# C:\Users\Messom\Desktop\Monash.jpg

# FIT5032 Design Report

# Major Application Development

# High Distinction

# MedConnect

Edward Shen:30594863

**Your design report must include the following (Pass Level and Above):**

1. **Web Application Title and Description**

MedConnect

1. **User stories and Use case diagram**

**Use Case Diagram:**

Diagram

Description automatically generated

**User Stories:**

1. As an Admin I would like to Create a Staff Account to management my employees.
2. As an Admin I would like to Add my business’s location details into my account by marking geo-locations (search & show location) so staff & clients can see where to come for their appointments.
3. As an Admin or Client, I would like to be able to register a new account for myself so that I can have an account to manage.
4. As an Admin I would like to manage my search, sort & limit by tables for location information & users so that I can quickly find the user & business location data I want.
5. As an Admin I would like to send a bulk email with an attachment to all of my staff members working for a business make a big announcement.
6. As an Admin I would like to find the route between two locations when creating a location so that I can find the distance & shortest paths between two locations
7. As an Admin I would like to enable geolocation to see where my location is relative to the business locations I am about to create.
8. As a Patient I would like to Login & Authenticate to my account so that I can access my information
9. As a user I would like to view my appointments in a calendar interface to see when and where my appointments are.
10. As a Staff I would like to Book appointments so that I can know when and where to meet up with a client.
11. As a Staff I would like to Cancel appointments so that I can reschedule my own & my client’s time.
12. As a Staff/Client I would like to see my Appointments in an organized fashion (best Calendar) to find when and where my appointments are.
13. As a Staff/Client I would like my Appointments to never overlap/conflict with one another so that I don’t accidentally book two appointments where I would have to skip one.
14. As a Staff/Client I would like to be notified every time that I book an appointment via email, so I don’t lose track of appointments when not using the application.
15. **Block/Functional diagram**

Diagram

Description automatically generated

Diagram

Description automatically generated Diagram

Description automatically generated

Diagram

Description automatically generated

1. **Your selected approach when constructing the application.**

I chose to go with a Code First approach. There was for no reason aside from the fact that I felt as though it was a lot more flexible and gave me more control than the other approaches. In addition to this

**Additional Distinction Level (the above and the following)**

1. **Class Diagram or Entity Relation Diagram**

Diagram

Description automatically generated

1. **Data dictionary**

Appointments Table

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Nullable | Description |
| Id | nvarchar(128) | False | Id of the Appointment |
| ClientUserId | nvarchar(MAX) | True | Appointment’s Client |
| StaffUserId | nvarchar(MAX) | True | Appointment’s Staff Member |
| AppointedDateTimeStart | datetime | True | Time when the appointment starts |
| AppointedDateTimeEnd | datetime | True | Time when the appointment end |
| AppointedLocationName | nvarchar(MAX) | True | Location of the appointment |
| AppointedLocationLat | real | True | Lat Coordinate of the appointment |
| AppointedLocationLong | real | True | Long Coordinate of the appointment |
| Description | nvarchar(MAX) | True | Text Describing the purpose of the appointment |

AppointmentUsers Table

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Nullable | Description |
| Id | nvarchar(128) | False | Id of the AppointmentUser |
| AppointmentId | nvarchar(MAX) | True | Id of the corresponding appointment |
| UserId | nvarchar(MAX) | True | User involved in the appointment |
| UserRole | nvarchar(MAX) | True | User’s corresponding role in the appointment |

AspNetRoles Table

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Nullable | Description |
| Id | nvarchar(128) | False | Id of the Role Type |
| Name | nvarchar(256) | False | Name/Title of the Role (e.g Admin, Staff, User) |

AspNetRoles is used in case future roles need to be introduced, we can just scale using this table.

AspNetUserRoles

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Nullable | Description |
| UserId | nvarchar(128) | False | User with the assigned role |
| RoleId | nvarchar(128) | False | Role Type Id (pointing to AspNetRoles) |

AspNetUsers Table

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Nullable | Description |
| Id | nvarchar(128) | False | Id of the User |
| Email | nvarchar(256) | True | Email of the User |
| EmailConfirmed | bit | False | Whether the Email has been confirmed or not |
| PasswordHash | nvarchar(MAX) | True | Field used to store the password |
| SecurityStamp | nvarchar(MAX) | True |  |
| PhoneNumber | nvarchar(MAX) | True |  |
| PhoneNumberConfirmed | bit | False |  |
| TwoFactorEnabled | bit | False |  |
| LockoutEndDateUtc | datetime | True |  |
| LockoutEnabled | bit | False |  |
| AccessFailedCount | int | False |  |
| UserName | nvarchar(256) | False | User’s Name when using the web application. |

Note: The undescribed fields. I do not directly use in my application.

BusinessEmployees

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Nullable | Description |
| Id | nvarchar(128) | False | Id of the Employee Assignment |
| BusinessId | nvarchar(MAX) | True | Business the employee belongs to |
| UserId | nvarchar(MAX) | True | The user that is being assigned within the business |
| Job | nvarchar(MAX) | True | Job Title |

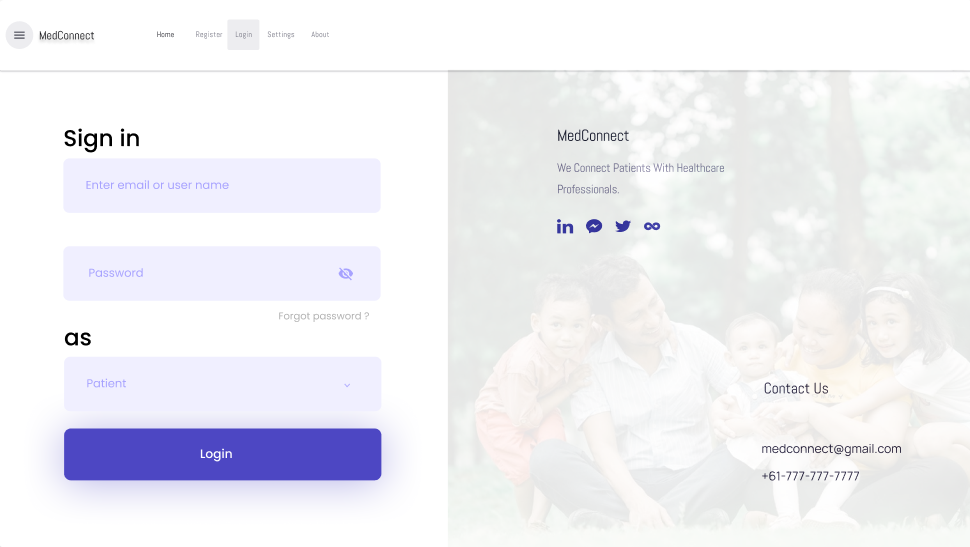
Businesses

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Nullable | Description |
| Id | nvarchar(128) | False | Id of the business within the application |
| AdminId | nvarchar(MAX) | True | The Admin User’s Id |
| BusinessName | nvarchar(MAX) | True | Business’s Name |
| RatingAggregateTotal | real | False | Rating Aggregate Total Sum |
| RatingCount | int | False | Total Amount of times it was rated |
| RatingAverage | real | False | Average Rating |
| RatingTop | real | False | Highest rating out of 5 ever made. |

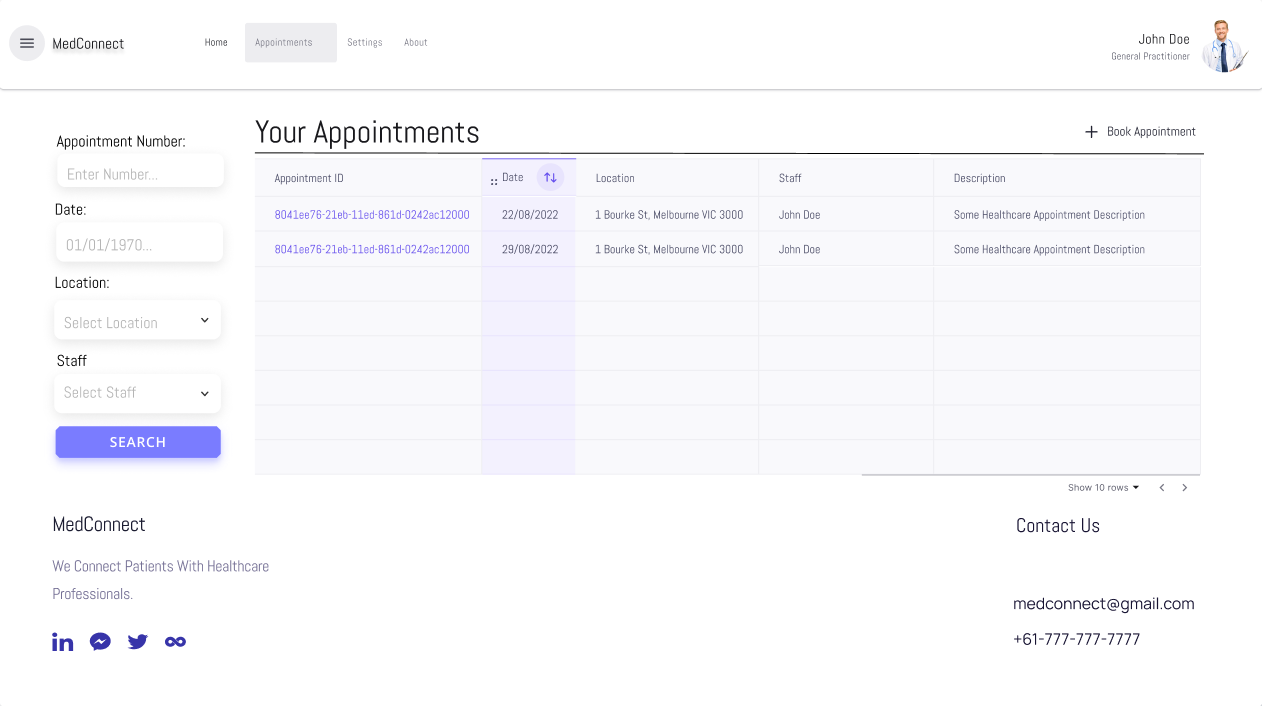
BusinessLocations

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Nullable | Description |
| Id | nvarchar(128) | False | Business Location Id |
| BusinessId | nvarchar(MAX) | True | Id of the business the location belongs to. |
| BusinessLocationName | nvarchar(MAX) | True | Location’s name |
| BusinessLocationLat | real | False | Location’s Lat Coordinate |
| BusinessLocationLong | real | False | Location’s Long Coordinate |

1. **Mockup prototypes and implementation with user registration and authentication**



*Figure 3: Mock-up for Role based Authentication (BR B.1)*



*Figure 4: Mock-up for Interactive Table Data (BR C.3)*



*Figure 5: Mock-up for Geo-Location Map to Record down Appointment locations (BR E.1)*

The default controller providing authentication in my current project is AccountController. Link:

<https://github.com/3drdsh3in/FIT5032/blob/master/assignments/portfolio/FIT5032_Assignment_Portfolio/FIT5032_Assignment_Portfolio/Controllers/AccountController.cs>

Graphical user interface, application, website

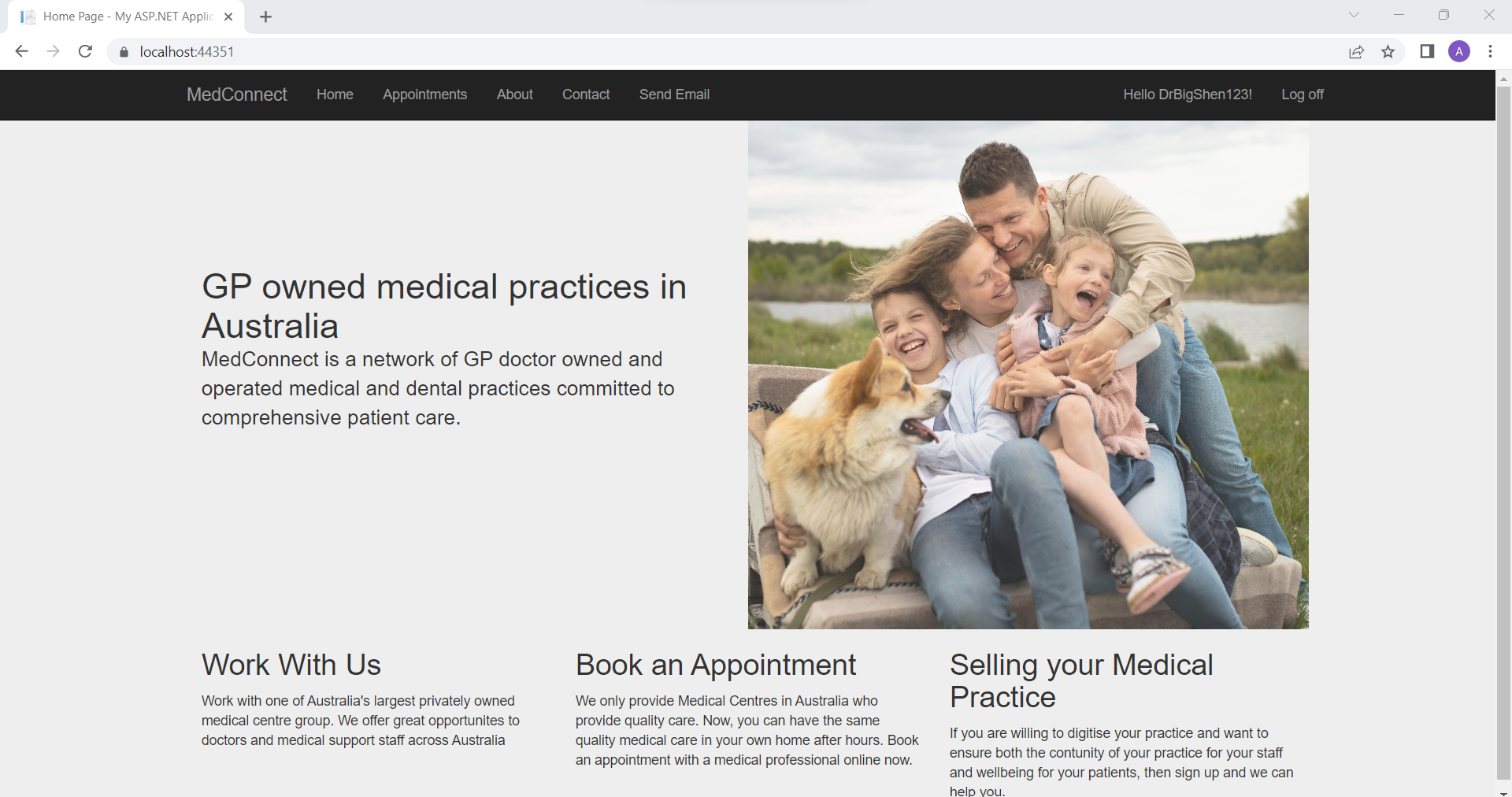
Description automatically generated

*Figure 2: Registration Form*

Graphical user interface, application

Description automatically generated

*Figure 3: Logging In*



*Figure 4: Log in Screen upon clicking Create or Login.*

1. **Usability Design Review**

I tried applying Donald Norman’s 6 Principles of Design through the implementation of this project.

Visibility Principle:

This principle refers to how well users using the application’s user interface can figure out or know what options to interact with. In MedConnect the navigation bar always keeps all the primary business functions readily available directly to the user. The Black & White Text aim to make the interface simple & avoid complexity (Admin User screenshot options below)



Feedback Principle:

Feedback is how well a system provides user with information regarding what has been done/completed. In MedConnect, an instance where this was heavily applied was around validation of forms (To notify users of completion, errors etc.). An Example can be seen below after attempting to unsuccessfully login.

Graphical user interface, application, Teams

Description automatically generated

The detailed error message tells the user that provides feedback that specifically tells the user that the Role ‘Client’ does not belong to the given login user inputted. For the most part I think this is well done in my application as there is a lot of validation.

Another Example with Sending Emails:

Before Send:

Graphical user interface

Description automatically generated

After Send (Note the message at the top):

Graphical user interface

Description automatically generated

Affordance Principle:

This principle refers to how intuitive elements of the UI are. In MedConnect Affordance has been applied when creating a new location for a given business.

**Graphical user interface, website, map

Description automatically generated**

As we can see, the map widget for entering location is very intuitive and when moving between locations tells the user that they can drag/swipe it to change the location.

Mapping Principle:

This principle refers to how close the application’s relationships between control & effect are. In MedConnect, I tried to implement this by included the use of icons.

Calendar

Description automatically generated

For Example, the ‘+’ sign just left of the button for creating a new appointment. Here In this case (I would like to think) there is a clear mapping as the icon intuitively tells the users the fact, they are pressing a link that enables them to create a new appointment.

Constraints Principle:

This principle tells us how well, we try to limit the range of possible interactions with the UI to guide the user in the correct direction when navigating the UI. In MedConnect, I tried implementing this via simpler login/registration form.

Graphical user interface, application, Teams

Description automatically generated

Notice, you can only interact with about 5 things (Email Input, Password Input, Role Type Select, Login Button & Login Here) when attempting to login directly. This is aimed to avoiding confusion for the user. By keeping it simple, the finiteness of my UI meets this principle well and avoid confusion.

Consistency Principle:

Refers to how consistent & uniform the operations & visual elements are for achieving similar tasks. I applied this principle by reusing UI components that performed very similar operations such as Registration & Login as follows:

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Notice how the login & registration forms are for the most part uniform and almost identical (albeit with a few minor text changes). This consistency (while a bit bland to look at) confuses the user a lot less.

For the most part, I think that MedConnect does a fine job at being clean and usable. However, there are still flaws in the design as the user interface still does not meet some of the 6 Design Principles (by Donald Norman) all that well. For Example, there is a bit of a contrast when interacting with buttons in the web application. In some places they are a blue color, in

**Additional High Distinction Level (the above and the following)**

1. **Development Methodology**

I was really hoping to use a Test-Driven Development (TDD) approach, however due to my unfamiliarity with the C# & ASP.NET ecosystem I had to abandon it for the most part and approach development using the Code & Fix model. Essentially, all the QA that I had done for this assignment was via manual QA (No Unit, Component, Blackbox tests, with test suites & ci automations despite having a fondness for them)

1. **Versioning**

For this assignment git was used, I found there was no need for a more sophisticated git workflow than to simply push all my changes directly to a singular master/main branch (As no pair or team programming was really needed). Additionally, the pushes I made to the repository were simply “bulk” commits (lots of features & fixes at once) and were not really divided based on features, tasks or fixes (again there was no need to do so since I was not working in a team, so I chose to keep it easy for myself).

<https://github.com/3drdsh3in/FIT5032/tree/master/assignments>

1. **Innovation and Research**

N/A didn’t complete.