



Software Project

Liam Ronan

N00212101

Software Project

Develop a Barbershop booking system with authentication.

Year 2 2022-23

DL836 BSc (Hons) in Creative Computing

Link to resources created as part of the project.

GitHub	Liam Ronan GitHub
Video	Link to your video file (MS Stream, YouTube)
Miro	Miro Board
Figma	Figma Wireframe Design

Table of Contents

1	Introduction	6
2	Business Concept	7
2.1	Business Idea	7
2.2	Business model	7
2.3	Market Research	7
2.4	Marketing/Advertising	7
2.5	Suppliers	8
2.6	Competitors	8
2.7	Employees	8
2.8	Environmental Impact	9
3	Requirements	9
3.1	Introduction	9
3.2	Requirements gathering	9
3.2.1	Similar applications	9
3.2.2	Interviews	12
3.3	Requirements modelling	16
3.3.1	Functional requirements	16
3.3.2	Non-functional requirements	16
3.3.3	Use Case Diagrams	17
3.4	Feasibility	18
4	Web application Design	19
4.1	Layout	19
4.2	Interaction	19
4.3	Colour schemes	19
4.4	Font choices	20

4.5	Wireframes	21
5	Database Design	27
5.1	Introduction	27
5.2	Description	27
5.3	Business Reporting Requirements	27
5.4	Textual Representation of Dataset	28
5.5	Business Rules	28
5.6	Entity Relationship Diagram	28
5.7	Tables	29
5.8	Database Dictionary	30
6	System Design/ Architecture Overview	33
6.1	Introduction	33
6.1.1	Laravel Version	33
6.1.2	Dependencies	33
6.1.3	Front-End.....	33
6.1.4	Benefits.....	Error! Bookmark not defined.
6.1.5	Directory Structure	Error! Bookmark not defined.
6.2	Model View Controller	33
6.2.1	Explanation	33
6.2.2	My Implementation	34
6.3	User Authentication	37
6.4	Routing	37
6.5	Templating	38
7	Testing	40
7.1	Introduction	40
7.2	Functional Testing	40

7.2.1	Render Pages	41
7.2.2	Authentication	42
7.2.3	Seeding	42
7.2.4	Discussion of Functional Testing Results	43
7.3	User Testing	43
7.4	Conclusion	43
8	Project Management	45
8.1	Introduction	45
8.2	Project Phases	45
8.2.1	Requirements	45
8.2.2	Design	45
8.2.3	Implementation	45
8.2.4	Testing	45
8.3	Project Management Tools	46
8.3.1	GitHub Project	46
8.3.2	GitHub	47
9	Reflection	47
9.1	Your views on the project	47
9.2	How could the project be developed further?	47
9.3	Assessment of your learning.	48
9.4	Completing a large software development project	48
9.5	Technical skills	48
9.6	Further competencies and skills	48
10	References	49

1 Introduction

This software project report covers the design and development of a booking system for barbershops that aimed at streamlining the booking process for both clients and barbers.

The hairdressing and grooming field, notably barbershops, was the application area.

- **Tech Stack:** The tech stack for this project was chosen after considering the project's goals and objectives. I chose MySQL as my database management system, Laravel for the back end, and Bootstrap 5 and SASS for the front-end.
- **Tools:** A variety of tools were used during the development process to help with different tasks. Code was written and edited in VS Code. The user interface was wireframed and designed using Figma. Miro was used to document findings and various forms of research. The database was managed and tracked using phpMyAdmin.
- **Project Management:** The project management approach chosen for this was the Agile methodology. Agile is a flexible and iterative methodology that supports quick prototyping and development, which makes it perfect for software development projects. I used GitHub to manage and store the projects' code and resources.

2 Business Concept

2.1 Business Idea

The business concept entails creating a web application for a barbershop that allows registered users to make an appointment, view the various services the barbershop offers, and read about the barbers that work in the shop. The application administrator may have control over bookings, barbers, and the services.

2.2 Business model

The business model for this project, which was intended at developing a booking system for a barbershop, was to present a means for the barbershop to simplify their booking process and enhance client experience. The system also wanted to give clients a simple and convenient booking experience.

1. **User registration and login:** Enable users to set up an account and log in or register.
2. **Booking system:** Implement a booking system online so that clients may choose their preferred date, time, and barber when making an appointment.

2.3 Market Research

Before implementing the website, it's important to conduct various forms of market research to gather information about the target audience and competition.

1. **Target audience:** To define the target audience, firstly must observe the clients of the barbershop. Gather information such as demographics, location, and needs.
2. **Analyze competition:** Learn about local business rivals. Examine the methods they use to market their products and services.
3. **Survey customers:** Customers' preferences for making reservations and making purchases should be enquired about.
4. **Evaluate the market:** Determine the demand for an online booking system for the barbershop.

2.4 Marketing/Advertising

The primary marketing and advertising method would be through digital media as it will reach a wider audience, cost effectiveness, easily tracked and through targeted advertising.

1. **Social media marketing:** Utilize social media sites like Facebook, Instagram, and Twitter to increase awareness about the brand and reach a larger audience.
2. **Influencer marketing:** Partner with local influencers and bloggers who have a large following on social media to promote the barbershop.
3. **Search Engine Optimization:** Include relevant keywords, meta tags, and title descriptions on the barbershop website to make it more search engine friendly.

2.5 Suppliers

The suppliers for the barbershop booking system may include:

1. **Cloud hosting providers:** for example, Amazon Web Services, Microsoft Azure, or Google cloud could host the application and data.
2. **Payment Processing:** such as Stripe or PayPal for handling payments and transactions.
3. **Software Development tools and services:** for managing the codebase and automating testing and deployment.

2.6 Competitors

Although there are many competitors in the local area offering similar services, currently no other barbershop has a website with an integrated booking system. A few barbershops do have a website, but they only contain a time schedule of the shop or a link to the Facebook page. This extends the time to make a booking with these shops dramatically.

The only other main competitor that has a website and booking system is targeting a very specific audience explained in the competitor analysis.

2.7 Employees

The primary service providers are the barbers. however, several employees may be hired if the business continues to grow such as:

1. **Marketing and advertising specialists:** These will oversee promoting the business using a variety of channels, including SEO, email marketing, and social media.
2. **Operations and logistics personnel:** Employees are responsible for overseeing the daily activities of the barbershop, including placing orders for supplies, handling payments, and setting up appointments.

2.8 Environmental Impact

Due to the location of the barbershop being on the main street with very little parking, this alone will mitigate the impact by encouraging customers to walk a certain distance. Overall, the environmental impact of a barbershop booking system is very minimal as the system isn't very demanding or doesn't require large servers that need power.

3 Requirements

3.1 Introduction

In this section of the report, I will outline the requirements of the barbershop booking system, which will be developed based on the input and feedback received from user questionnaires. The system will be integrated into a website that is built around the context and clients of the existing shop.

3.2 Requirements gathering

The requirements' gathering process is an essential phase in the software development process. This section will cover the process for gathering and assessing the barbershop booking system's needs.

3.2.1 Similar applications

This section will look at a few applications that are comparable to the barbershop booking system.

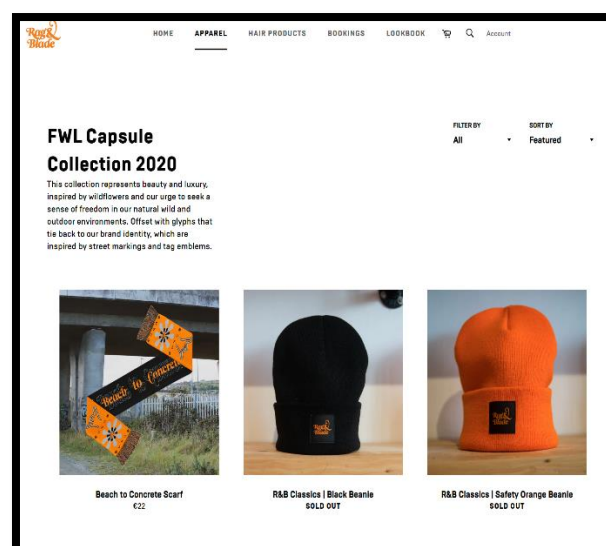


Figure 1

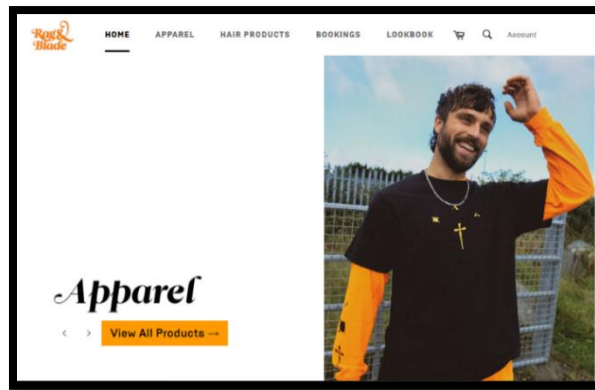


Figure 2

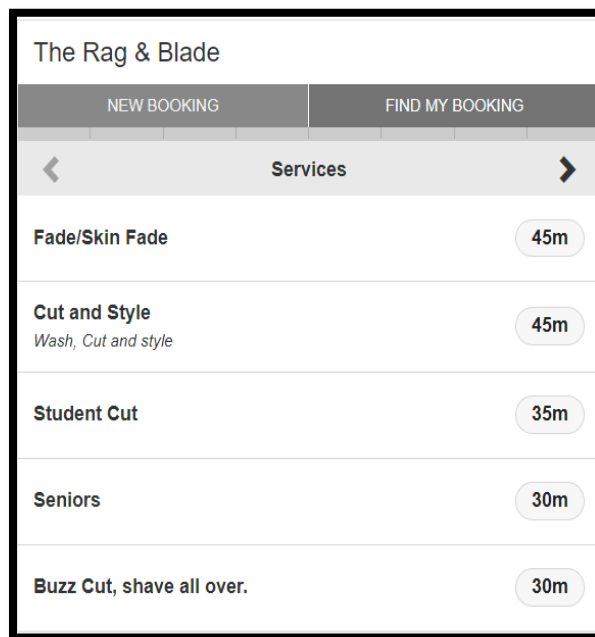


Figure 3

An application for a business in the same area with a similar focus—primarily on apparel and oriented toward a younger customer base shown above. Users of this application can reserve bookings, buy clothes, and choose from a modest selection of things like hair wax, gels, and other items. The website's design, including the typography, layouts, images, and other elements, is geared for a younger demographic.

Advantages

This application has the advantage of accomplishing its goals, such as emphasizing apparel and products. Their design, which incorporates images, clothing, and other elements in a throwback 1990s manner, further reinforces this.

Disadvantages

The business is still a barbershop with their primary income sourced from booked appointments. Therefore, the booking system used, as shown above, lacks the same design and style as the rest of the website.

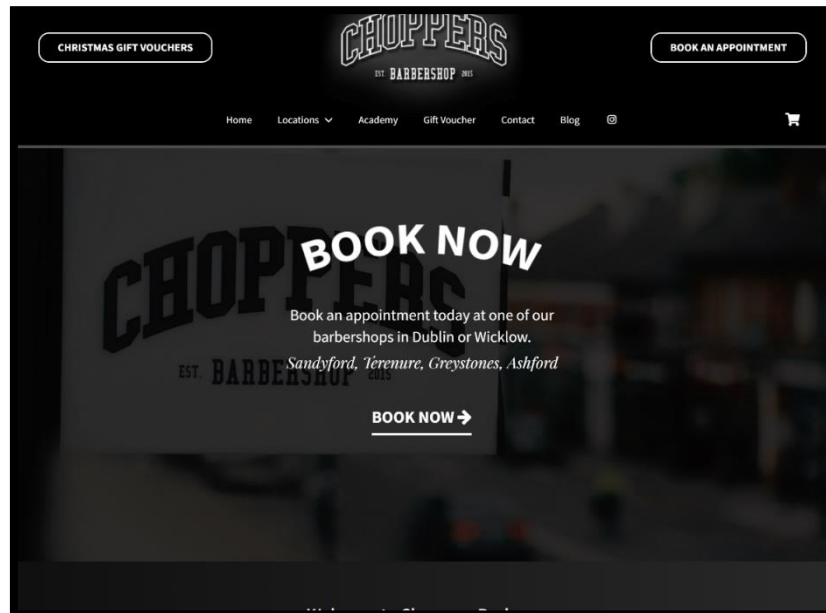


Figure 4

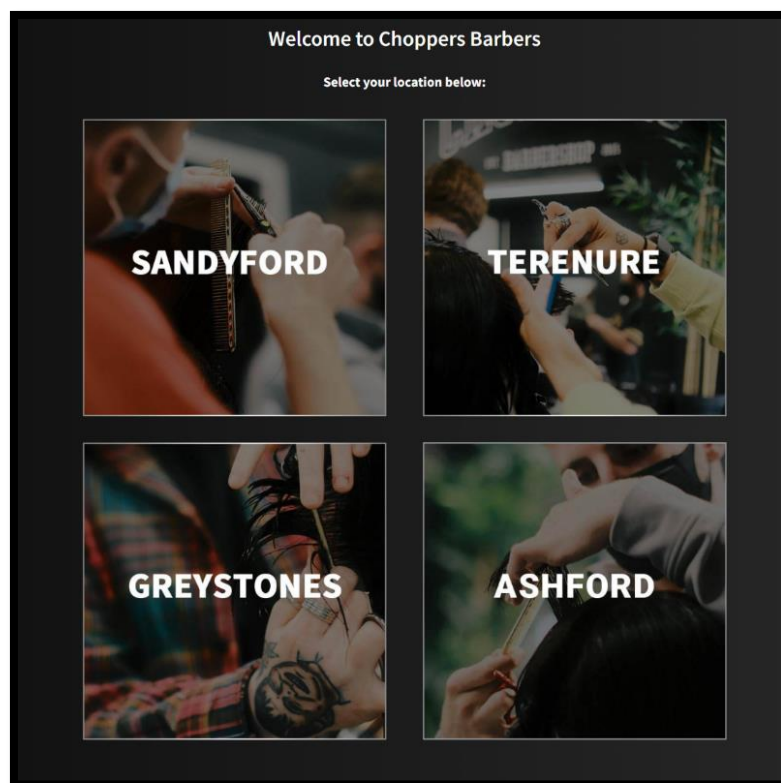


Figure 5

The website in figure 4, 5, 6 for a sizable chain of barbershops with locations throughout Dublin and Wicklow.

Advantages

This website has the benefit of drawing the user's attention to the site's main emphasis in the center of the screen right away from the homepage. It is evident that they want people to schedule appointments quickly.

Disadvantages

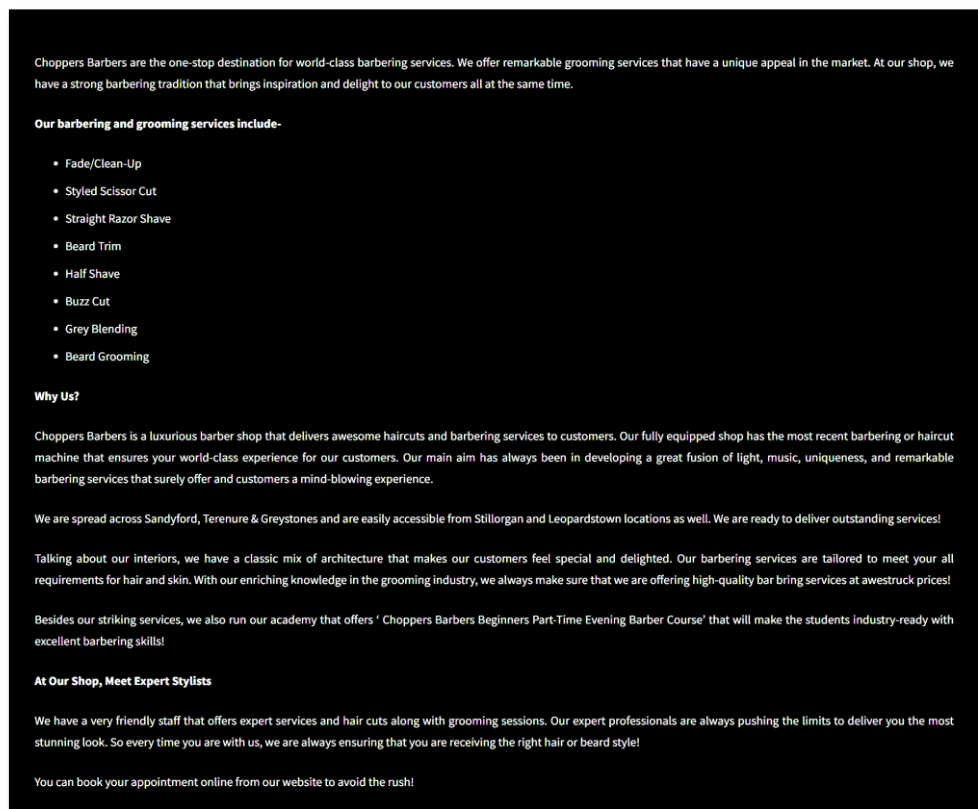


Figure 6

There is a substantial amount of text below the location buttons. It's quite difficult to read because of the white on black colour scheme and the size of the text.

3.2.2 Interviews

The interview section tries to gather information about the online booking experiences of people who have used similar systems. Two people were questioned, and the replies were analyzed to see which aspects they thought need to be prioritized in the design of a barbershop booking system. Additionally, any challenges they had using a booking system were discovered using their feedback. The interview questions and answers are seen in figures 8, 9, 10.

Software Project Interview

I am conducting this interview to gather information to aid in the design and functionality of an online barbershop application where users may login/register, book appointments and purchase various hair products such as hair gels, waxes and barbering items.

1. Describe your online booking experience?

1st Interviewee:

- Mostly hassle-free.
- May view prices easily.
- Booking online can be cheaper.

2nd Interviewee:

- There are sometimes free cancellations.
- There may be better deals booking online.

3rd Interviewee:

- Personal details can viewed easily.
- Refund policies are in place sometimes.

2. Describe your experience navigating through a booking system?

1st Interviewee:

- Can be confusing and difficult.
- Navigating through sites and their booking systems may be awkward and ask too much of the user.

2nd interviewee:

- There is too much to read at times.
- The website may remove the user from the process if the page is refreshed or if they click to return to a previous page.

3rd Interviewee:

- Terms and Conditions can be overwhelming.
- The prices should take priority.

3. Would you be interested in the website's online shop highlighting sustainability awareness?

1st Interviewee:

- I would personally like to see more websites display this kind of information.

Figure 7

2nd Interviewee:

- People are more sustainably aware, therefore more websites should display info regarding this.

3rd Interviewee:

- It wouldn't bother me if this information was beside the products.

4. Is there anything you would change with the online shopping experience?

1st Interviewee:

- Search results could be more accurate sometimes.
- The availability of products should be always correct.

2nd Interviewee:

- Sometimes the product in information is not correct.
- The imagery should always be accurate.

3rd Interviewee:

- Different versions of the websites for different languages can be confusing.
- The layouts could be simpler to navigate.
- Websites sometimes ask the user to download a PDF to view information or a menu.

5. Do you agree that the booking section should take precedence on the website?

1st Interviewee:

- Yes, I agree for the barbershop website the booking section should be front and centre.

2nd Interviewee:

- In this regard, the website should have the booking button or section on the homepage

3rd Interviewee:

- Yes, as this would make it easier for the user to make a booking

Figure 8

6. In terms of product details, do you prefer large descriptions or concise information?

1st Interviewee:

- There should be the option to view more information.

2nd Interviewee:

- Maybe shorter descriptions with more accuracy.

3rd Interviewee:

- For hair products, the descriptions should be in depth.
- Detailed descriptions and uses of the gels, waxes etc.

Figure 9

Due to their prior experiences with other websites, the results of the interview show clients prefer a simple and user-friendly booking system. Additionally, it was determined that the booking section of the barbershop website should be prominently displayed on the homepage as it is the primary feature that users engage with. This information emphasizes how important it is to design an easy-to-use and accessible booking system to improve user experience and speed up the booking process for clients.

3.3 Requirements modelling

This report's section on requirements modeling focuses on describing the functional and non-functional requirements for the barbershop booking system. This section's goal is to clearly explain what the system should do and how it should behave in different situations. A use case diagram was made to do this, outlining the various user roles and the actions that go along with them in the system.

3.3.1 Functional requirements

The application requires a certain amount of functionality as listed below in order of priority:

1. The most important functionality requirement is to allow a user to make a booking with the application.
2. The ability for users to view their appointment, make changes or cancel their booking.
3. Allow users to view the various services offered and the barbers associated with them.
4. Authentication for the user with a login/register functionality.
5. Admin should have complete control over bookings, barbers, and services. Admin may perform CRUD functionality with all aspects listed.

3.3.2 Non-functional requirements

There are various non-functional requirements that I will list below in further detail:

1. **CSS:** In terms of the user interface and UX design, ensuring that the CSS is up to standard is a major priority with color and font consistencies, readable and correctly sized text, and usable components such as date picker, cards etc.
2. **Semantic tags:** Using semantic HTML tags for user accessibility and users with screen readers. Semantic tags also may be used to enhance SEO.
3. **Performance:** To increase performance, a limited amount of JavaScript will be used. If JavaScript is used, I will make sure it is correctly written, so it doesn't slow down the website such as load times and rendering speeds. If large or many images are used, they will be compressed to speed up load times.
4. **Security:** In terms of security, against SQL injection make sure that requests or form inputs are correctly processed. The system should use strong encryption and authentication to protect sensitive data.

5. **Usability:** The application should be easy to use and navigate, with clear instructions and helpful feedback such as flash messages.

3.3.3 Use Case Diagrams

A use case diagram was created (Figure 11) to demonstrate the functioning of the user and admin. The "includes" connection is used in the diagram to highlight the several functions that must be executed after an earlier function. Additionally, it uses the "extends" connection to indicate the optional features of each piece of functionality.

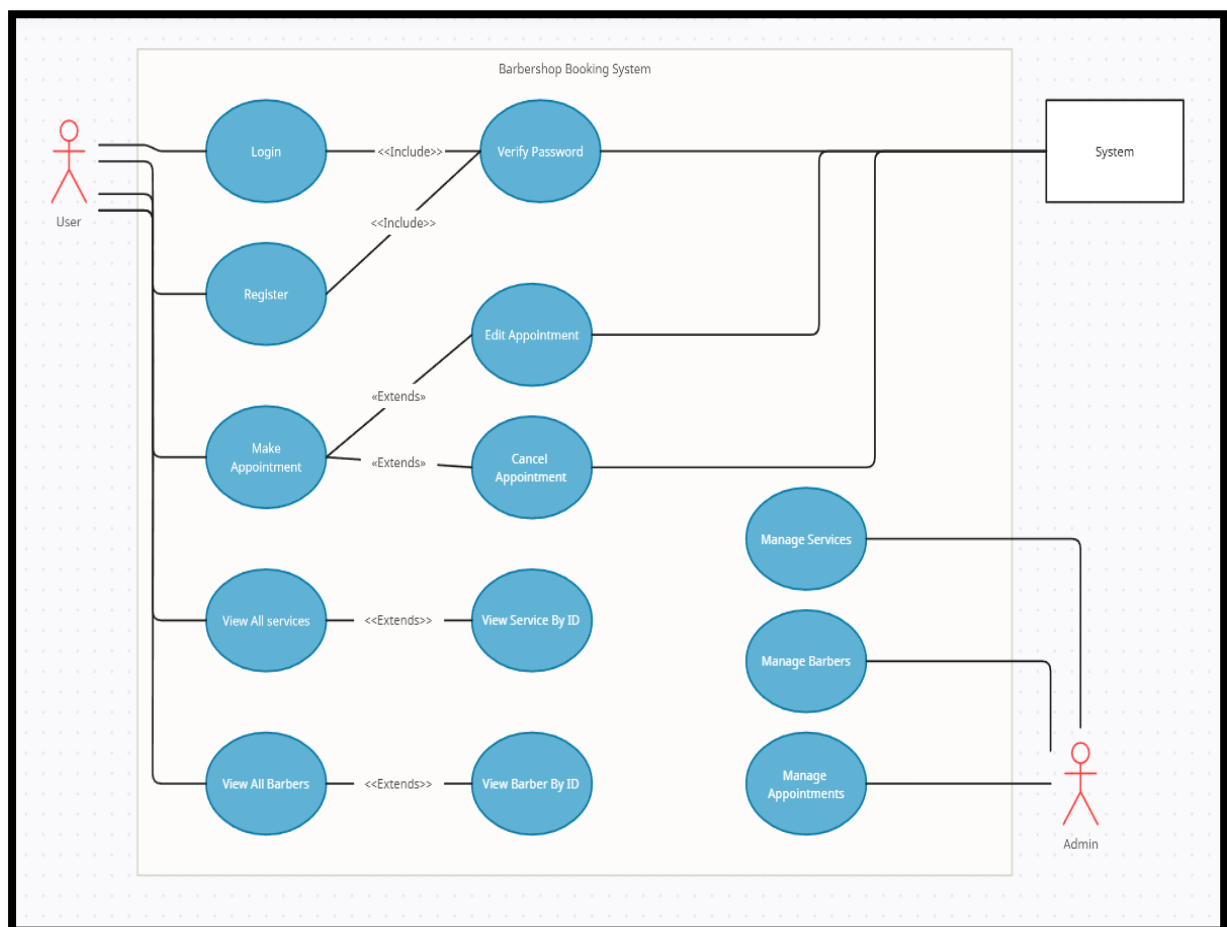


Figure 10

3.4 Feasibility

The technologies I plan to use within the project are listed below:

1. **Laravel:** Using this PHP framework will be used as the main building blocks for the project. CRUD functionality, database and authentication will all be made possible using Laravel.
2. **Bootstrap 5:** This framework will be used for the HTML Layouts such as the header, footer, and navigation. I may use pre-existing bootstrap components for the products, forms, and buttons.
3. **SASS:** To further customize the design of the website to suit the users' needs and existing design styles of the barbershop. I may use sass functions, variables, partials and nesting.
4. **Clockwork extension:** I use this extension in the browser to view any currently executing SQL commands within the website.
5. **PHPpest:** This is a larval package for feature testing and unit testing to automate the testing process of various functions within the application.

4 Web application Design

4.1 Layout

The main layout of the web application will be created with Bootstrap 5, and further customizability, such as button styles, size, and spacing, can be accomplished with SASS. A straightforward navigation bar, a sizable background image with a prominent booking button, and text paired with numerous photos will make up the homepage layout.

Users can select a barber, a date and time, and the sort of service they need on the booking. A simple schedule, email address, phone number, and background picture with text are just a few of the Blade components that will be developed and may be used across multiple pages to maintain consistency and efficiency. Using the elements listed above, the About page will be dynamic and present barbers from the database.

Bootstrap will make it possible to create responsive layouts. Overall, the usage of Bootstrap and SASS will make it easier to create a user-friendly interface.

4.2 Interaction

Firstly, the user may login or register to the application. Once registered, the user will instantly see the book appointment button on the homepage. Various links will be displayed in the nav bar such as the about, home, bookings, and contact pages. If the user clicks on bookings from the nav, the page will tell the user no appointments have been made.

Once the user makes an appointment, the bookings page will show their appointment where they made change the appointment details or cancel the appointment.

The user shall view all the barbers from the about page and view their personal profile. They also can view the services offered and view more details about said service.

4.3 Colour schemes

In accordance with the barbershop's usage of black and red for its signs, T-shirts, and logos, the website's color scheme will mostly be darker hues. During the design phase, this aspect was considered. The background colors will mostly be deeper in tone with occasional red undertones. While secondary buttons will have a border color, primary buttons will have a gradient effect. Although certain headlines may have strong last words with the same red color used elsewhere on the page, text will often be white.

Here can we see an image of the colour palette and the gradient button in figures 12 and 13.

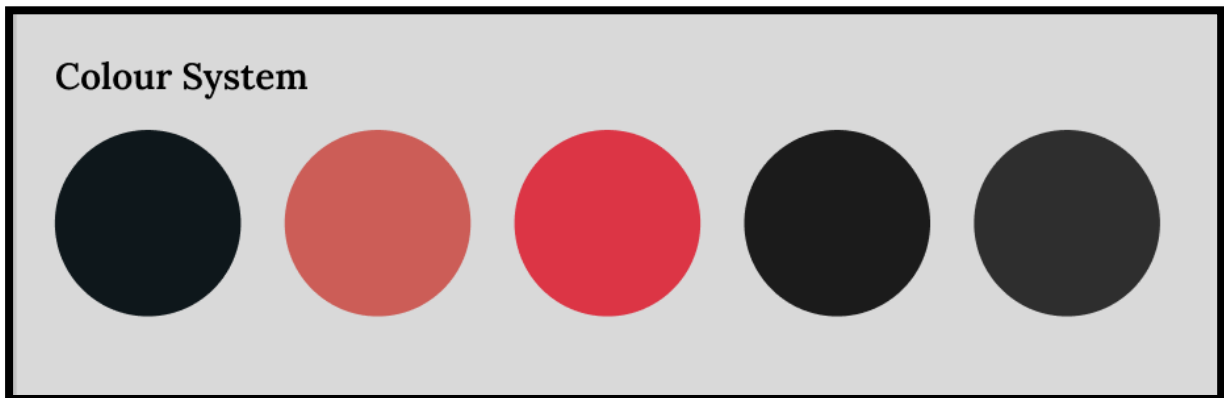


Figure 11



Figure 12

4.4 Font choices

The typography used for this project was carefully chosen to ensure that they were both aesthetically pleasing and simple to read. The choice of using Roboto and Lora was made after reviewing a few font alternatives. The elegant and timeless design of the serif the font Lora makes it the ideal choice for a traditional barbershop. Roboto is a clear, contemporary sans-serif typeface that goes well with Lora's traditional appearance. Both typefaces are also very readable, which makes them a great option for an online booking system where customers must swiftly navigate through the website's content. Overall, the combination of Roboto and Lora contributes to the creation of a user-friendly, and aesthetically pleasing design.



Figure 13

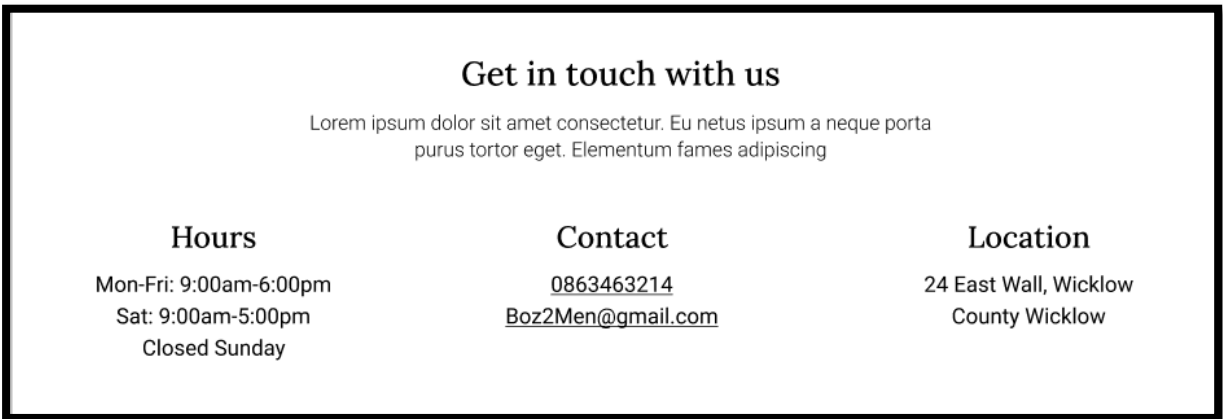


Figure 14

4.5 Wireframes

Wireframes were created using Figma to offer an easily understood visual representation of the application's design. Wireframes were made for most main pages of the application in this project, including the homepage, login/register page, the booking section, the about page, and the contact page. With placeholders for text and images, the wireframes were made to be simple and clear.

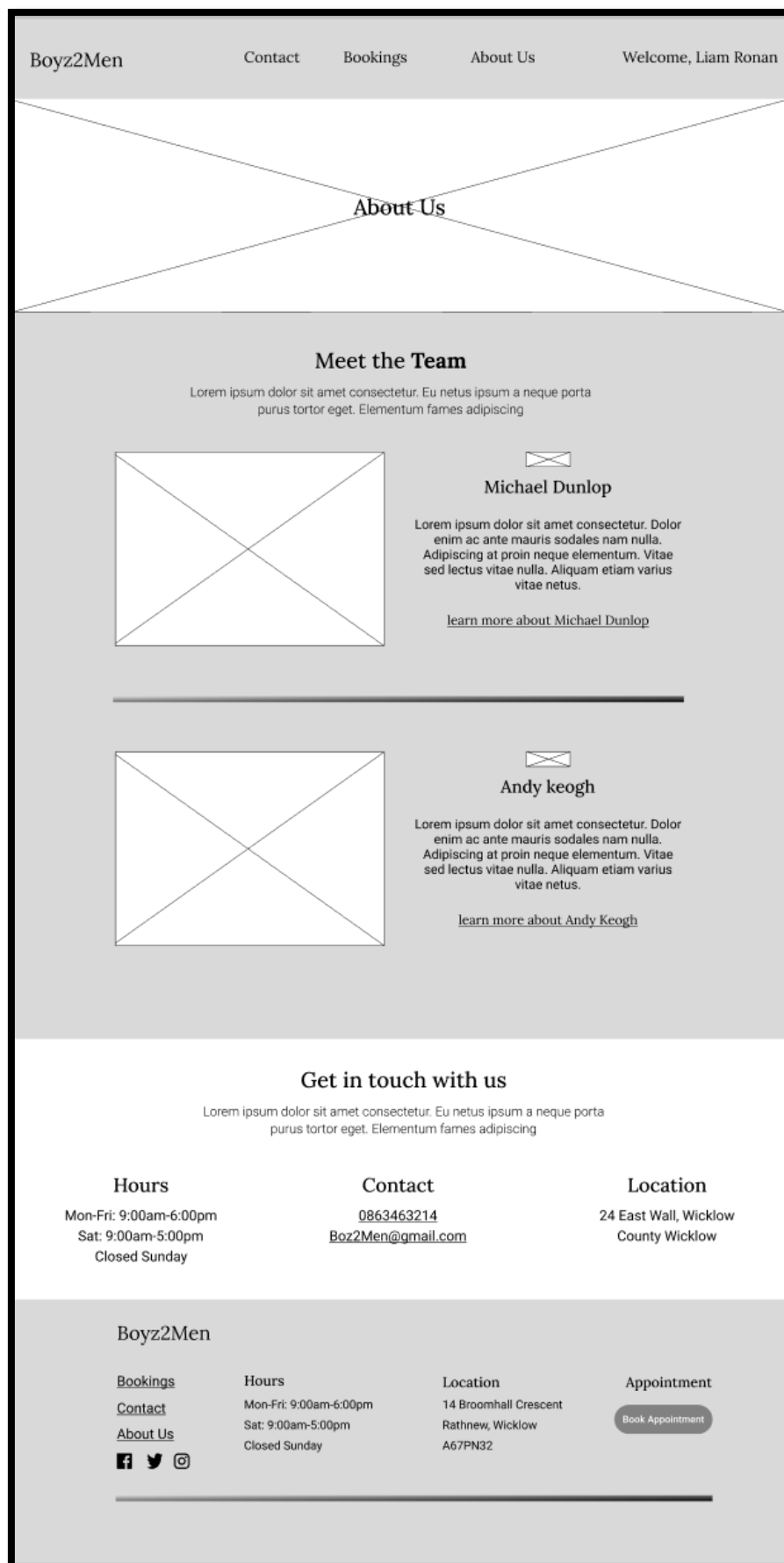


Figure 15

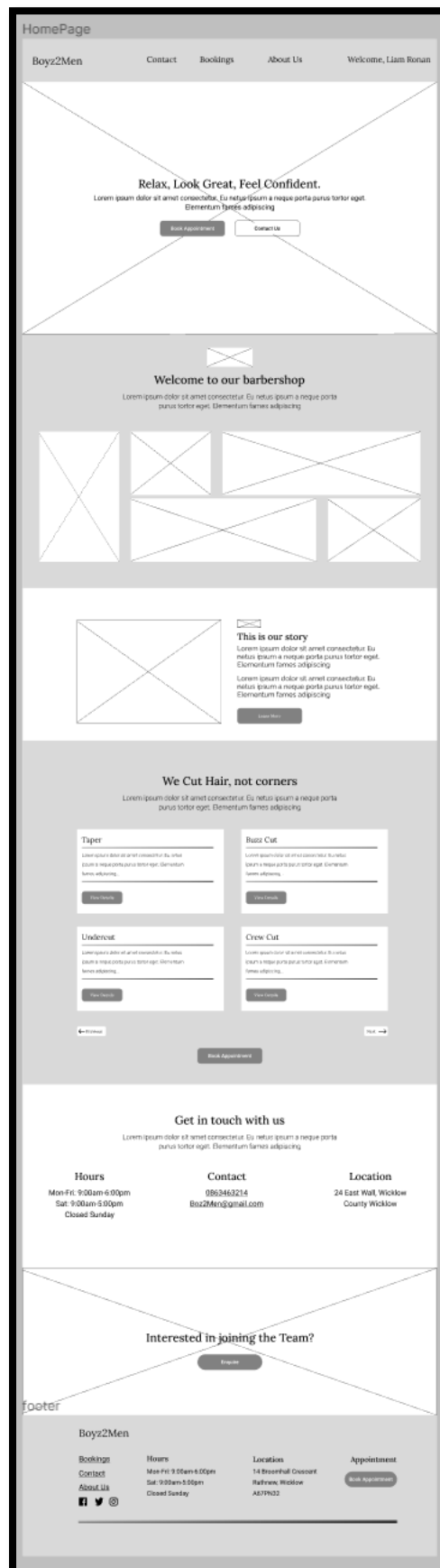


Figure 16

Contact

Boyz2Men

Contact

Bookings

About Us

Welcome, Liam Ronan

Contact Us

Get In Touch

Name

Email

Message

Send Message

Get in touch with us

Hours

Mon-Fri: 9:00am-6:00pm

Sat: 9:00am-5:00pm

Closed Sunday

Contact

0853463214

Boyz2Men@gmail.com

Location

24 East Wall, Wicklow

County Wicklow

Bookings

Contact

About Us

Hours

Mon-Fri: 9:00am-6:00pm

Sat: 9:00am-5:00pm

Closed Sunday

Location

14 Broomhall Crescent

Rathnew, Wicklow

A67PN32

Appointment

Book Appointment

Figure 17

24

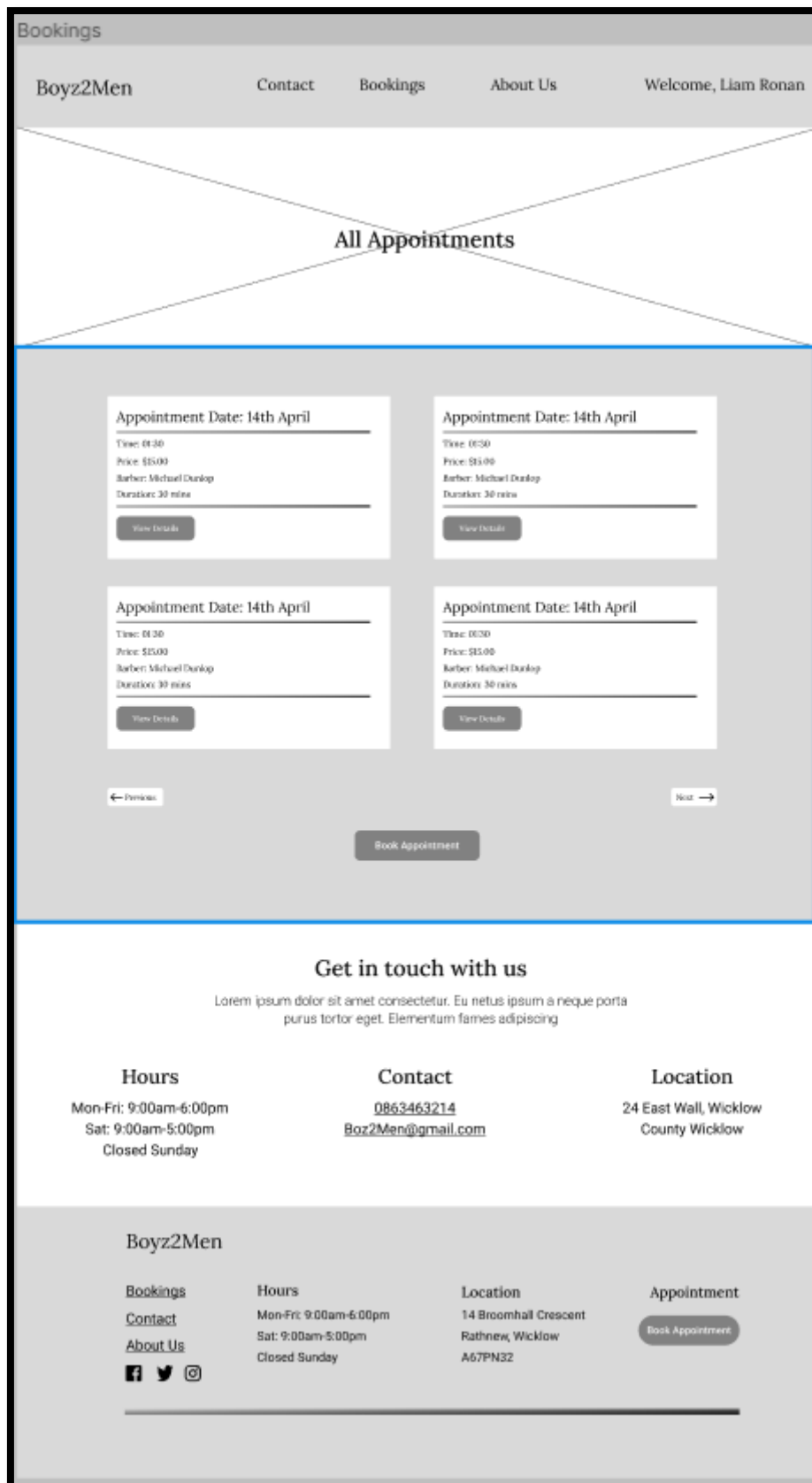


Figure 18

Register

Register

Name

Enter Name

Email Address

Enter Email Address

Password

Password

Confirm Password

Confirm Password

Register

[Forget Your Password?](#)

[Already have an account? Login here](#)

The background image for the register form shows various hair styling tools on a wooden surface, including a hairbrush, a hair dryer, and some hair product containers. The form is centered and has a semi-transparent dark background.

Figure 19

Log In

Log In

Email Address

Enter Email Address

Password

Password

Log In

[Forget Your Password?](#)

[Don't have an account? Create One here](#)

The background image for the login form shows a bathroom interior with a brick wall, a window with a plant, and a shelf with various bottles. The form is centered and has a semi-transparent dark background.

Figure 20

5 Database Design

5.1 Introduction

The database design and implementation for the barbershop booking system will be discussed in this section. The description of the database, including its objectives and purposes, will be covered in this portion. In addition, I will include a textual breakdown of the dataset's numerous tables, columns, and connections.

5.2 Description

The barbershop application will consist of a database that stores all the bookings, barbers, services, users, and user roles.

- **Bookings:** For each booking made, the database will need to store the date and time.
- **Barbers:** The barber's name, bio, email, and phone number needs to be stored in the database.
- **Services:** The services table will contain the haircut, description, price, duration, and an image.
- **Users:** This table will have a name and an email associated.
- **User roles:** will have a name and description.

5.3 Business Reporting Requirements

Below is a list of the business reporting requirements in order of importance.

1. Users must be able to make a booking, view the booking, edit the booking, and cancel a booking.
2. Users must be able to view their most recent booking.
3. Users must be able to login and register to the application.
4. Users may view all services offered by the barbershop.
5. Users may view all the barbers that work within the shop.
6. Users may view a specific service and read the details.
7. Users can view a specific barber's profile.
8. Admin can view all bookings, edit a booking, delete a booking, and view a single booking.
9. Admin may add a new barber to the shop, edit an existing barber, remove a barber, or update a barber's profile.

10. Admin will be able to add a new haircut to the list of services, remove a haircut, update a haircut, and view all.

5.4 Textual Representation of Dataset

Below I will outline a textual representation of the dataset that will be used within the application.

BOOKINGS (id, date, time, barber_id, services_id, user_id)

USERS (id, name, email)

BARBERS (id, name, email, phone_number, bio, image)

ROLES (id, name, description)

USER_ROLE (id, user_id, role_id)

SERVICES (id, haircut, description, price, duration, image)

5.5 Business Rules

The rules listed below outline how the application's tables will communicate using their assigned relationships.

- A **User** has many **bookings**.
- A **Barber** has many **bookings**.
- A **Service** has many **bookings**.
- A **Barber** has many **services**.
- A **Service** has many **barbers**.
- A **Role** has many **users**.
- A **User** has many **roles**.

5.6 Entity Relationship Diagram

The relationships between the entities in the database are visually represented by the Entity Relationship Diagram (ERD). The ERD displays the connections between the main database elements, such as users, bookings, barbers, and services. The ERD may be seen in figure 17

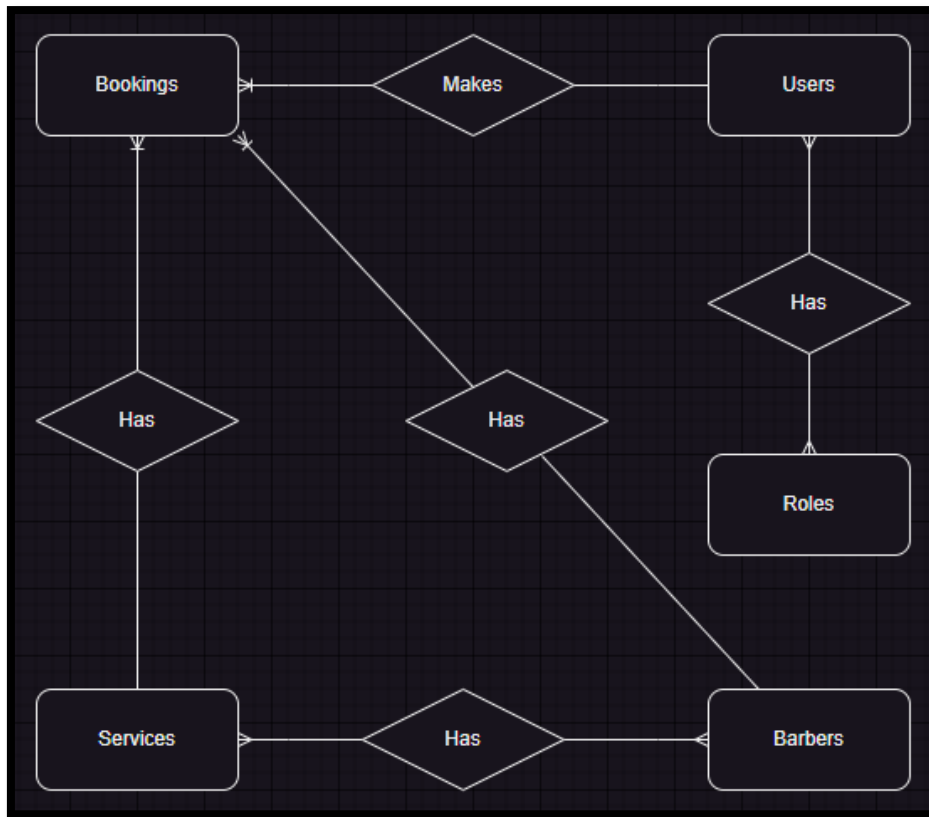


Figure 20

5.7 Tables

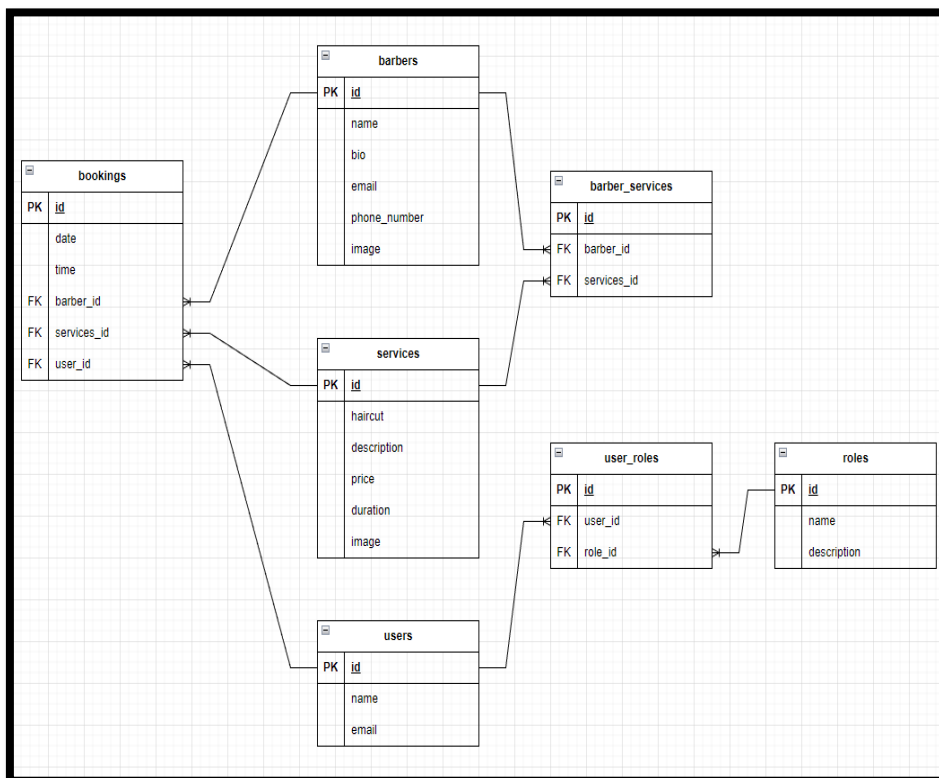


Figure 21

5.8 Database Dictionary

A reference that gives an in-depth description of each table and its attributes in the database schema is called the database dictionary.

Barbers: Figure

Column	Type	Null	Default	Links to	Comments	Media type
id (<i>Primary</i>)	bigint(20)	No				
name	varchar(255)	No				
bio	longtext	No				
email	varchar(255)	No				
phone_number	varchar(255)	No				
image	varchar(255)	Yes	NULL			
created_at	timestamp	Yes	NULL			
updated_at	timestamp	Yes	NULL			

Figure 22

Bookings: Figure 20

Column	Type	Null	Default	Links to	Comments	Media type
id (<i>Primary</i>)	bigint(20)	No				
date	date	No				
time	time	No				
created_at	timestamp	Yes	NULL			
updated_at	timestamp	Yes	NULL			
barber_id	bigint(20)	No		barbers -> id		
services_id	bigint(20)	No		services -> id		
user_id	bigint(20)	Yes	NULL	users -> id		

Figure 23

Services: Figure 21

Column	Type	Null	Default	Links to	Comments	Media type
id (<i>Primary</i>)	bigint(20)	No				
haircut	varchar(255)	No				
description	longtext	No				
price	double(8,2)	No				
duration	varchar(255)	No				
image	varchar(255)	Yes	<i>NULL</i>			
created_at	timestamp	Yes	<i>NULL</i>			
updated_at	timestamp	Yes	<i>NULL</i>			

Figure 24

Users: Figure 22

Column	Type	Null	Default	Links to	Comments	Media type
id (<i>Primary</i>)	bigint(20)	No				
name	varchar(255)	No				
email	varchar(255)	No				
email_verified_at	timestamp	Yes	<i>NULL</i>			
password	varchar(255)	No				
remember_token	varchar(100)	Yes	<i>NULL</i>			
created_at	timestamp	Yes	<i>NULL</i>			
updated_at	timestamp	Yes	<i>NULL</i>			

Figure 25

Barber_services: Figure 23

Column	Type	Null	Default	Links to	Comments	Media type
id (<i>Primary</i>)	bigint(20)	No				
barber_id	bigint(20)	No		barbers -> id		
services_id	bigint(20)	No		services -> id		
created_at	timestamp	Yes	NULL			
updated_at	timestamp	Yes	NULL			

Figure 26

Roles: Figure 24

Column	Type	Null	Default	Links to	Comments	Media type
id (<i>Primary</i>)	bigint(20)	No				
name	varchar(255)	No				
description	varchar(255)	No				
created_at	timestamp	Yes	NULL			
updated_at	timestamp	Yes	NULL			

Figure 27

User_Role: Figure 25

Column	Type	Null	Default	Links to	Comments	Media type
id (<i>Primary</i>)	bigint(20)	No				
user_id	bigint(20)	No		users -> id		
role_id	bigint(20)	No		roles -> id		
created_at	timestamp	Yes	NULL			
updated_at	timestamp	Yes	NULL			

Figure 28

6 System Design/ Architecture Overview

6.1 Introduction

Laravel tries to make certain functions, such authentication, routing, sessions, and caching, simpler to accomplish. Laravel is an incredibly scalable web framework due to the scaling-friendly nature of PHP.

6.1.1 Laravel Version

The version used in my barbershop booking system application is Laravel 9. Although Laravel 10 has been released lately, I decided to continue using Laravel 9 as I have more experience using this version.

6.1.2 Dependencies

Various dependencies are required to run Laravel for windows which is the OS I am currently using. These include PHP, composer, npm that also comes preinstalled with Node, if you plan on using a database, you may use MariaDB 10.3+, MySQL 5.7+, PostgreSQL 10.0+, SQLite 3.8.8+ or SQL Server 2017+.

6.1.3 Front-End

In terms of the front-end, Laravel may use blade components and views. Blade is a lightweight templating language that provides convenient, short syntax for displaying data, and iterating over data. You may opt to use Laravel Livewire to build powerful frontends that are dynamic and modern such as Vue or React.

6.2 Model View Controller

6.2.1 Explanation

Model View Controller is a software architectural design pattern.

- Model: The backend that contains the data logic.
- View: The front-end or graphical user interface.
- Controller: The Controller serves as the link between the View and the Model and is in charge of managing the application logic.

Below is an image of the MVC Architecture pattern in figure 18.

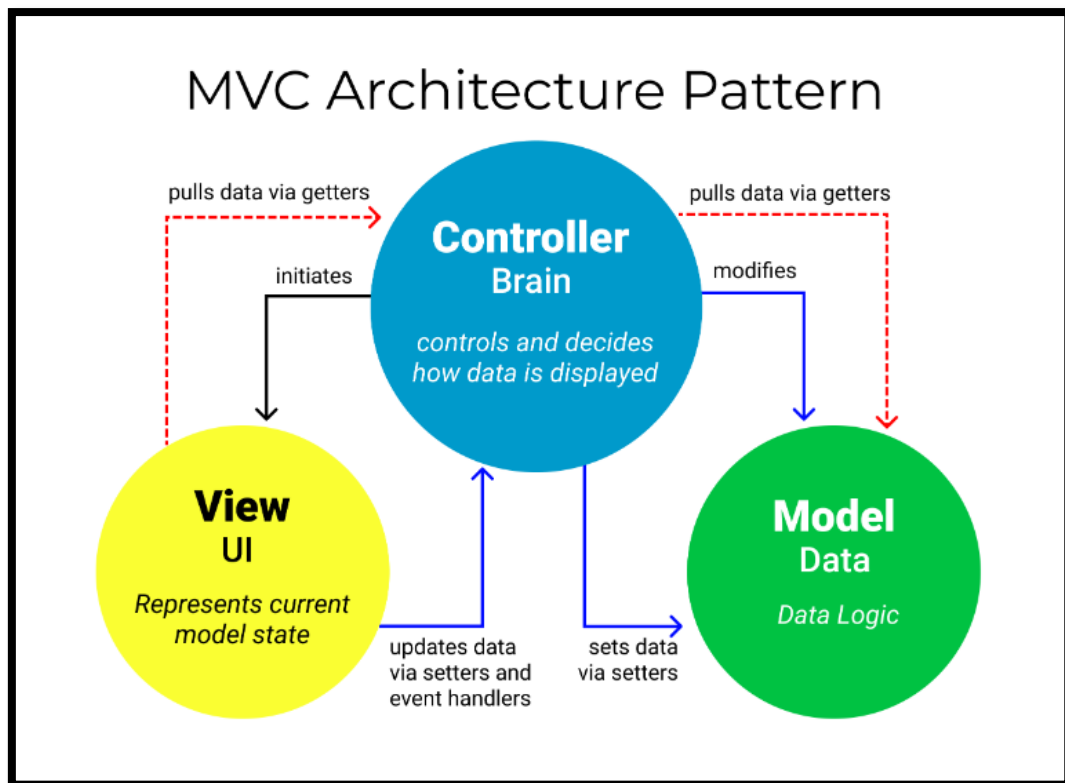


Figure 29

Currently the MVC pattern is used to create modern web applications as it allows the app to be scalable, maintainable, and easy to expand.

6.2.2 My Implementation

6.2.2.1 Models

In terms of my models, I have a model for each table in the database such as Barber.php, Booking.php, Role.php, Services.php, User.php.

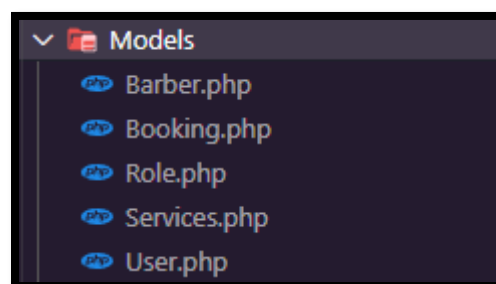


Figure 30

Within each model I define the relationship between each model such as a barber has many bookings, or a service belongs to many barbers. Below is an image of how these relations are defined and implemented in figure 20.

```

public function booking() {
    return $this->hasMany(Booking::class);
}

public function service() {
    return $this->belongsToMany(Services::class)->withTimestamps();
}

```

Figure 31

Above we can see that using the methods `hasMany()` and `belongsToMany()` and calling the appropriate class, I can set the relationships.

6.2.2.2 Views

The views folder is segregated as in there is sub-directory for admin views, as well as user views. This is to enable different functionality for each.

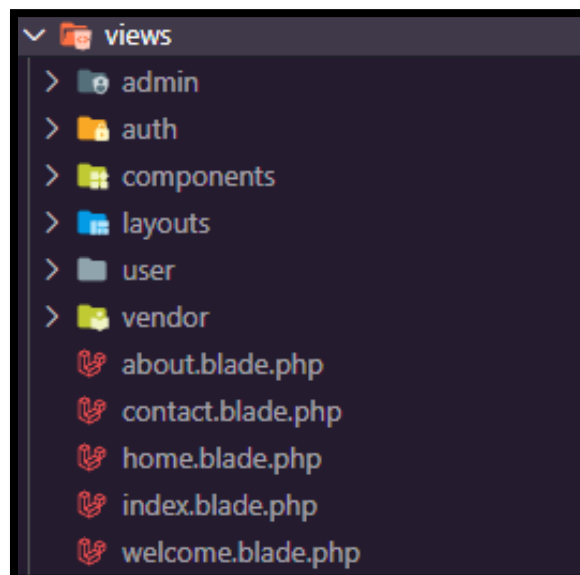


Figure 32

Above in figure 21, we can see a `components` folder which is for my reusable components which may be implemented across all views, a `layouts` folder containing the main `layout.blade.php` that is used with most views in the application.

6.2.2.3 Controllers

The controllers, same as before are segregated into different folders for user and admin as seen below in figure 22.

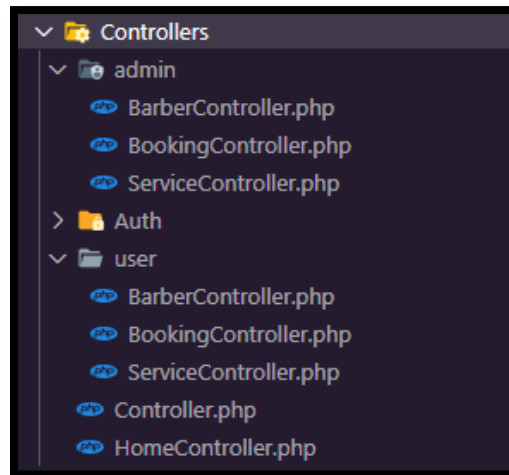


Figure 33

Each of the controllers for the admin contain all the relative CRUD functionality. In each function I declare the type of role the current user logged into the application has and return the appropriate view as seen in figure 23.

```
public function index()
{
    $user = Auth::user();
    $user->authorizeRoles('admin');

    $bookings = Booking::with('barber', 'services')->latest()
        ->simplePaginate(4);

    return view('admin.bookings.index')->with('bookings', $bookings);
}
```

Figure 34

In the above index function, I am returning the authorized user the admin.bookings.index view with all bookings, barbers, and services and paginating the results.

6.3 User Authentication

Laravel provides the tools to implement authentication quickly, securely, and easily. I will explain the few way Laravel achieves this.

1. **User Model:** Laravel offers a standard User model that symbolizes a user in your application.
2. **Auth Routes:** For authentication, Laravel provides pre-built routes including login, logout, and registration.
3. **Password Hashing:** User passwords are hashed and stored in the database by Laravel using bcrypt.
4. **Auth Middleware:** To prevent unauthorized users from accessing certain routes, Laravel has middleware that checks for user authentication.

6.4 Routing

Laravel routing establishes how incoming requests to the application should be handled. Depending on the URL the client requests, it determines which code should be executed. I will explain in detail the image below of my defined routes in the web.php file in figure 24.

```
Route::get('/', function () {
    return view('auth/login');
});

Route::get('/home', [App\Http\Controllers\HomeController::class, 'index'])->name('home');

Route::get('/about', [App\Http\Controllers\HomeController::class, 'about'])->name('about');

Route::get('/contact', [App\Http\Controllers\HomeController::class, 'contact'])->name('contact');

/* Admin routes */
Route::resource('admin/bookings', AdminBookingController::class)->middleware(['auth'])->names('admin.bookings');

Route::resource('admin/services', AdminServiceController::class)->middleware(['auth'])->names('admin.services');

Route::resource('admin/barbers', AdminBarberController::class)->middleware(['auth'])->names('admin.barbers');

/* User Routes */
Route::resource('user/services', UserServiceController::class)->middleware(['auth'])->names('user.services');

Route::resource('user/bookings', UserBookingController::class)->middleware(['auth'])->names('user.bookings');

Route::resource('user/barbers', UserBarberController::class)->middleware(['auth'])->names('user.barbers');

/* This code defines a GET method that allows a user to view the details of a service type from the appointments page. */
Route::get('/services/{id}', [App\Http\Controllers\user\ServiceController::class, 'showService'])->name('user.services.show');

Route::get('/barbers/{id}', [App\Http\Controllers\user\BarberController::class, 'showBarber'])->name('user.barbers.show');

Auth::routes();
```

Figure 35

These routes together set up a set of endpoints for accessing and interacting with the application's resources, such as user authentication, about page, home page create booking, and resource management for both admins and users.

6.5 Templating

Blade is a templating engine that Laravel uses to create views. Blade provides a simple syntax for building templates that enables modular, reusable code.

Below are two code snippets of blade components I created to reuse throughout the application in figure 33 and figure 34.

```
@props(['service'])

<div class="row">
  <h5 class="heading fs-4 card-title">{{ $service->haircut }}</h5>
  <div class="col-sm-12">
    <hr class="border border-danger border-1 opacity-50">
  </div>
  <p class="card-text para">{{ Str::words($service->description, 20) }}</p>
</div>
<div class="row">
  <div class="col-sm-12">
    <hr class="border border-danger border-1 opacity-50">
  </div>
</div>
<div class="col-md-6">
  @if (Auth::user()->hasRole('user'))
    <button class="gradient btn btn-md my-4 mx-2">
      <a class="text-light p-3 text-decoration-none heading fw-normal" href="{{ route('user.services.show', $service) }}">View Details</a>
    </button>
  @else
    <button class="gradient btn btn-md my-4 mx-2">
      <a class="text-light p-3 text-decoration-none heading fw-normal" href="{{ route('admin.services.show', $service) }}">View Details</a>
    </button>
  @endif
</div>
```

Figure 36

This Blade component lists information about a service and has a button that directs users to a page with further information. To specify a parameter for a component that expects a \$service variable to be handed in, use the @props(['service']) directive.

With a heading with the service's name (\$service->haircut), the component then provides a brief explanation of the service (\$service->description). The additional method Str::words() shortens the description to 20 words.

Additionally, the component has a button that, depending on the user's role, directs them to the relevant information page for the service. The button has a border and a gradient the background.

```

{{-- When I call the component, i will pass two parameters, $image & $heading
--}}
<div class="my-background-image" style="background-image: url('{{ asset('images/
' . $image) }}');">
  {{-- The preg_replace is used to make the last word red, done this way to
  be reusable --}}
  <h1 class="display-3 text-center heading text-light">{!! preg_replace('/(\w
+)\z/', '<span class="bg-Text slider-link fw-bold">$1</span>', $heading) !!}
  </h1>
</div>

```

Figure 37

This Blade component shows an image background that is the entire width with a heading text that is centered over it.

The sequence diagram in figure 35 showcasing the system architecture in terms of how the suer navigates through the application to make a booking and then view that booking.

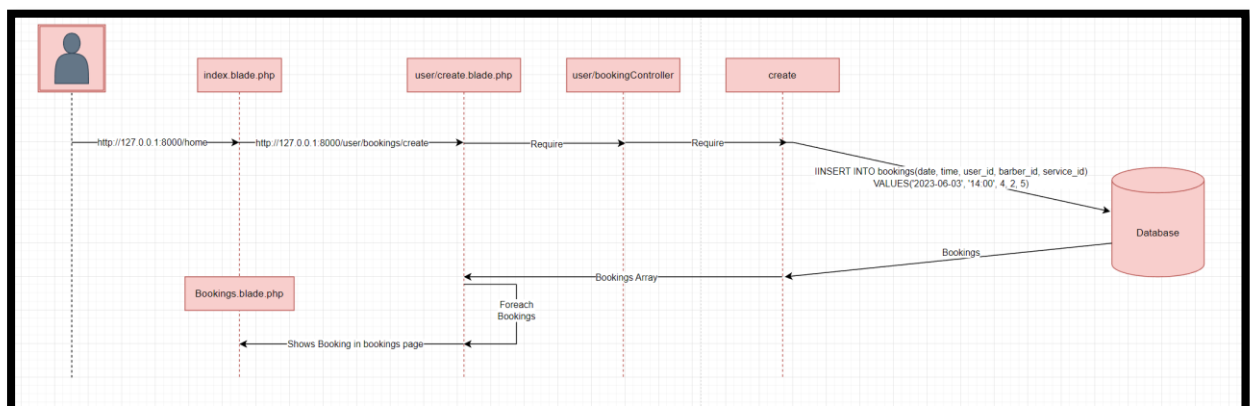


Figure 38

7 Testing

7.1 Introduction

- **Functional Testing:**

Is a type of software testing that evaluates the ability of an application or system by evaluating how it works with a particular set of results. It checks that the system works as expected and meets functional standards.

In the context of Laravel, functional testing involves looking at the functionality of the application's routes and controllers, including how data is entered and outputted.

- **User Testing:**

User testing is a method that evaluates an application or system by watching how actual users interact with it. To improve the functionality and design of the application, it involves gathering feedback and data about user behaviour, preferences, and expectations.

7.2 Functional Testing

To confirm the Barbershop project's various features, I ran a few functional tests on it.

First, perform render page tests to make that every page in the application renders correctly. These tests confirm that the front-end pages rendered successfully. I also ran authentication tests to make sure the application's user authentication functions are working properly. These assessments show clients may log in.

I ran seeding tests, and the results showed that the data seeding process functions as expected. These checks guarantee that the data was correctly put into the database.

- Render Pages
- Authentication
- Seeding

7.2.1 Render Pages

Test No	Description of test case	Input	Expected Output	Actual Output	Comment
1	Application Returns a successful log in page	/	HTTP Status code 200 & page rendering	Page returns with status 200 and page renders	Page renders successfully
2	Application Returns a successful register page	/Register	HTTP Status code 200 & page rendering	Page returns with status 200 and page renders	Page renders successfully
3	Application Returns a successful home page after user is authenticated	/Home	HTTP Status code 200 & page rendering	Page returns with status 200 and page renders	Page renders successfully
4	Application Returns a successful about page after user is authenticated	/About	HTTP Status code 200 & page rendering	Page returns with status 200 and page renders	Page renders successfully
5	Application Returns a successful contact page after user is authenticated	/Contact	HTTP Status code 200 & page rendering	Page returns with status 200 and page renders	Page renders successfully
6	Application Returns a successful services details page after user is authenticated	/services/id	HTTP Status code 200 & page rendering	Page returns with status 200 and page renders	Page renders successfully
7	Application Returns a successful barber details page after user is authenticated	/barbers/id	HTTP Status code 200 & page rendering	Page returns with status 200 and page renders	Page renders successfully

7.2.2 Authentication

Test No	Description of test case	Input	Expected Output	Actual Output	Comment
1	Admin can log in with admin credentials	/Login	Redirected to /home after login form posts	Admin gets redirected to /home	Test successful
2	Admin can log out	/Logout	Redirected to / after logout	Admin gets redirected to /	Test successful
3	User can register	/Register	Redirected to /home	User gets redirected to /home	Test successful
4	User can log in	/Login	Redirected to /home	User gets redirected to /home	Test successful
5	User with incorrect credentials cannot log in	/Login	Error – wont log user in	User gets redirected back to /login	Test successful
6	User can log out	/Logout	Redirected to /	User gets redirected to /	Test successful

7.2.3 Seeding

Test No	Description of test case	Input	Expected Output	Actual Output	Comment
1	Bookings table is seeded with fake data	Check there is 12 bookings	12 bookings found in the database	12 bookings were found	Test successful
2	Barbers table is seeded with fake data	Check there is 4 barbers	4 barbers found in the database	4 barbers were found	Test successful
3	Services table is seeded with fake data	Check there is 8 services	8 services found in the database	8 services were found	Test successful
4	Roles table has 2 roles	Check there is 2 roles	2 roles found in database	2 roles were found	Test successful
5	Users table has users	Check there is 2 users	2 users found when seeded	2 users were found	Test successful

7.2.4 Discussion of Functional Testing Results

Based on the results of the functional testing, the web application seems to be functioning successfully.

The render page tests confirmed that each page loaded successfully. The authentication tests validated that users and administrators could successfully log in and out of the system. Based on the outcomes of the seeding trials, the database is being loaded with the information required to execute the program.

7.3 User Testing

The testing's goals were to evaluate the system's usability and find any areas that may want improvement. Two users had the chance to use the user interface and perform many tasks, including signing up, logging in, creating a booking, editing an existing booking, and cancelling a booking. Using the feedback, the system's user interface was improved, making it more user-friendly and intuitive.

Testing Results - Amy:

- When Amy attempted to register for the application and entered the wrong password, the text was overly wide, and the error notice was in red.
- Even when she attempted to log in but entered the wrong password, the form stayed posted, she was routed to the home page.
- Following the booking, the flash message was too tiny and difficult to see.

When Amy clicked on the bookings page, she could see each fake booking that had been seeded for the admin.

Testing Results - Ellie:

- Password issues on the login/register screens were challenging to comprehend.
- As Ellie moved through the various services, the pagination sent her back to the top of the page.
- The learn more about barber link needs to be displayed better and was not immediately apparent on the about us page.

As a result, extensive bug fixing was undertaken to ensure an intuitive and simple user experience throughout the web application. Various issues that were raised during the user testing phase were fixed such as I added user ownership to the bookings, images, text, clarity, readability were all edited to be more user-friendly.

7.4 Conclusion

Functional and user testing were carried out to ensure that the application functions correctly. The tests helped with finding potential issues and confirmed that the application worked as intended.

On the other hand, user testing was carried out to assess the application's usability and user experience. By improving the application in response to the evaluation of user input, the user experience has been enhanced.

Overall, testing has played a role in making sure the application is usable and functioning.

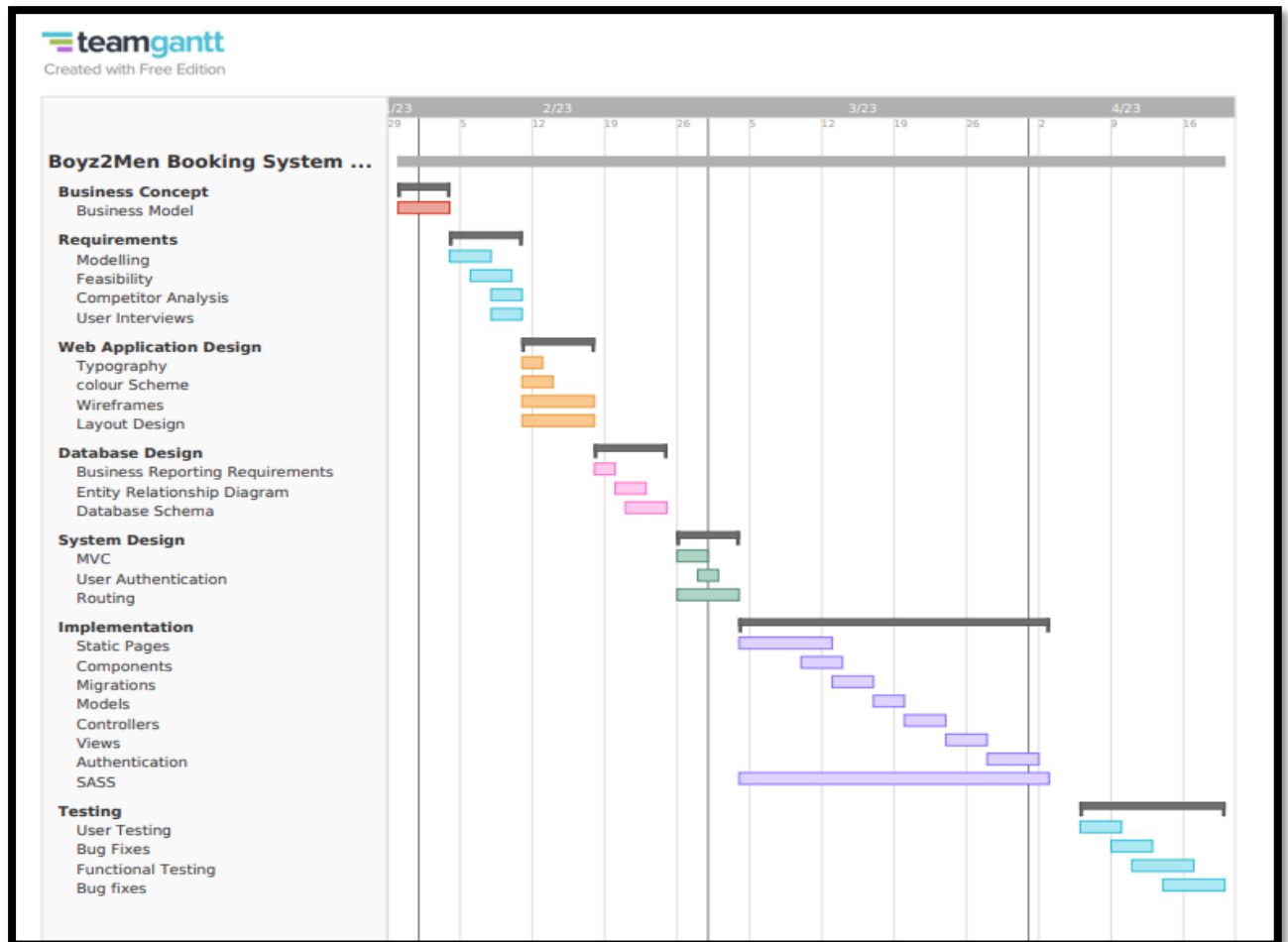


Figure 39

8 Project Management

8.1 Introduction

Agile project management methodologies provide for flexibility and adaptation to changes that might take place throughout the duration of the project. This section outlines each of the several project phases, from the requirements gathering phase through the testing phase.

8.2 Project Phases

To ensure that the project is successfully finished within the set time limit, several phases are often mentioned. The usual phases include planning, gathering requirements, designing, developing, testing.

8.2.1 Requirements

The requirements gathering phase is a necessary part of every software development project since it provides the structure for the rest of the project. The initial phase of the barbershop booking application's development involved a number of tasks, such as competitor analysis, interviews, requirements modelling, feasibility, and functional and non-functional requirements.

8.2.2 Design

During the design process, I focused on creating a user-friendly and visually appealing interface for the barbershop booking system. I used Bootstrap 5 and Sass to create a flexible and dynamic layout. Wireframes were created in Figma to mimic how the user would interact with the application, helping in the representation of the design.

8.2.3 Implementation

During the implementation phase, the actual application was being developed in line with the requirements and design phases. Laravel's models were used to create the relationships between the different things, such as bookings, users, and services. The application now has CRUD (Create, Read, Update, Delete) features for both the user and admin sections. Blade, the templating engine included with Laravel, was used to create specific components.

8.2.4 Testing

To try to make sure that the application worked as intended, a number of methods for testing were used during the project's testing phase. One of the main testing methods used was user testing, in which users were given specified activities to do within the application.

Functional feature testing was also done, which involves testing out various application features including user authentication and page rendering.

8.3 Project Management Tools

8.3.1 GitHub Project

In general, GitHub had a significant role in the development of the barbershop booking system. The process to track the development of my project was greatly improved and made easier by using GitHub. The ERD, database schema, use case diagrams, and documentation like this report and the README.md were all stored on GitHub along with code.

GitHub provides a range of tools and insights that can help in the development of your project, such as providing the percentage of each language used, the number of commits and the most recent commit date, and the overall number of additions and deletions, as seen in the figures below in 27, 28, and 29.

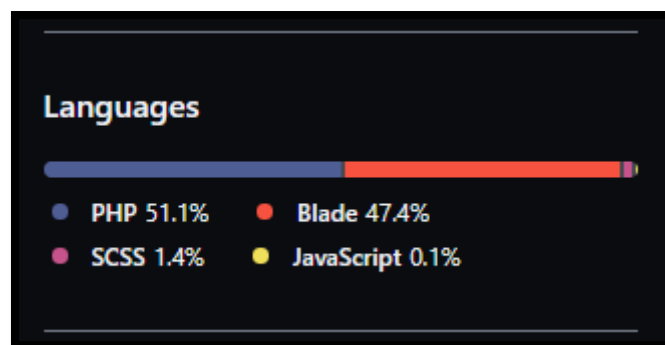


Figure 40

Above we can see the percentage of how much each technology or language was used throughout the project.







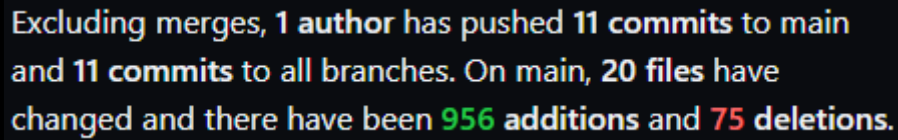
	Liam-Ronan fixed renderPages Test	e9508cb yesterday	54 commits
	Diagrams	added updated version of database schema and ERD	4 days ago
	Documents	Updated section 3 word doc	2 months ago
	References	added database scheme diagram to diagrams folder	2 months ago
	laravel9-Barbershop	fixed renderPages Test	yesterday
	README.md	Update README.md	2 days ago

Figure 41

Here we can see when each folder was last updated, total commits etc.



Excluding merges, 1 author has pushed 11 commits to main and 11 commits to all branches. On main, 20 files have changed and there have been 956 additions and 75 deletions.

Figure 42

Above shows the total additions and deletions throughout the software development lifecycle.

8.3.2 GitHub

Using a website called GitHub, developers may organize, store, and collaborate on code for software projects. It utilizes the Git version control system, which allows programmers to keep track of changes made to their codebase over time. On GitHub, users can create repositories, which are simply folders with code files and other resources. After that, users may invite others to the repository so they can create new branches, make changes to the codebase, and submit pull requests.

9 Reflection

9.1 Your views on the project

I am quite happy with how the project came out overall. I was able to complete the application's major functionality and get it where I wanted it to be with a lot of work, attention, and focus. The usage of multiple diagrams and models, including the use case diagram, ERD, and database schema, was one of the things that really helped me in this project as well as the Figma wireframe designs. I was able to understand the needs of the application and how the various components fit together due to these visuals. Additionally, I was able to gain important information from the questionnaire and the interviews, which enabled me to develop an application that genuinely fulfilled the needs of its users. Overall, I believe the project was a success.

9.2 How could the project be developed further?

The application may be expanded in the future to include an online store where users may purchase grooming and hair supplies. It would be necessary to develop a checkout process, a shopping cart, and a database to keep track of stocks, prices, and product details. To manage online transactions securely, a payment system would also need to be

implemented. Customers who would prefer to buy goods online rather than in stores would benefit from the convenience as well as the rise in revenue.

9.3 Assessment of your learning.

Throughout the course of this project's development, I learned quite a bit about the different technologies and processes used for this software development project. Firstly, I've become much more proficient with Laravel. The application's user interfaces were designed in a way that is both aesthetically appealing and easily adaptable because to my improved skills with Sass and Bootstrap 5.

9.4 Completing a large software development project

Finally, I've learned how important consistency and discipline are to a large software project. I began to understand how important it is to have a particular plan in place and to stick with it and assess and change it as needed. I've learned a lot about the software development lifecycle and the value of project management skills by completing a major software project.

9.5 Technical skills

- **Laravel:** I am now quite proficient with Laravel. This involves understanding of Eloquent ORM, migrations, middleware, routing, and MVC design.
- **Sass and Bootstrap 5:** I am skilled at applying Sass and Bootstrap 5 to layout my web application. These covers understanding of custom CSS, forms, buttons, forms, and responsive design.
- **PHP:** I improved my ability to write clear, effective, and efficient PHP code and gained an improved understanding of the PHP syntax.
- **Database relationships:** I now have better knowledge of foreign keys, constraints, ER modeling, database normalization, and database relationships.

9.6 Further competencies and skills

I improved both my technical and soft skills throughout the course of this project, both of which will be helpful in the future. Initially, I enhanced my time management skills by making sure that I followed the project's plan and met all of my deadlines. Identifying and overcoming issues that occurred during the project, I have improved my problem-solving skills.

10 References

- The 2022 sprout social index: Social Media Trends for the UK & Ireland. Sprout Social. (2022, October 26). Retrieved February 2, 2023, from <https://sproutsocial.com/insights/data/uk-ireland-social-media-trends-2022/>
- Patoli, Z. (2022, November 8). Council post: Why email marketing is still relevant in 2022. Forbes. Retrieved February 2, 2023, from <https://www.forbes.com/sites/forbesagencycouncil/2022/07/28/why-email-marketing-is-still-relevant-in-2022/?sh=5af1da4527cf>
- *The PHP framework for web artisans*. Laravel. (n.d.). Retrieved April 14, 2023, from <https://laravel.com/>
- *17 benefits of Laravel Framework*.- IC Studio, Україна. (n.d.). Retrieved April 14, 2023, from <https://icstudio.online/en/post/17-benefits-laravel-framework>
- Hernandez, R. D. (2021, April 20). *The model view controller pattern – MVC architecture and Frameworks explained*. freeCodeCamp.org. Retrieved April 14, 2023, from <https://www.freecodecamp.org/news/the-model-view-controller-pattern-mvc-architecture-and-frameworks-explained/>