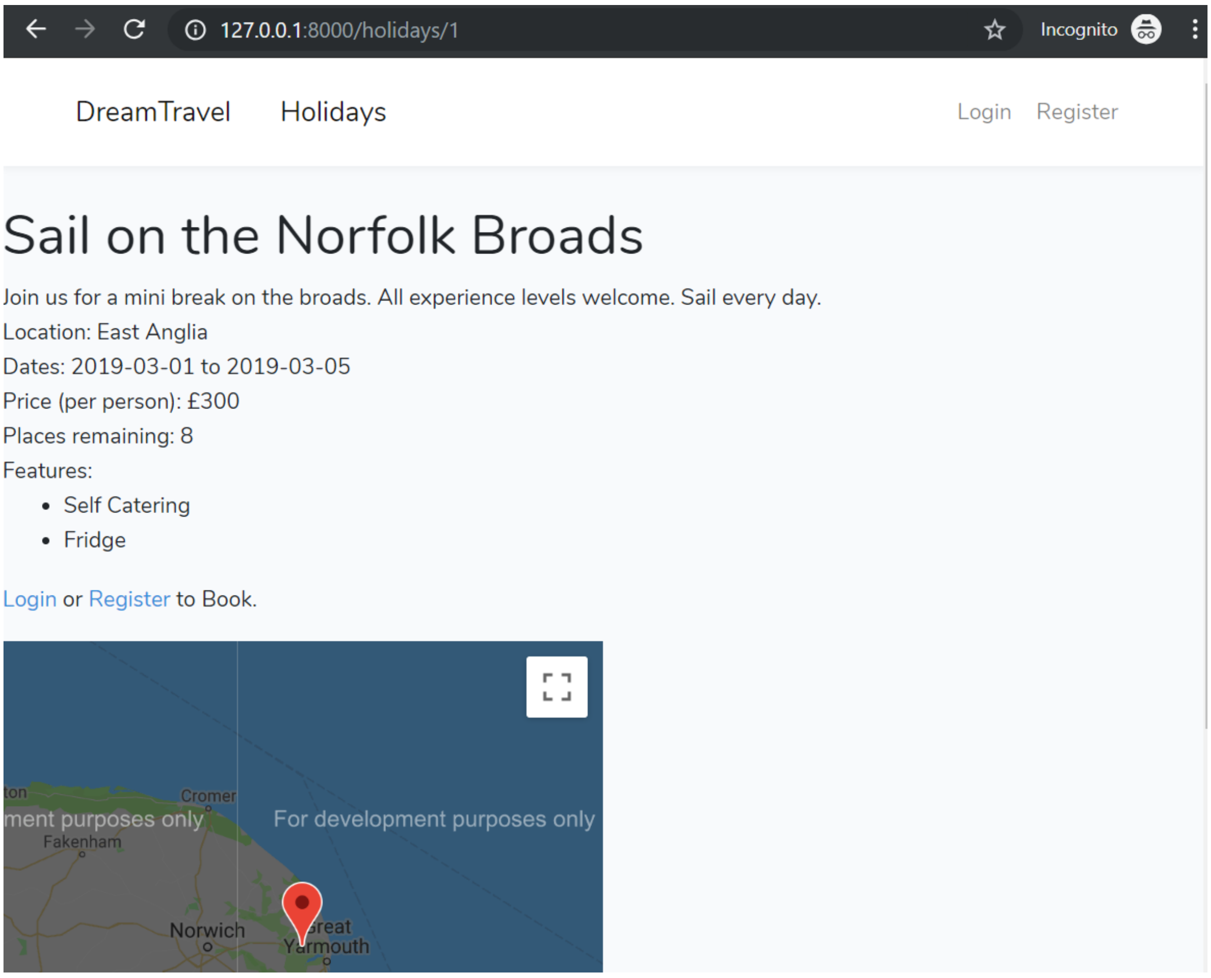
Figure 21: Holidays controller index function

4.7 Holiday View

The files containing the code for the holiday view are:

- dreamtravel\resources\views\holidays\show.blade.php
- dreamtravel\app\Http\Controllers\HolidaysController.php

The holiday details can be viewed by any user. The map showing the location of the holiday uses the Google Maps API, which is discussed in the My Locations section.


127.0.0.1:8000/holidays/1

DreamTravel Holidays Login Register

Sail on the Norfolk Broads

Join us for a mini break on the broads. All experience levels welcome. Sail every day.
Location: East Anglia
Dates: 2019-03-01 to 2019-03-05
Price (per person): £300
Places remaining: 8
Features:

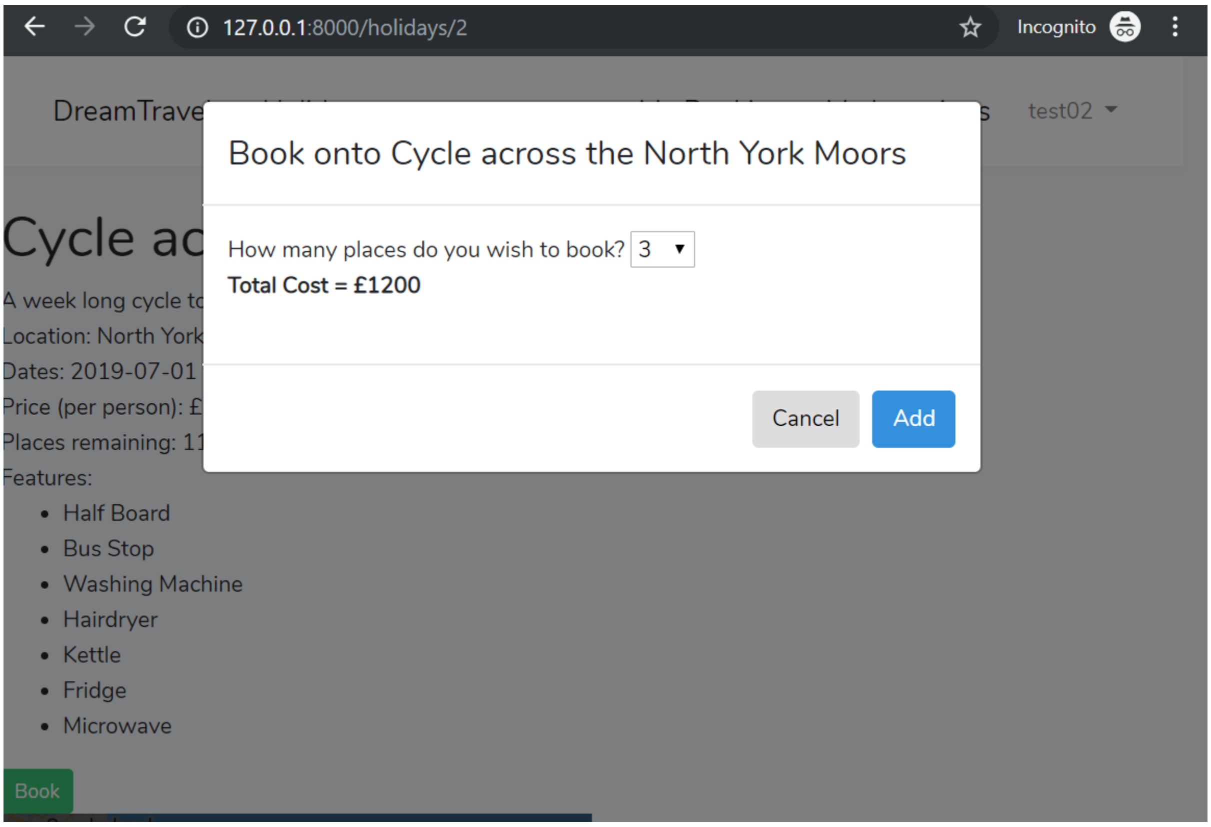
- Self Catering
- Fridge

Login or Register to Book.

Figure 22: View holiday details

A guest user will see "Login or Register" above the map. An authenticated user does not see these hyperlinks but will see a "Book" button if the start date is after the current date and there are places left on the holiday.

When an authenticated user clicks the book button a modal popup appears asking how many places they wish to book on the holiday.



Book onto Cycle across the North York Moors

How many places do you wish to book? 3

Total Cost = £1200

Figure 23: Book holiday popup

The Total Cost price is updated using JavaScript when the number of places in the drop down is changed.

```
<script>
    function costCalculate() {
        var x = document.getElementById("num_places").value;
        document.getElementById("demo").innerHTML = x * {{ $holiday->price }};
    }
</script>
```

Figure 24: JavaScript to dynamically update Total Cost as the number of places is changed

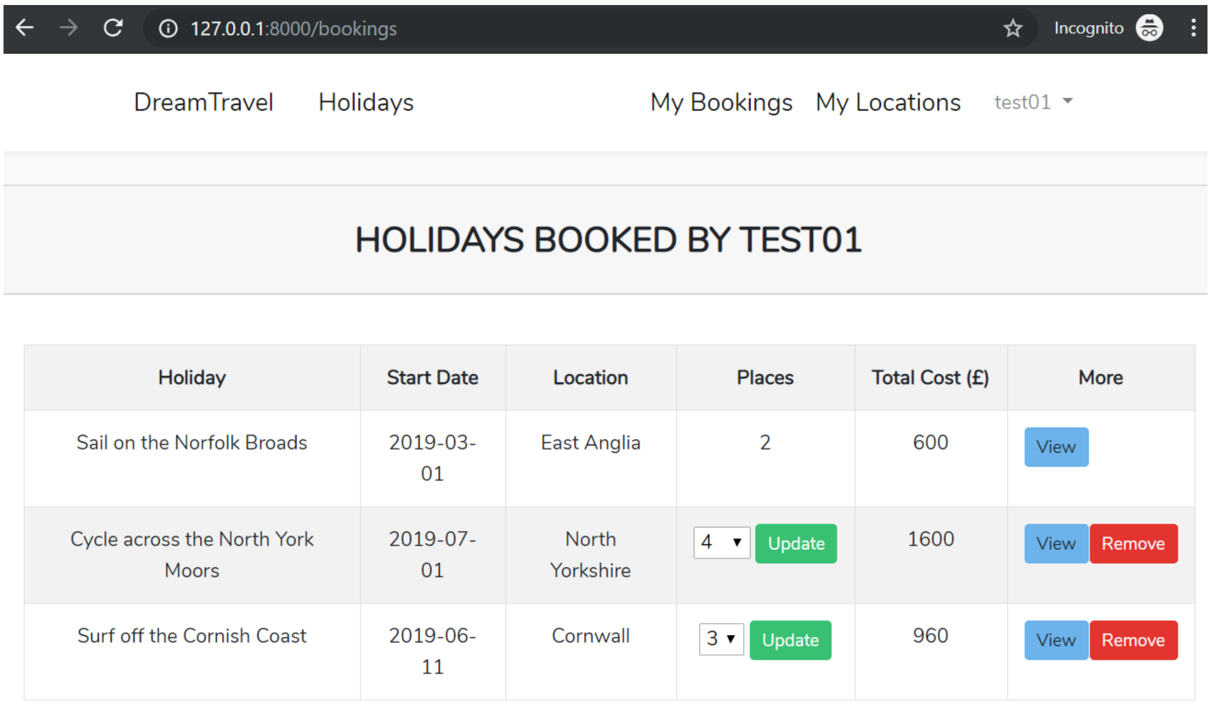
If the user clicks Cancel then the modal popup closes and they are returned to the holiday details page. If the user clicks Add then the Bookings Controller Store function adds the booking to the HolidayBookings database table. The user is then directed to the My Bookings page.

4.8 My Bookings Page

The files containing the code for the bookings are:

- dreamtravel\resources\views\bookings\index.php
- dreamtravel\app\Http\Controllers\HolidayBookingsController.php

The Bookings page contains an HTML table which displays all the holidays which the authenticated user has booked in the application. Clicking on the View button will take the user back to the Holiday information page. If the start date of the holiday is prior to the current date, then the user cannot modify the booking. Otherwise, the user has the options to update the number of places they have booked or remove the booking entirely.



HOLIDAYS BOOKED BY TEST01

Holiday	Start Date	Location	Places	Total Cost (£)	More
Sail on the Norfolk Broads	2019-03-01	East Anglia	2	600	View
Cycle across the North York Moors	2019-07-01	North Yorkshire	4 Update	1600	View Remove
Surf off the Cornish Coast	2019-06-11	Cornwall	3 Update	960	View Remove

Figure 25: User bookings page

If the user changes the number of places in the drop down, a jQuery script is used to show the new price.

```
<script>
   function getval(sel, price, id)
    {
        var textprice = "For " + sel.value + " places, the cost will be £" + sel.value * price;
        $("newprice"+id).html(textprice);
    }
</script>
```

Figure 26: jQuery script to show the new price

Clicking the Update button confirms the new number of places and stores it in the database. The page refreshes to show the current information.

4.9 My Locations Page

The files containing the code for the locations are:

- dreamtravel\resources\views\bookings\locations.php

- dreamtravel\app\Http\Controllers\HolidayBookingsController.php

The Locations page uses the Google Maps API to display to the user the locations of all the holidays that they have booked.

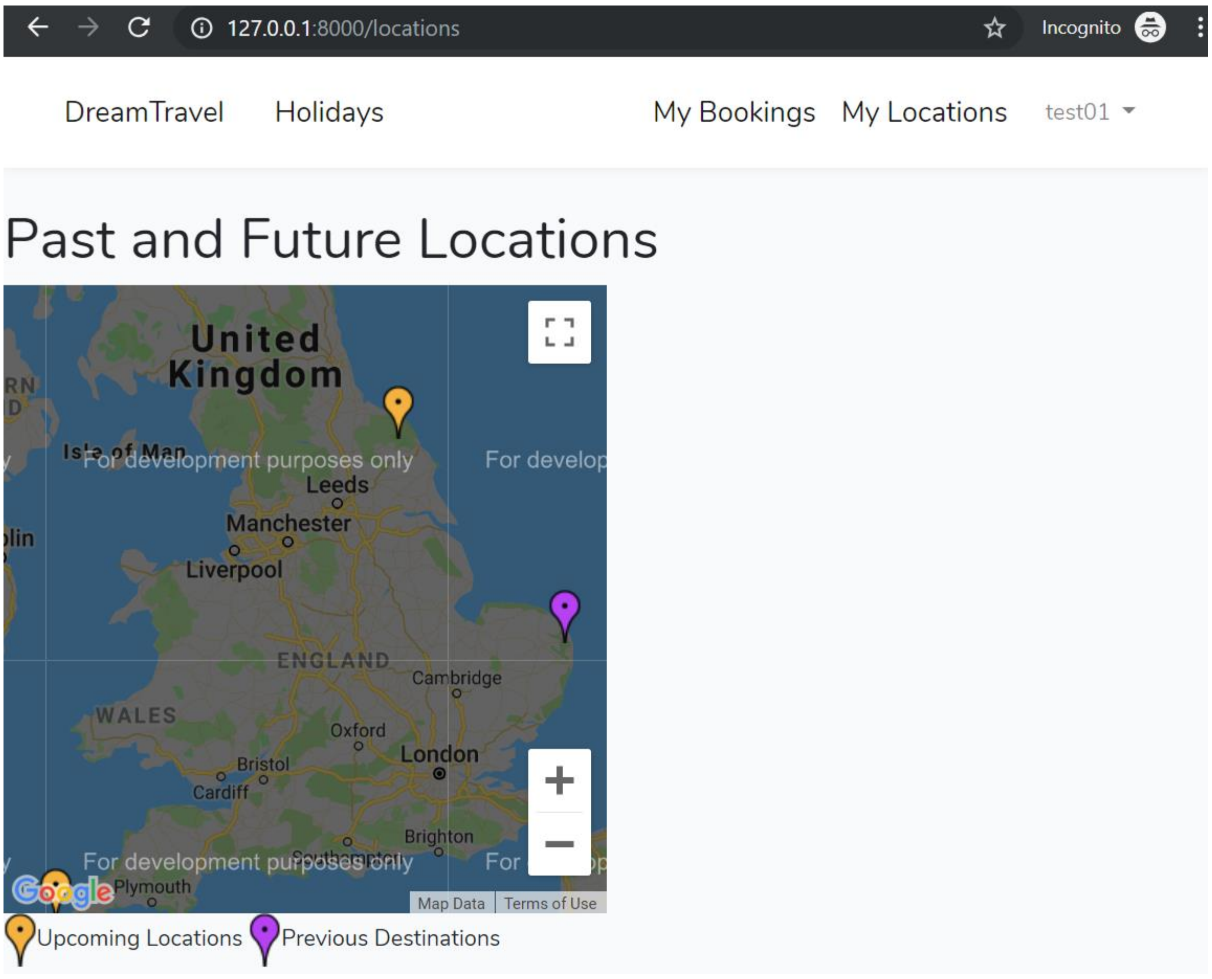
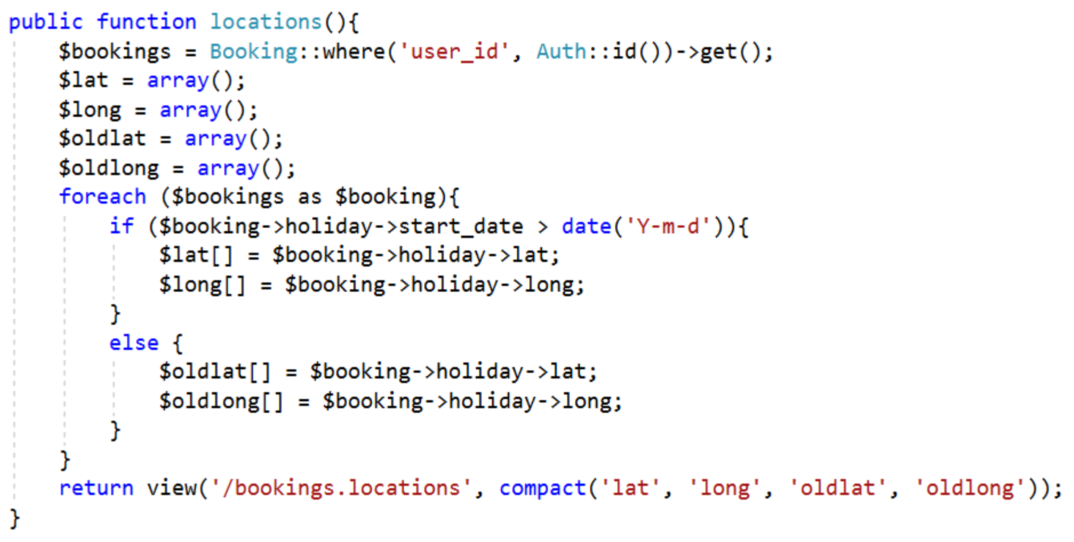


Figure 27: User locations page

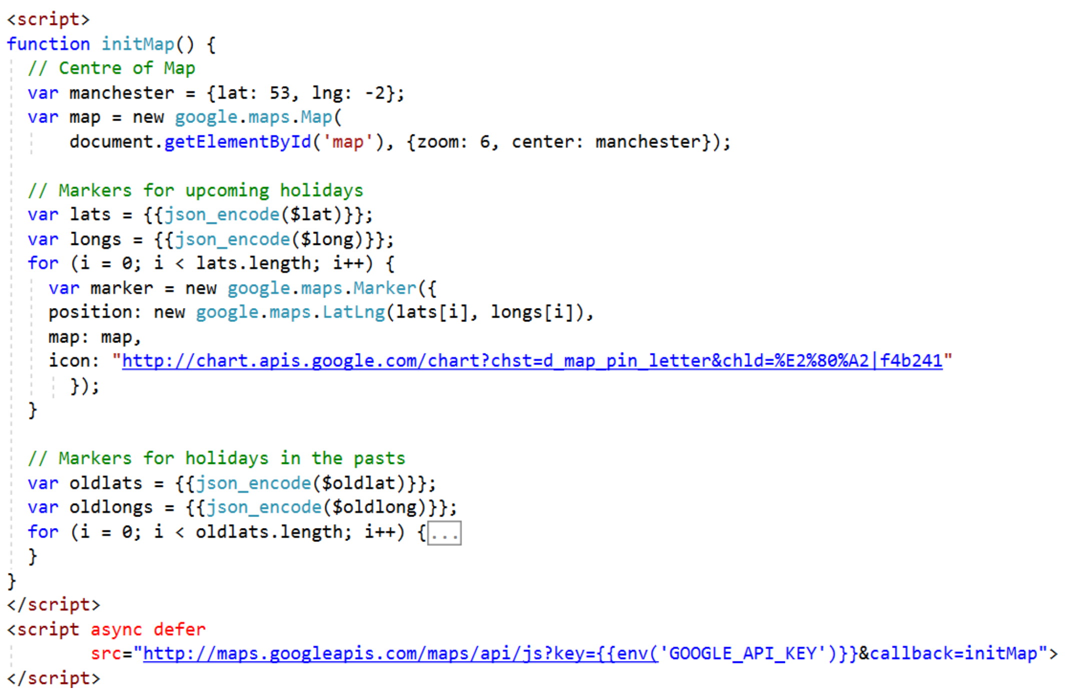
The HolidayBookingsController locations function returns arrays with the co-ordinates of holidays which have start dates before and those which have start dates after the current date.



```
public function locations(){
    $bookings = Booking::where('user_id', Auth::id())->get();
    $lat = array();
    $long = array();
    $oldlat = array();
    $oldlong = array();
    foreach ($bookings as $booking){
        if ($booking->holiday->start_date > date('Y-m-d')){
            $lat[] = $booking->holiday->lat;
            $long[] = $booking->holiday->long;
        }
        else {
            $oldlat[] = $booking->holiday->lat;
            $oldlong[] = $booking->holiday->long;
        }
    }
    return view('/bookings.locations', compact('lat', 'long', 'oldlat', 'oldlong'));
}
```

Figure 28: HolidayBookingsController locations function

The script in the view takes the arrays and uses different pin colours for the old and upcoming holidays. The map is centred on Manchester. The Google API Key is stored in the .env file.



```
<script>
function initMap() {
  // Centre of Map
  var manchester = {lat: 53, lng: -2};
  var map = new google.maps.Map(
      document.getElementById('map'), {zoom: 6, center: manchester});

  // Markers for upcoming holidays
  var lats = {{json_encode($lat)}};
  var longs = {{json_encode($long)}};
  for (i = 0; i < lats.length; i++) {
    var marker = new google.maps.Marker({
    position: new google.maps.LatLng(lats[i], longs[i]),
    map: map,
    icon: "http://chart.apis.google.com/chart?chst=d_map_pin_letter&chld=%E2%80%A2|f4b241"
      });
  }

  // Markers for holidays in the pasts
  var oldlats = {{json_encode($oldlat)}};
  var oldlongs = {{json_encode($oldlong)}};
  for (i = 0; i < oldlats.length; i++) {...
  }
}
</script>
<script async defer
        src="http://maps.googleapis.com/maps/api/js?key={{env('GOOGLE_API_KEY')}}&callback=initMap">
</script>
```

Figure 29: Google Maps API code

4.10 Admin Pages

The admin pages are additional functionality beyond the coursework brief. They were added to experiment with different access levels according to role type and ease the process of adding holidays. They allow an admin user to add, update and remove feature types and features, create holidays, update existing holidays and remove holidays. The styling has not been applied to these pages due to time constraints.

The admin pages could be improved and extended in several ways, such as:

- having an image upload field for use when creating a new holiday
- only allowing an admin to remove a holiday if all the bookings for that holiday are cancelled
- requiring the admin to send a notification to users that have booked a holiday if they modify it
- the ability to see, modify and remove all bookings and users in the system through the application
- the ability to elevate another user to the admin role.

The code for the admin pages is found in:

- dreamtravel\app\Http\Controllers\FeaturesController.php
- dreamtravel\app\Http\Controllers\FeatureTypesController.php
- dreamtravel\app\Http\Controllers\HolidaysController.php
- dreamtravel\resources\views\featuretypes\create.php
- dreamtravel\resources\views\featuretypes\edit.php
- dreamtravel\resources\views\featuretypes\index.php
- dreamtravel\resources\views\featuretypes\show.php
- dreamtravel\resources\views\holidays\create.php
- dreamtravel\resources\views\holidays\edit.php

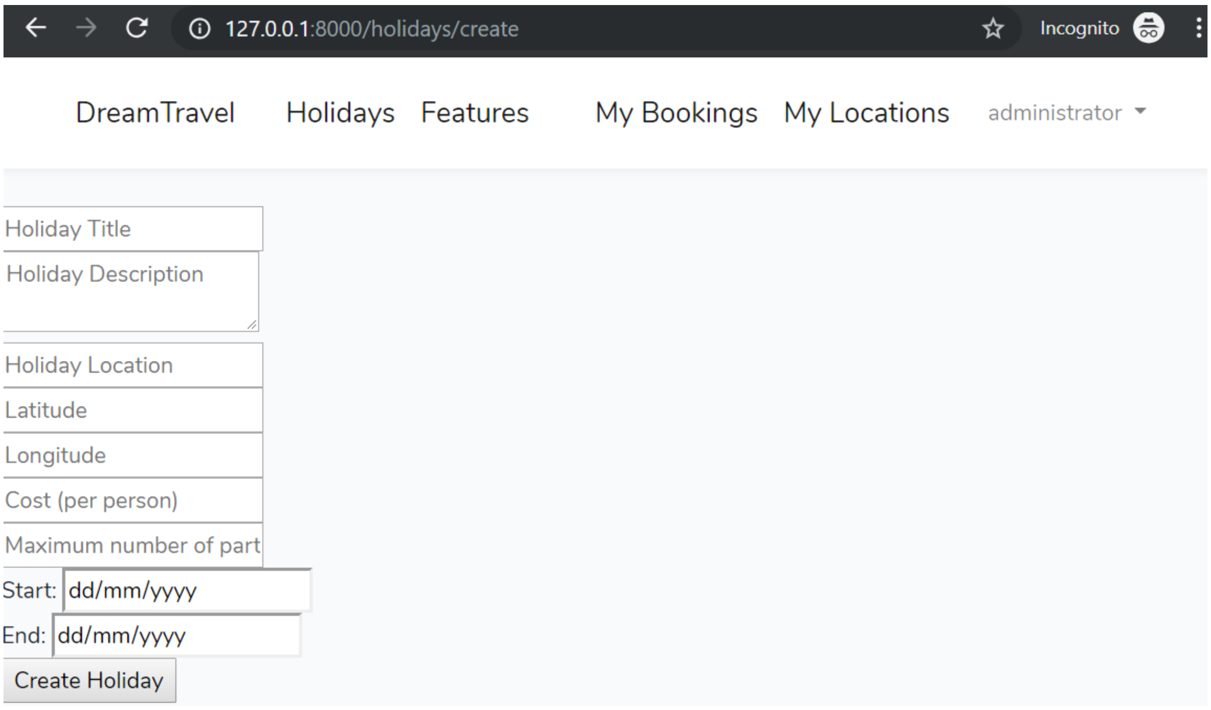


Figure 30: Admin only - add a holiday

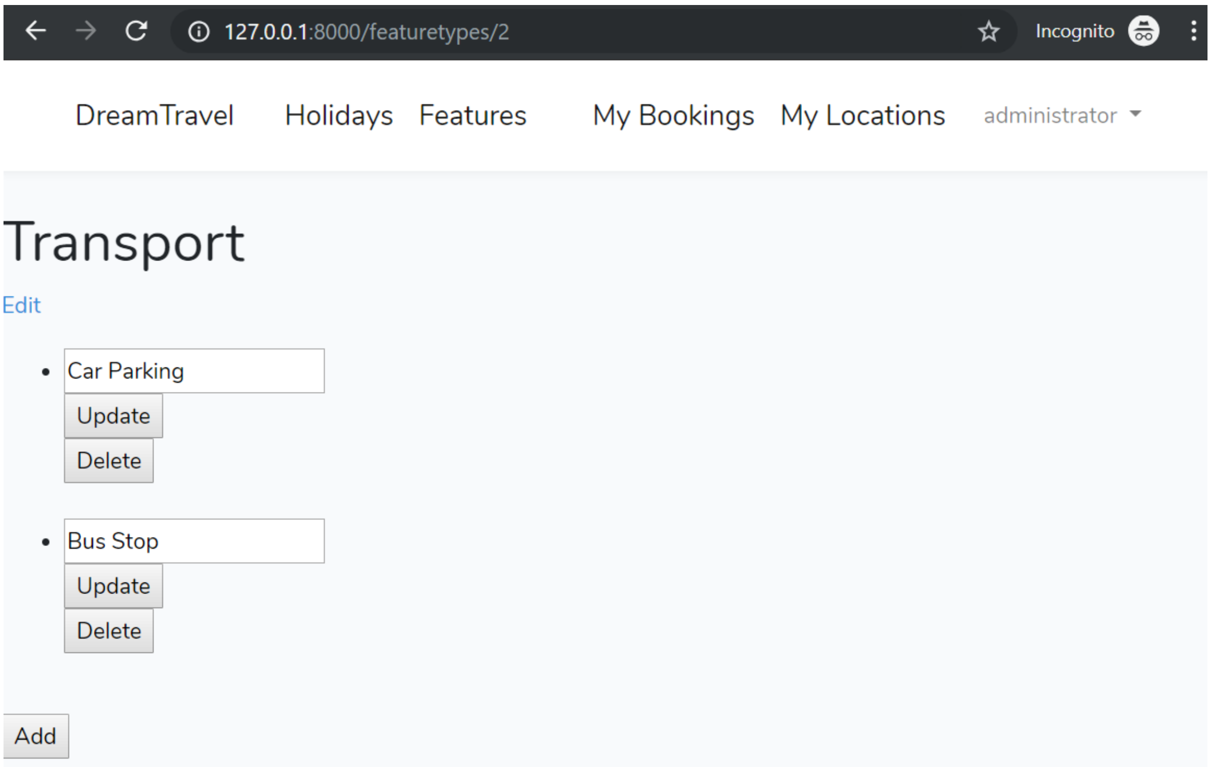


Figure 31: Admin only - modify a feature type and add features to a feature type page

4.11 Environment Details

The file containing the environment variables is:

- dreamtravel\.env

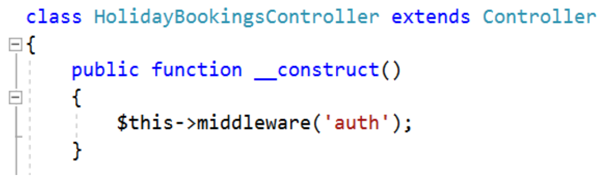
This file stores the Google API Key and the credentials needed to connect to the database. In a development scenario where there were multiple environments, each one would have its own .env file with credentials relating to that environment.

4.12 Middleware

The files containing code for the middleware are in the folder:

- dreamtravel\app\Http\Middleware

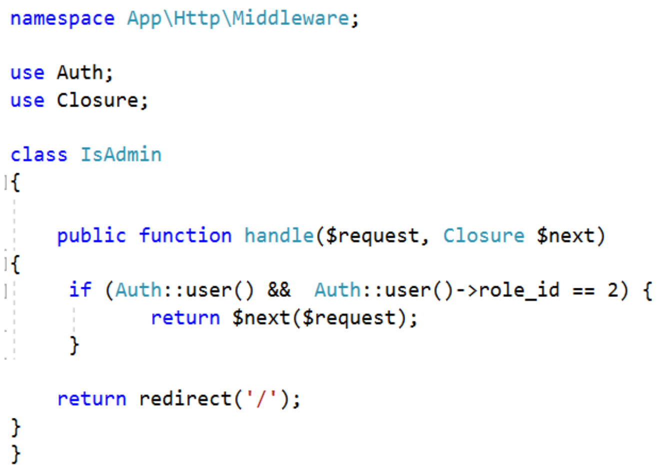
The application uses middleware to determine what type of user can perform certain actions.



```
class HolidayBookingsController extends Controller
{
    public function __construct()
    {
        $this->middleware('auth');
    }
```

Figure 32: Middleware applied to HolidayBookingsController

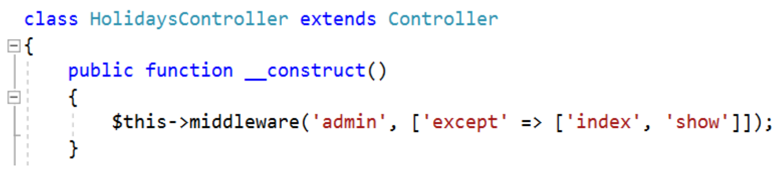
The HolidayBookingsController functions only work for authenticated users.



```
namespace App\Http\Middleware;

use Auth;
use Closure;

class IsAdmin
{

    public function handle($request, Closure $next)
{
     if (Auth::user() &&  Auth::user()->role_id == 2) {
            return $next($request);
     }

    return redirect('/');
}
}
```

Figure 33: IsAdmin middleware

The IsAdmin middleware was written in order to determine whether the logged in user has the admin role or not.



```
class HolidaysController extends Controller
{
    public function __construct()
    {
        $this->middleware('admin', ['except' => ['index', 'show']]);
    }
```

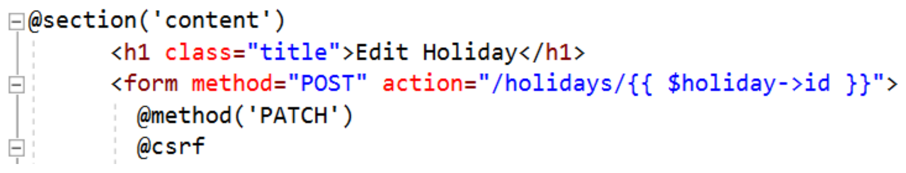
Figure 34: Middleware applied to HolidaysController

The HolidaysController functions only work for admin users, except for the index and show functions (used to display holidays on the holidays page, and holiday information for the holiday information pages) which will work for everyone.

4.13 Security

Laravel makes it easy to protect the application from cross-site request forgery (CSRF) attacks. Laravel automatically generates a CSRF "token" for each active user session managed by the application. This token is used to verify that the authenticated user is the one actually making the requests to the application. The VerifyCsrfToken middleware, included in the web middleware group, will automatically verify that the token in the request input matches the token stored in the session (Otwell, n.d.).

To avoid Cross Site Scripting (XSS) attacks the double brace {{ }} syntax is used in the blade files. This sanitises user inputs and escapes the data. Using Eloquent queries to access data tables prevents SQL injection. If raw SQL queries are used then the database could be vulnerable.



```
@section('content')
        <h1 class="title">Edit Holiday</h1>
        <form method="POST" action="/holidays/{{ $holiday->id }}">
          @method('PATCH')
          @csrf
```

Figure 35: csrf token and double brace are used to make the application secure

5 Testing

The table below contains a log of the testing performed with Guest access, standard user access and admin user access. The testing has been carried out on Chrome and Firefox browsers.

Access	Action	Outcome	Chrome	Firefox
Guest	Navigate to http://127.0.0.1:8000/ and logout if necessary	Home Page Shown	Y	Y
Guest	Click "DreamTravel" in Ribbon	Home Page Shown	Y	Y
Guest	Click "Holidays" in Ribbon	Directed to Holidays Page	Y	Y
Guest	Click on a Holiday	Directed to Holiday details page, login or register buttons are shown.	Y	Y
Guest	Click Login	Directed to Login page	Y	Y
Guest	Go back in the browser, click Register	Directed to Register page	Y	Y
Guest	Navigate to http://127.0.0.1:8000/holidays/create	Welcome Page Shown	Y	Y
Guest	Navigate to http://127.0.0.1:8000/holidays/1/edit	Welcome Page Shown	Y	Y
Guest	Navigate to http://127.0.0.1:8000/featuretypes	Welcome Page Shown	Y	Y
Guest	Navigate to http://127.0.0.1:8000/featuretypes/1	Welcome Page Shown	Y	Y
Guest	Navigate to http://127.0.0.1:8000/featuretypes/1/edit	Welcome Page Shown	Y	Y
Guest	Navigate to http://127.0.0.1:8000/featuretypes/create	Welcome Page Shown	Y	Y
Guest	Navigate to http://127.0.0.1:8000/bookings	Welcome Page Shown	Y	Y
Guest	Navigate to http://127.0.0.1:8000/locations	Welcome Page Shown	Y	Y
Guest	Click on "Register" in Ribbon	Directed to Register page	Y	Y
Guest	Click Register button	"Please fill out this field" popup appears by Name field	Y	Y
Guest	Enter details for Name, click register	"Please fill out this field" popup appears by Email field	Y	Y
Guest	Enter a single character in Email field, click register	"Please include @ in email address" popup appears by Email field	Y	Y
Guest	Add an @ symbol in Email field, click register	"Please enter a part following @" popup appears by Email field	Y	Y

Guest	Enter a plausible Email, click register	"Please fill out this field" popup appears by Confirm Password field	Y	Y
Guest	Enter a single character in Password field, click register	"Please fill out this field" popup appears by Password field	Y	Y
Guest	Enter a different character in Confirm Password field, click register	"The password must be at least 6 characters" and "The password confirmation does not match" error messages appear	Y	Y
Guest	Enter the same string of 8 characters into both the Password and Confirm Password fields, click register	Directed to Home page. My Bookings, My Locations and the User's Name appear in the Ribbon	Y	Y
Guest	Select the User's Name drop down and choose Logout	Welcome Page Shown	Y	Y
User	Navigate to http://127.0.0.1:8000/login, use the credentials Email: test01@test.com, Password: test01 to login	Directed to Home page. My Bookings, My Locations and test01 appear in the Ribbon	Y	Y
User	Click "DreamTravel" in Ribbon	Home Page Shown	Y	Y
User	Click "Holidays" in Ribbon	Directed to Holidays Page	Y	Y
User	Click on a Holiday	Directed to Holiday details page, "Book" button shown	Y	Y
User	Click Book	Pop up appears asking how many places you wish to book	Y	Y
User	Click Cancel	View returns to the Holiday details page	Y	Y
User	Click Book, choose a number from the drop down	Total Cost dynamically updates	Y	Y
User	Choose number of places = 0, and click Add	"The num places must be at least 1" error shows	Y	Y
User	Click Book, choose a number from the drop down which isn't 0 and click Add.	Directed to bookings page	Y	Y
User	Navigate to http://127.0.0.1:8000/bookings	Holidays with start date before today do not have Update or Remove buttons	Y	Y
User	Select "View" for a holiday with start date prior to today	Directed to the holiday details page. Note the Book button is not present	Y	Y

User	Go Back. Select "View" for a holiday with start date after today	Directed to the holiday details page. Note the Book button is present. Note the number of places available.	Y	Y
User	Go Back. For the same holiday as above, choose a row of the "Places" column and change the number	The new cost is displayed under the Update button	Y	Y
User	Select "Update" for the cell	The "Places" and "Total Cost" columns are updated.	Y	Y
User	Select "View" for the holiday	Note the number of places in the holiday details page has updated.	Y	Y
User	Go Back. Select "Remove" for any Holiday	The holiday is removed from the table.	Y	Y
User	Navigate to http://127.0.0.1:8000/locations	Directed to locations page, where holiday locations are plotted on a map.	Y	Y
User	Navigate to http://127.0.0.1:8000/holidays/create	Welcome Page Shown	Y	Y
User	Navigate to http://127.0.0.1:8000/holidays/1/edit	Welcome Page Shown	Y	Y
User	Navigate to http://127.0.0.1:8000/featuretypes	Welcome Page Shown	Y	Y
User	Navigate to http://127.0.0.1:8000/featuretypes/1	Welcome Page Shown	Y	Y
User	Navigate to http://127.0.0.1:8000/featuretypes/1/edit	Welcome Page Shown	Y	Y
User	Navigate to http://127.0.0.1:8000/featuretypes/create	Welcome Page Shown	Y	Y
User	Navigate to http://127.0.0.1:8000/login	Directed to Home page	Y	Y
User	Navigate to http://127.0.0.1:8000/register	Directed to Home page	Y	Y
Admin	Navigate to http://127.0.0.1:8000/login, use the credentials Email: admin@dreamtravel.com, Password: administrator to login	Directed to Home page. Features, My Bookings, My Locations and administrator appear in the Ribbon	Y	Y
Admin	Navigate to http://127.0.0.1:8000/holidays/create	Directed to page for adding holidays	Y	Y
Admin	Navigate to http://127.0.0.1:8000/holidays/1/edit	Directed to page for editing the holiday	Y	Y
Admin	Navigate to http://127.0.0.1:8000/featuretypes	Directed to page with list of feature types	Y	Y
Admin	Navigate to http://127.0.0.1:8000/featuretypes/1	Directed to page with list of features under the parent feature type	Y	Y

Admin	Navigate to http://127.0.0.1:8000/featuretypes/1/edit	Directed to page for editing the feature type	Y	Y
Admin	Navigate to http://127.0.0.1:8000/featuretypes/create	Directed to page for creating a new feature type	Y	Y

Table 1: Log of testing for DreamTravel web application

6 Conclusion

The application meets the criteria set out in the brief in that a user can register to create an account, book, change or cancel a holiday and view the locations where they have been. The application consumes the Google Maps API in order to display the location information.

In the literature review it was identified that a scalable, easily managed application would be preferable to static HTML content for the future growth and maintainability of the web application. Based upon this research, the DreamTravel application was designed so that adding a holiday requires minimal work from an administrator. As mentioned in 4.10, the admin view of the application could be extended to include more functionality so that direct database changes are never required. The Features and FeatureTypes tables were created with the intention of having a preferences section on the Holidays page for users to filter holidays based upon their features, in a similar manner to the Thomas Cook site in 1.1.2 – this could be completed with more development time.

Another improvement would be to add confirmatory popups before committing to the booking, updating or cancelling of holidays in order to prevent user errors before critical actions are carried out (Nielsen, 2018).

The testing could be further enhanced by writing unit tests with the Laravel framework in order to test in detail (Kumar, 2018) and reduce the chance of new code introducing bugs to the existing application (Novoseltseva, 2017).

The DreamTravel application is secure and has been extensively tested. It was designed with users in mind, uses a popular PHP framework and has been developed in an agile manner.

7 References

Adams, P., 2016. *Why cards are the future of the web*. [Online]
Available at: https://www.intercom.com/blog/why-cards-are-the-future-of-the-web/
[Accessed 30 April 2019].

Anon., 2019. [Online]
Available at:
https://www.telegraph.co.uk/content/dam/property/2017/08/18/TELEMMGLPICT000137499465_trans_NvBQzQNjv4BqDYAoRqCxn9rh7bhJY-GZd0XAZNwMxFR5WejpuNqmd6Q.jpeg?imwidth=1400

Anon., 2019. *Surfing Beeches in Cornwall*. [Online]
Available at: https://www.akholidays.co.uk/2017/12/09/surfing-beaches-cornwall/

Armel, J., 2014. *Web application development with Laravel PHP Framework version 4*, Helsinki: Metropolia.

Bean, M., 2015. *Laravel 5 Essentials*. Birmingham: PACKT Publishing.

Ertaul, L., n.d. *Implementation and Performance Analysis of PBKDF2, Bcrypt, Scrypt Algorithms*, Hayward: CSU East Bay.

Farm, C. H., 2019. [Online]
Available at: https://www.pinterest.co.uk/pin/462744930444551884/

Farrell, J., 2007. Rich Internet Applications The Next Stage of Application Development. *IEEE*.

Fraternali, P., 2010. Rich Internet Applications. *IEEE*.

Jain, N., 2014. REVIEW OF DIFFERENT RESPONSIVE CSS FRONT-END FRAMEWORKS. *Journal of Global Research in Computer Science*, p. 7.

Johnson, J., 2014. *Designing with the Mind in Mind*. Waltham: Elsevier.

Kumar, P., 2018. *Laravel Unit Testing*. [Online]
Available at: https://www.cloudways.com/blog/laravel-unit-testing/

Lancor, L., 2013. Analyzing PHP Frameworks for Use in a Project-Based Software Engineering Course. *SIGCSE*.

Myerscough, R., 2019. *23rd Broadland Youth Regatta*. [Online]
Available at: https://www.sail-worldcruising.com/news/192029/23rd-Broadland-Youth-Regatta-at-Wroxham

Nielsen, J., 2018. *Confirmation Dialog*. [Online]
Available at: https://www.nngroup.com/articles/confirmation-dialog/

Novoseltseva, E., 2017. *8 Benefits of Unit Testing*. [Online]
Available at: https://dzone.com/articles/top-8-benefits-of-unit-testing

Otwell, T., n.d. *CSRF Protection - Laravel - The PHP Framework For Web Artisans*. [Online]
Available at: https://laravel.com/docs/5.8/csrf
[Accessed 01 04 2019].

Pope, S., 1998. A cookbook for using the model - view controller user interface paradigm in Smalltalk - 80. *ResearchGate*, p. 32.

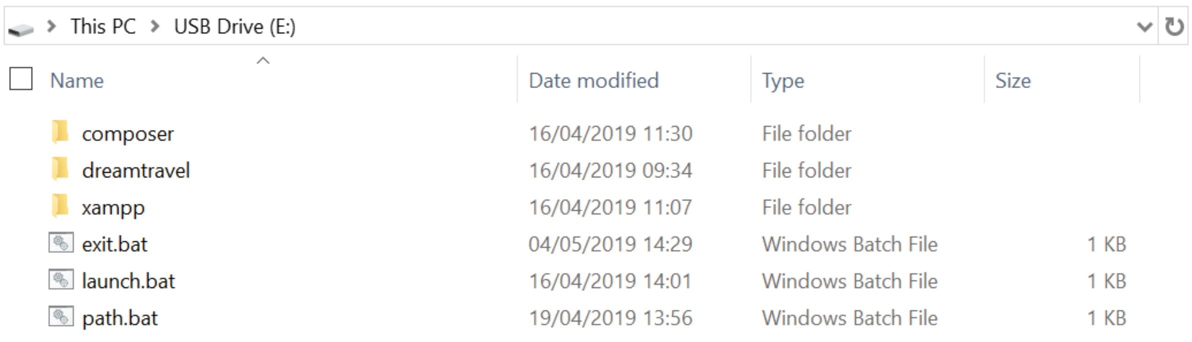
Saffer, D., 2007. Interface Design Basics. In: *Designing for Interaction*. Berkeley: AIGA, p. 128.

Suthar, S., 2017. *Top 6 reasons why live chat is winning the customer support race*. [Online]
Available at: https://acquire.io/blog/live-chat-winning-customer-support-race/amp/
[Accessed 30 April 2019].

8 User Guide

8.1 Set Up

Using a Windows computer, insert the USB into the E:\ drive.



This PC > USB Drive (E:)

Name	Date modified	Type	Size
composer	16/04/2019 11:30	File folder	
dreamtravel	16/04/2019 09:34	File folder	
xampp	16/04/2019 11:07	File folder	
exit.bat	04/05/2019 14:29	Windows Batch File	1 KB
launch.bat	16/04/2019 14:01	Windows Batch File	1 KB
path.bat	19/04/2019 13:56	Windows Batch File	1 KB

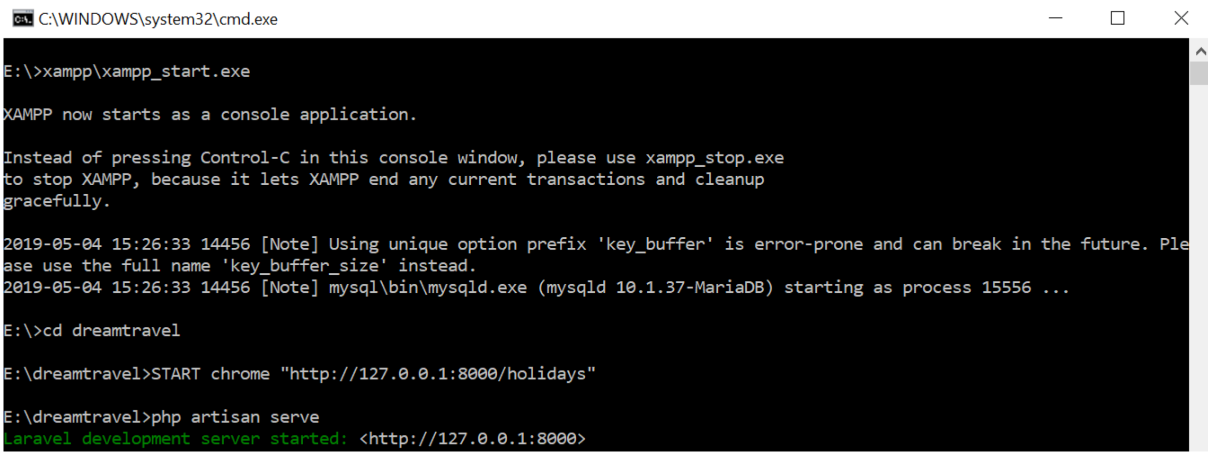
Figure 36: Files on the USB Drive

Open File Explorer and navigate to the E:\ drive.

If this is the first time the software is being launched on the computer, double click on path.bat (this sets the PHP path variable in the local environment).

8.2 Launch

To launch the application, double click on launch.bat.



C:\WINDOWS\system32\cmd.exe

```
E:\>xampp\xampp_start.exe

XAMPP now starts as a console application.

Instead of pressing Control-C in this console window, please use xampp_stop.exe
to stop XAMPP, because it lets XAMPP end any current transactions and cleanup
gracefully.

2019-05-04 15:26:33 14456 [Note] Using unique option prefix 'key_buffer' is error-prone and can break in the future. Ple
ase use the full name 'key_buffer_size' instead.
2019-05-04 15:26:33 14456 [Note] mysql\bin\mysqld.exe (mysqld 10.1.37-MariaDB) starting as process 15556 ...

E:\>cd dreamtravel

E:\dreamtravel>START chrome "http://127.0.0.1:8000/holidays"

E:\dreamtravel>php artisan serve
Laravel development server started: <http://127.0.0.1:8000>
```

Figure 37: Launching the application

This file starts XAMPP, opens a Chrome browser and starts the Laravel application.

8.3 Use

In the Chrome browser the user can now explore the application.

To login with an existing account, select "Login" in the navigation pane and choose a set of credentials from below:

User Role	Username	Password
Guest	test01@test.com	test01
Administrator	admin@dreamtravel.com	administrator

8.4 Exit

To exit the application safely, double click on exit.bat. A command prompt window will launch and stop XAMPP. When this window has closed itself, navigate to the command prompt window opened on launch, and press "Ctrl + C". A message will appear asking if the user wants to terminate the job. Select "Y".

The USB can now be ejected.