National College of Ireland

BSc (Honours) in Computing

Software Development

2020/2021

Anthony Cuddihy

X17103924

X17103924@student.ncirl.ie

Live Medical
Technical Report

Contents

Executive Summary

In this report ill be covering the technologies I used to develop my application and also how I achieved my final result.

1.0 Introduction

1.1. Background

With COVID-19 affecting the world it has become even more apparent that we need a faster, easier and more covenant way to be seen by a doctor. Web videos would facilitate this requirement allowing people to have a video call with their doctor. This web app will also help people how are afraid to go to doctors due to the pandemic or people who may my physically unable to go. These Videos can then be re watched if required.

Being able to login to a website and quickly see if a doctor is free to see you or being able to see when you doctor is next available will be massive for people who would like to manage their time.

My website will also have the ability to view your past medical reports. Having all previous appoints information in one convenient place. Along with this functionality will be the ability to see your family's hereditary diseases if the family members approve you viewing it.

I myself suffered with a hereditary disease that took almost 2 years to be diagnosed with. If my doctor were able to quickly see a list of health complications that I may be susceptible to I could have been diagnosed earlier.

1.2. Aims

My Aim is to build an application that will allow patients to schedule web meetings with doctors. Once the meetings are scheduled, they will be able to attend a video meeting together.
The patients will also be able to tag their profiles with hereditary diseases so they can easily share them with family members.

The application itself should be minimal and intuitive to use for anybody.

1.3. Technology

The front end will be built in rails using HAML, CSS, JavaScript, and bootstrap. I chose these technologies as they are powerful and used by many in the industry. They also have substantial amounts of documentation available if I do have issues during development.

The backend will also be using rails along with AWS cloud storage using Postgres. These were choosing as they are used by a lot of developers, and I wanted to make myself familiar with the technology.

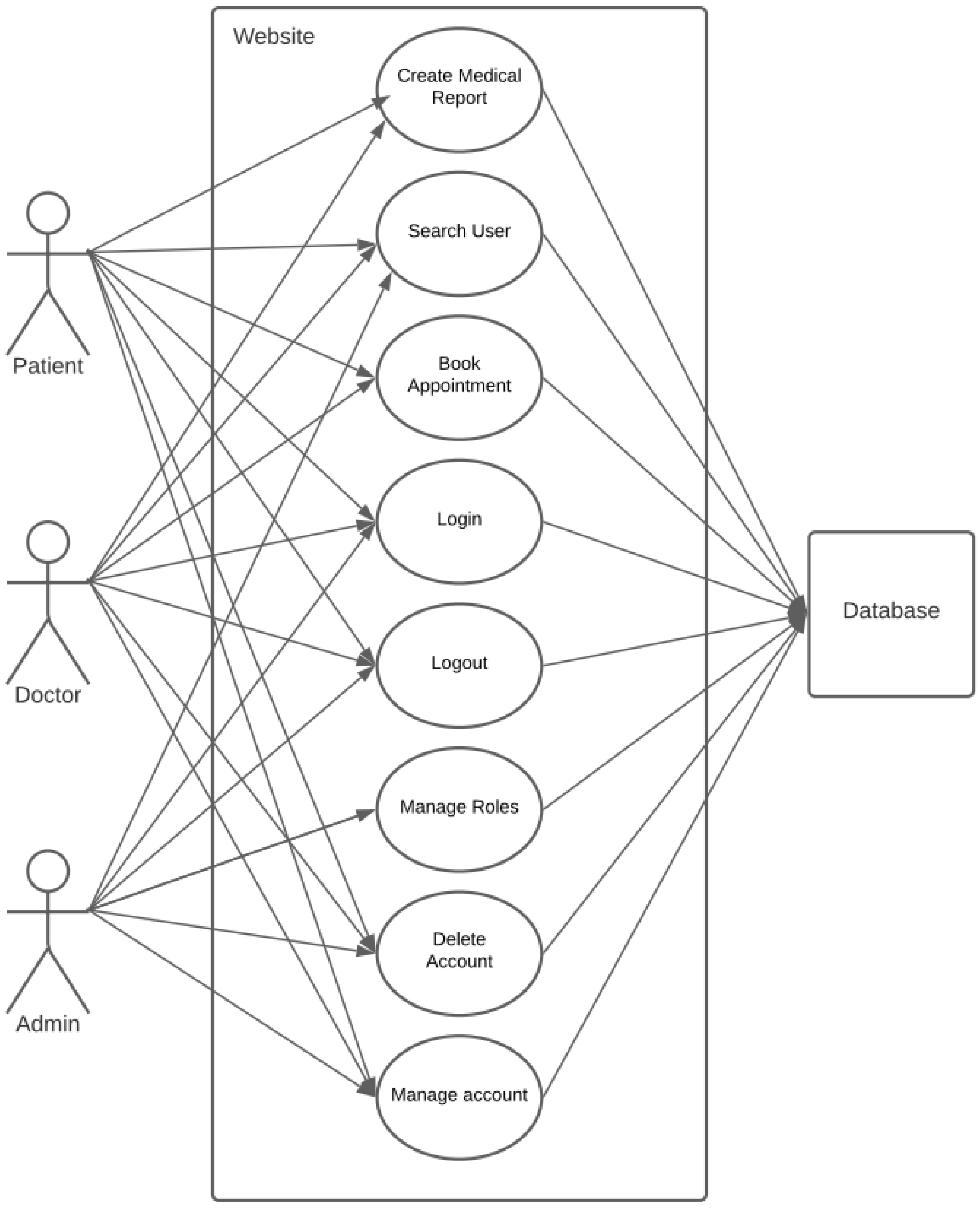
The last way I will be using technology in my application will be with Amazons Cloud9 Development Environment. This environment was setup to allow me to have a development environment and a production environment. The production environment was connected though GitHub that automatically built a new version of the application when I pushed my tested changes from my development environment to the production repository.

2.0 System

2.1. Requirements

2.1.1. Functional Requirements

2.1.1.1. Use Case Diagram



2.1.1.2. Requirement 1 <Create Account >

Description

The requirement is for all three types of users to be able to go to the website and create a new account.

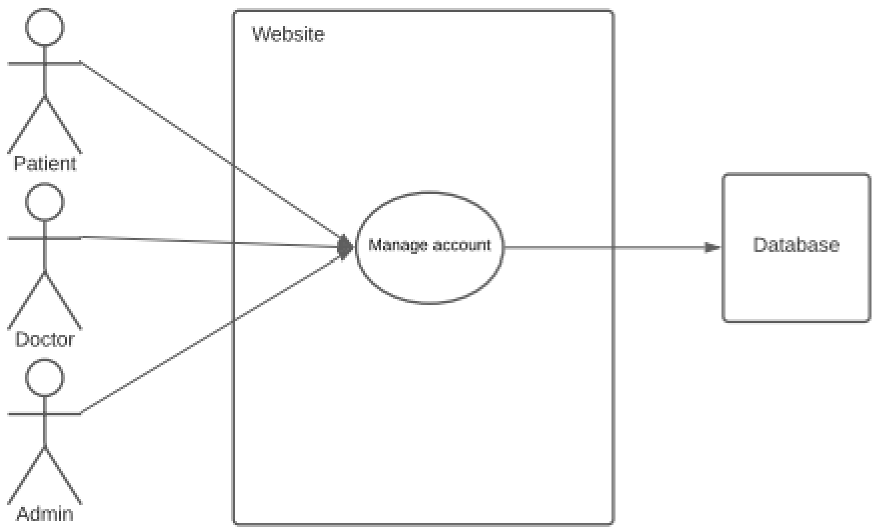
Use Case

UC-1

Scope

The scope of this use case is to show the process of a new user signing up to the website.

Use Case Diagram



Flow Description

Precondition

User is on website and wants to make an account.

Activation

This use case starts when a <patient>, <doctor> or <admin> wants to create an account.

Main flow

1. User is on the registration page.
2. The patient>, <doctor> or <admin> inputs the details required to register.
3. The system sends the data to the database.
4. The patient>, <doctor> or <admin> has made an account.

Alternate flow

1. User is on the registration page.
2. The patient>, <doctor> or <admin> inputs the details required to register.

3. The system sends the data to the database.
4. Database denies creation due to incorrect details or account already in database.

Exceptional flow

1. The <patient>, <doctor> or <admin> rails to complete registration.
2. Account not created.

Termination

The system presents the message “account created”.

Post condition

The system goes into a wait state.

2.1.1.3. Requirement 2 <Login >

Description

The requirement is for all three types of users to be able to go to the website and login.

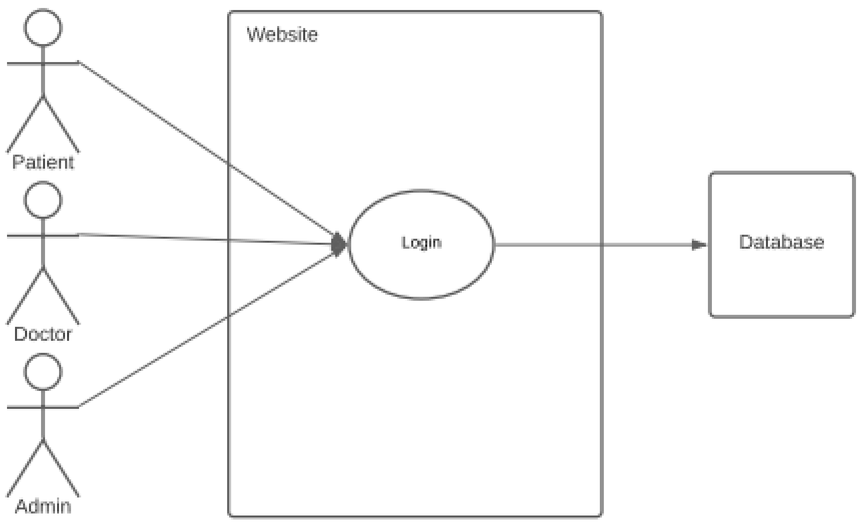
Use Case

UC-2

Scope

The scope of this use case is to show the process of a user logging into the website.

Use Case Diagram



Flow Description

Precondition

User is on website and wants to login after they have successfully made an account.

Activation

This use case starts when a <patient>, <doctor> or <admin> wants to login to their account.

Main flow

1. User is on the login page.
2. The patient>, <doctor> or <admin> inputs the details required to login.
3. The system sends the data to the database.
4. The patient>, <doctor> or <admin> has logged in.

Alternate flow

1. User is on the login page.
2. The patient>, <doctor> or <admin> inputs the details required to login.
3. The system sends the data to the database.
4. Database denies login due to incorrect details.
5. Error displayed to user.

Exceptional flow

The <patient>, <doctor> or <admin> fails to complete login.
Termination

The system presents the message “login successful”.

Post condition

The system goes into a wait state.

2.1.1.4. Requirement 3 <Logout >

Description

The requirement is for all three types of users to be able to logout from the website.

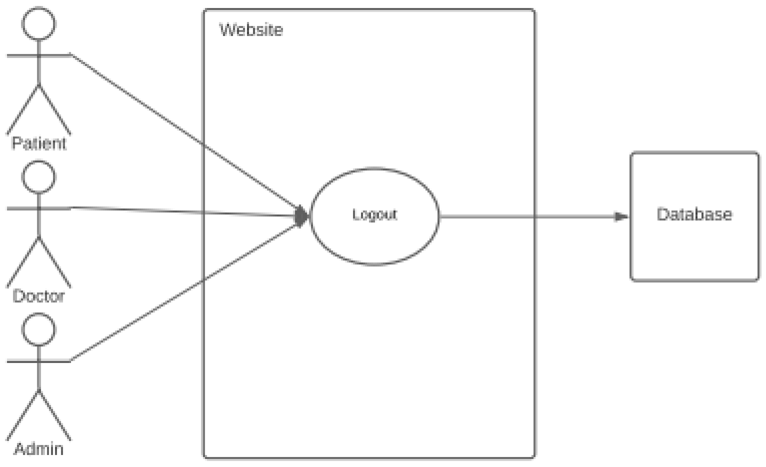
Use Case

UC-3

Scope

The scope of this use case is to show the process of a user login out from the website.

Use Case Diagram



Flow Description

Precondition

User is on website and wants to logout.

Activation

This use case starts when a <patient>, <doctor> or <admin> wants to logout from their account.

Main flow

1. User is on the logged in on the website.
2. The <patient>, <doctor> or <admin> clicked the logout button.
3. The system sends the data to the database.
4. The <patient>, <doctor> or <admin> has logged out.

Alternate flow

1. User is on the logged in on the website.
2. The <patient>, <doctor> or <admin> clicked the logout button.
3. The <patient>, <doctor> or <admin> denies the confirmation of logging out.
4. User remains logged in.

Exceptional flow

The <patient>, <doctor> or <admin> fails to complete logout.

Termination

The system presents the message “logout successful”.

Post condition

The system goes into a wait state.

2.1.1.5. Requirement 4 <Create Medical Report>

Description

The requirement is for <patient> and the <doctor> to be able to create medical reports.

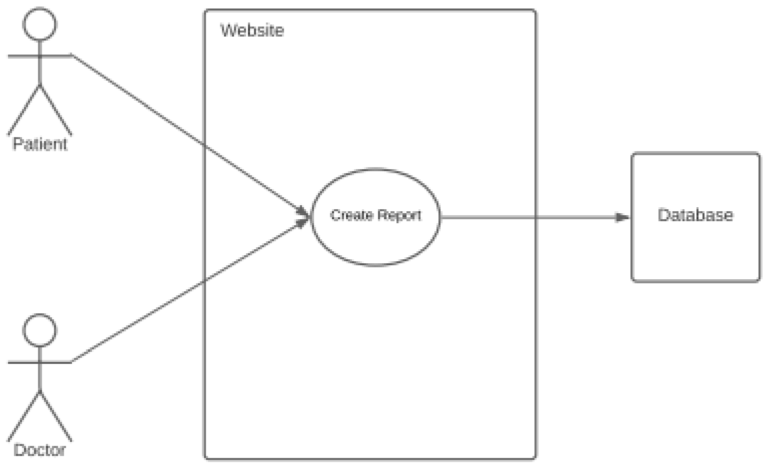
Use Case

UC-4

Scope

The scope of this use case is to show the process of a user creating a report.

Use Case Diagram



Flow Description

Precondition

User is on website and wants to make a medical report.

Activation

This use case starts when a <patient> or <doctor> to make a medical report.

Main flow

1. User is on the logged in on the website.
2. The <patient> or <doctor> clicked the create report button.
3. The <patient> or <doctor> files in the report.
4. The <patient> or <doctor> clicks submit.
5. The system sends the data to the database.
6. The system confirms the record has been added with a success message.

Alternate flow

1. User is on the logged in on the website.
2. The <patient> or <doctor> clicked the create report button.

3. The <patient> or <doctor> fills in the report.
4. The <patient> or <doctor> clicks submit.
5. The system sends the data to the database.
6. The system denies create of record due to incorrectly filled out report.

Exceptional flow

The <patient>, <doctor> or <admin> fails to complete report creation.

Termination

The system presents the message “Report created successful”.

Post condition

The system goes into a wait state.

2.1.1.6. Requirement 5 <Create Appointment>

Description

The requirement is for <patient> and the <doctor> to be able to create medical appointments.

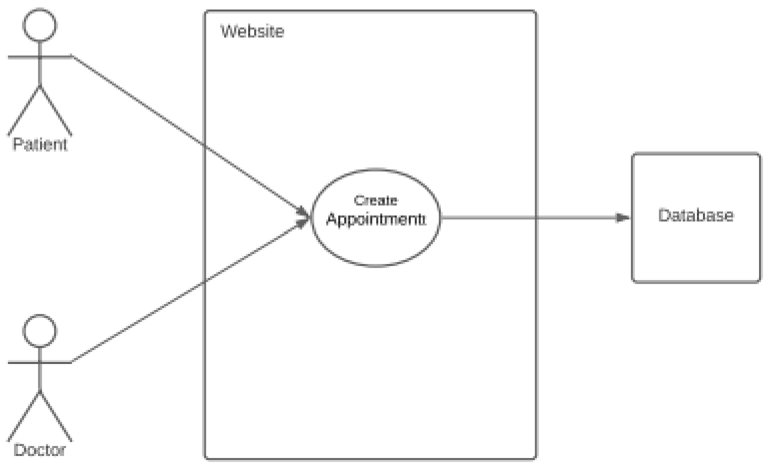
Use Case

UC-5

Scope

The scope of this use case is to show the process of a user creating an appointment.

Use Case Diagram



Flow Description

Precondition

User is on website and wants to create an appointment in the system.

Activation

This use case starts when a <patient> or <doctor> to make an appointment in the system.

Main flow

1. User is on the logged in on the website.
2. The <patient> or <doctor> clicked the create appointment button.
3. The <patient> or <doctor> fills in the appointment.
4. The <patient> or <doctor> clicks submit.
5. The system sends the data to the database.
6. The system confirms the appointment has been added with a success message.

Alternate flow

1. User is on the logged in on the website.

2. The <patient> or <doctor> clicked the create appointment button.
3. The <patient> or <doctor> files in the report.
4. The <patient> or <doctor> clicks submit.
5. The system sends the data to the database.
6. The system denies create of the appointment due to incorrectly filled out appointment.

Exceptional flow

The <patient>, <doctor> or <admin> fails to complete appointment creation.

Termination

The system presents the message "Appointment created successful".

Post condition

The system goes into a wait state.

2.1.1.7. Requirement 6 < Search User >

Description

The requirement is for <patient>, <doctor> and the <admin to be able to search for users in the system they have access to.

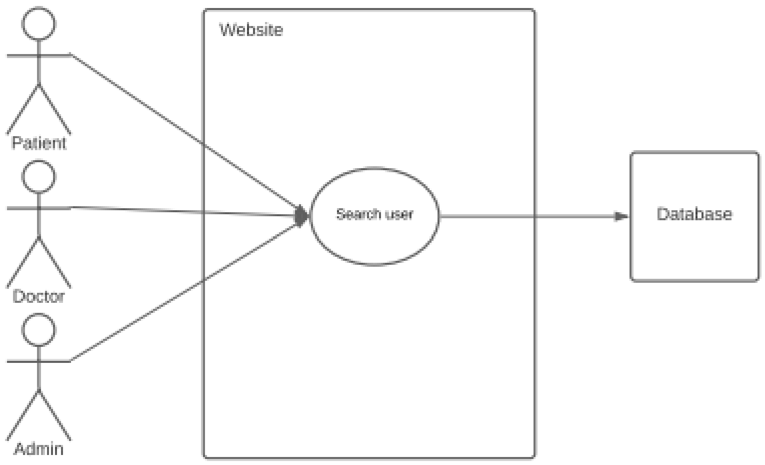
Use Case

UC-6

Scope

The scope of this use case is to show the process of a searching a user in the system.

Use Case Diagram



Flow Description

Precondition

User is on website and wants to find a user in the system.

Activation

This use case starts when a <patient>, <doctor> or <admin> wants to find a user in the system.

Main flow

1. User is on the logged in on the website.
2. The <patient>, <doctor> or <admin> clicked the user's button.
3. The <patient>, <doctor> or <admin> fills in the search field.
4. The <patient> or <doctor> clicks submit.
5. The system sends the data to the database.
6. The system the searched user with a success message.

Alternate flow

1. User is on the logged in on the website.

2. The <patient>, <doctor> or <admin> clicked the user’s button.
3. The <patient>, <doctor> or <admin> fills in the search field.
4. The <patient> or <doctor> clicks submit.
5. The system sends the data to the database.
6. The system returns error as no user was found in database.

Exceptional flow

The <patient>, <doctor> or <admin> fails to complete the search for a user.

Termination

The system presents the searched for user.

Post condition

The system goes into a wait state.

2.1.1.8. Requirement 7 < Delete Account>

Description

The requirement is for <patient>, <doctor> and the <admin to be able to delete their own account.

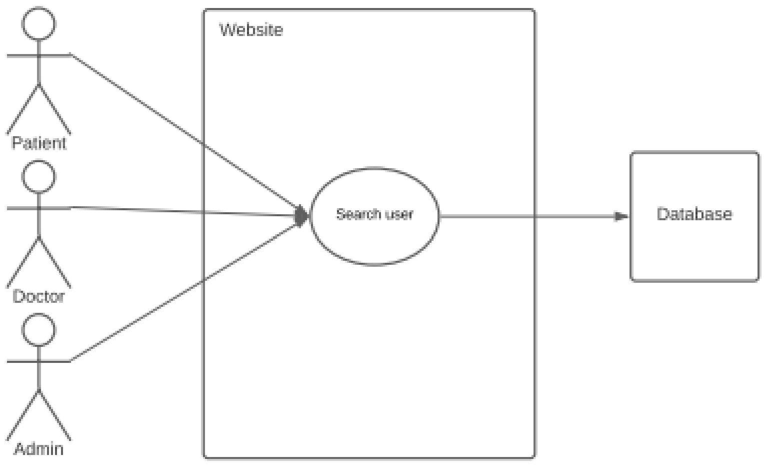
Use Case

UC-7

Scope

The scope of this use case is to show the process of deleting an account from the website.

Use Case Diagram



Flow Description

Precondition

User is on website and wants to delete the account from the website.

Activation

This use case starts when a <patient>, <doctor> or <admin> wants to delete their account.

Main flow

1. User is on the logged in on the website.
2. The <patient>, <doctor> or <admin> clicked the cancel my account button.
3. The <patient>, <doctor> or <admin> confirms the delete request.
4. The system sends the data to the database.
5. The system deletes all data related to the user from the system.

Alternate flow

1. User is on the logged in on the website.
2. The <patient>, <doctor> or <admin> clicked the cancel my account button.
3. The <patient>, <doctor> or <admin> denies the delete request.
4. The system returns the user to the home page.

Exceptional flow

The <patient>, <doctor> or <admin> fails to complete the account delete action.

Termination

The system presents user with a message confirming account is deleted.

Post condition

The system goes into a wait state.

2.1.1.9. Requirement 8 < Manage Roles>

Description

The requirement is for <admin > to be able to update user permissions/ roles on the website.

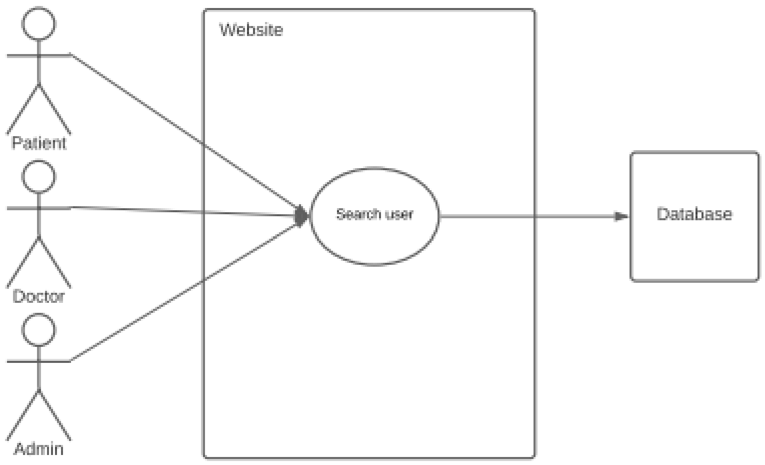
Use Case

UC-8

Scope

The scope of this use case is to show the process of an admin updating a user's access.

Use Case Diagram



Flow Description

Precondition

User is on website and wants to update a user's access.

Activation

This use case starts when a <admin> wants to update a user's access.

Main flow

1. User is on the logged in on the website.
2. The <admin> clicked the update users access button.
3. The <admin> input he changes.
4. The system sends the data to the database.
5. The system replies confirming access was updated.

Alternate flow

1. User is on the logged in on the website.
2. The <admin> clicked the update users access button.
3. The <admin> input he changes.
4. The system sends the data to the database.
5. The system replies denying the request as it was not submitted correctly.

Exceptional flow

The <admin> fails to complete the account permission change.

Termination

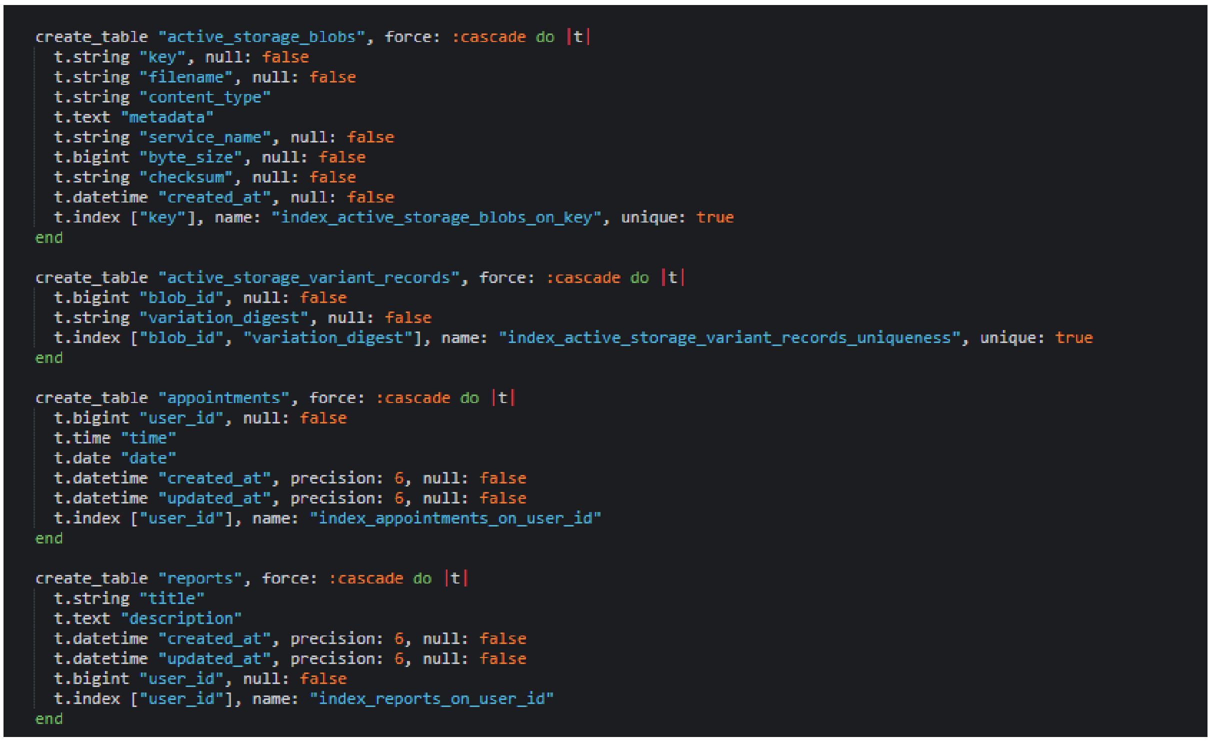
The system presents user with a message confirming account has been updated.

Post condition

The system goes into a wait state.

2.1.2. Data Requirements

Database was created in rails using PostgreSQL as the database engine.
The database migration and schema are found in the DB folder in my projects directory. Below is a small snip of my schema.



```
create_table "active_storage_blobs", force: :cascade do |t|
  t.string "key", null: false
  t.string "filename", null: false
  t.string "content_type"
  t.text "metadata"
  t.string "service_name", null: false
  t.bigint "byte_size", null: false
  t.string "checksum", null: false
  t.datetime "created_at", null: false
  t.index ["key"], name: "index_active_storage_blobs_on_key", unique: true
end

create_table "active_storage_variant_records", force: :cascade do |t|
  t.bigint "blob_id", null: false
  t.string "variation_digest", null: false
  t.index ["blob_id", "variation_digest"], name: "index_active_storage_variant_records_uniqueness", unique: true
end

create_table "appointments", force: :cascade do |t|
  t.bigint "user_id", null: false
  t.time "time"
  t.date "date"
  t.datetime "created_at", precision: 6, null: false
  t.datetime "updated_at", precision: 6, null: false
  t.index ["user_id"], name: "index_appointments_on_user_id"
end

create_table "reports", force: :cascade do |t|
  t.string "title"
  t.text "description"
  t.datetime "created_at", precision: 6, null: false
  t.datetime "updated_at", precision: 6, null: false
  t.bigint "user_id", null: false
  t.index ["user_id"], name: "index_reports_on_user_id"
end
```

2.1.3. User Requirements

Be able to access the website from any device with its full functionality.

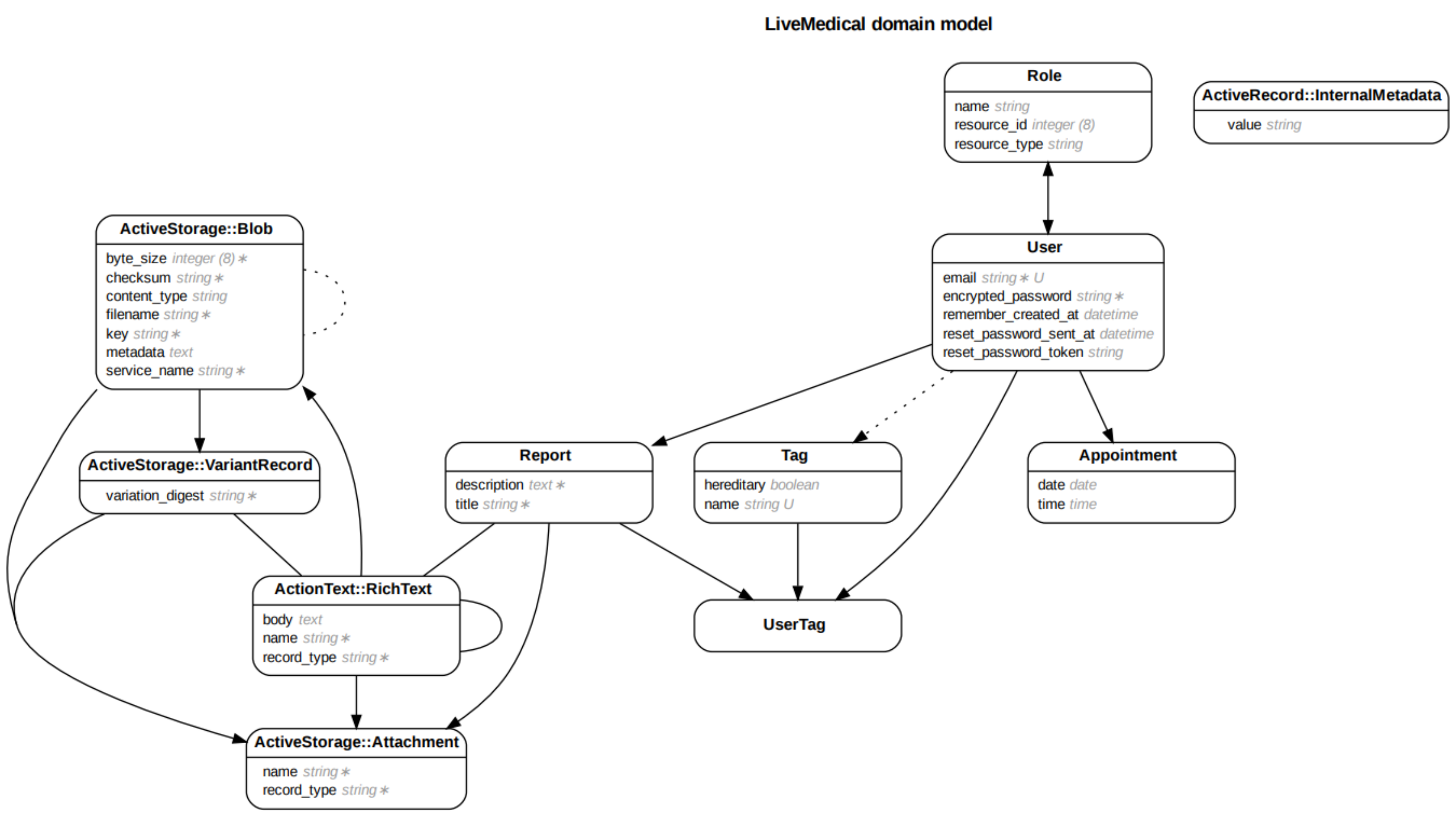
2.1.4. Environmental Requirements

Devices will need an internet connection and a device with a web browser to be able to view the website.

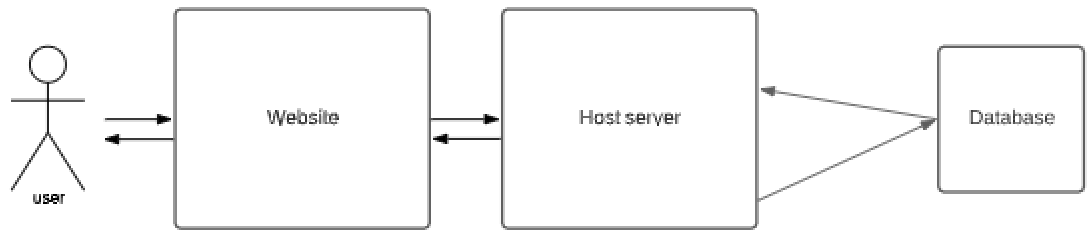
2.2. Design & Architecture

Domain model.

A full-size PDF will be included in my project's directory.

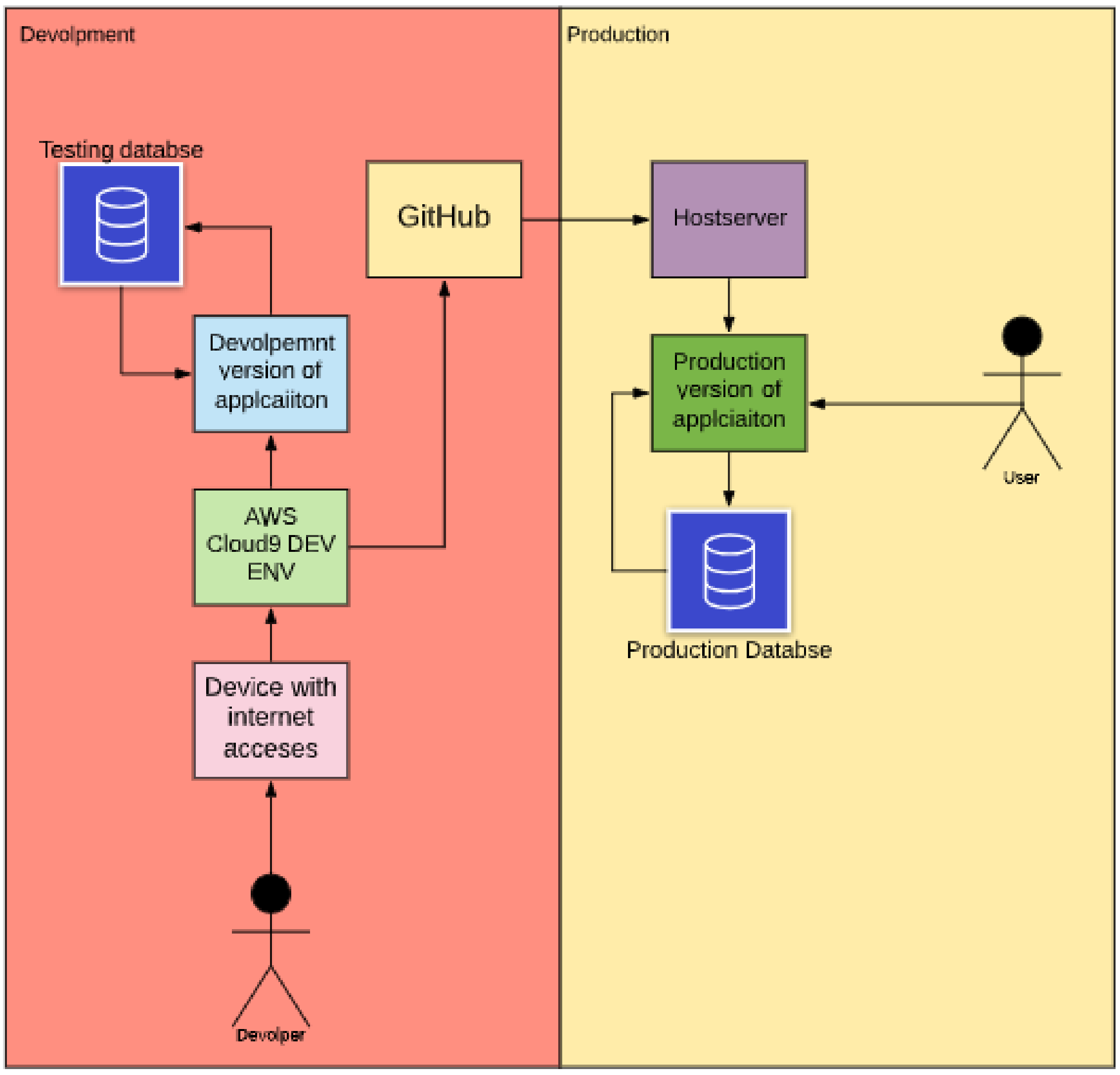


System architecture.



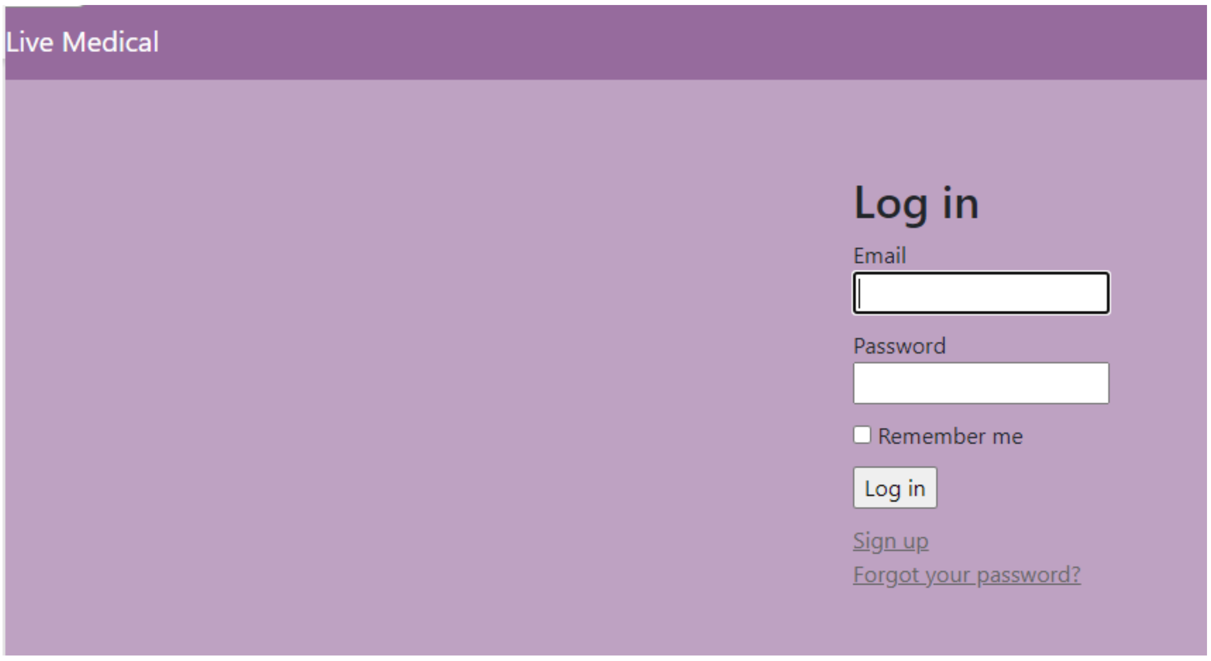
2.3. Implementation

The development of the project was done completely in the cloud with a separate development and production environment. Once I was happy in the development environment the changes were pushed to production though GitHub which automatically rebuilt the hosted application with all the changes I had made.   I chose this as it allowed me to develop my application from anywhere once I had an internet connection. While also simulation a SAAS applications development cycle.

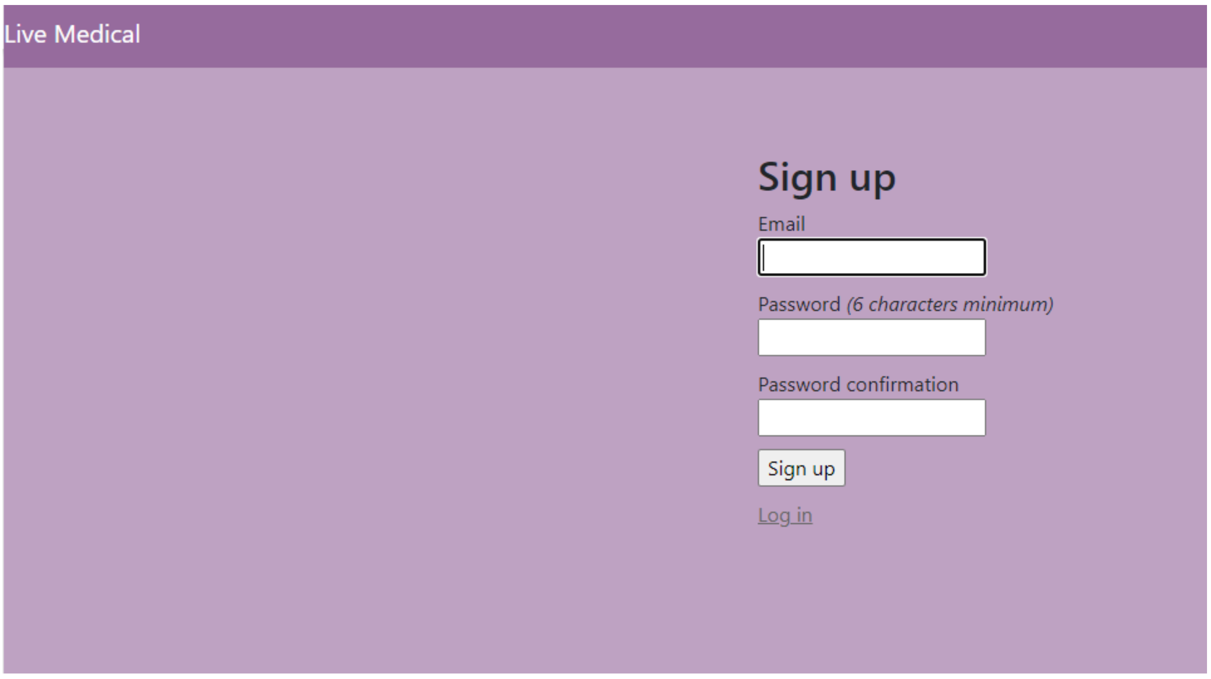


2.4. Graphical User Interface (GUI)

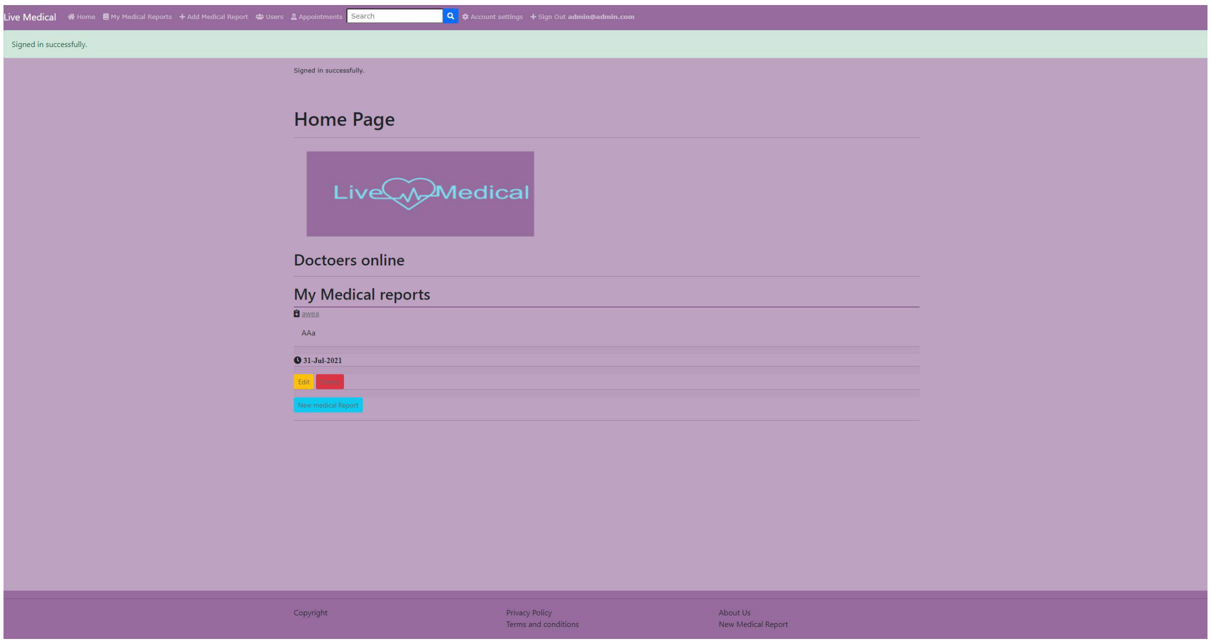
Login page



Sign up page



Home page with navbar, list of online doctors and list of user's medical reports


Live Medical · Home · My Medical Reports · Add Medical Report · Users · Appointments · Search · Account settings · Sign Out admin@admin.com

Signed in successfully.

Signed in successfully.

Home Page

Doctoers online

My Medical reports

awea

AAa

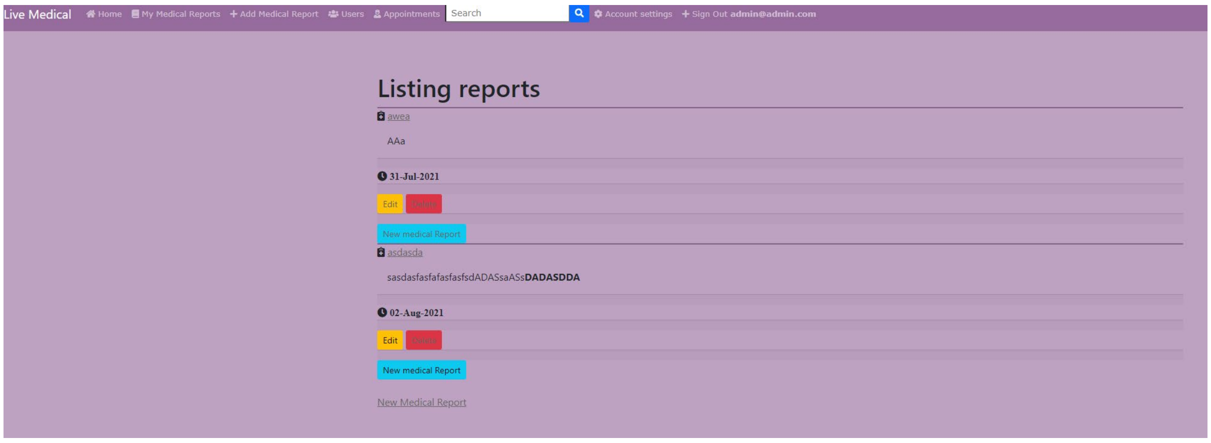
31-Jul-2021

Edit

New medical Report

Copyright · Privacy Policy · Terms and conditions · About Us · New Medical Report

My medical reports page shows all these users created reports


Live Medical · Home · My Medical Reports · Add Medical Report · Users · Appointments · Search · Account settings · Sign Out admin@admin.com

Listing reports

awea

AAa

31-Jul-2021

Edit

New medical Report

asdasda

sasdasfasfafasfasfsdADASsaASs**DADASDDA**

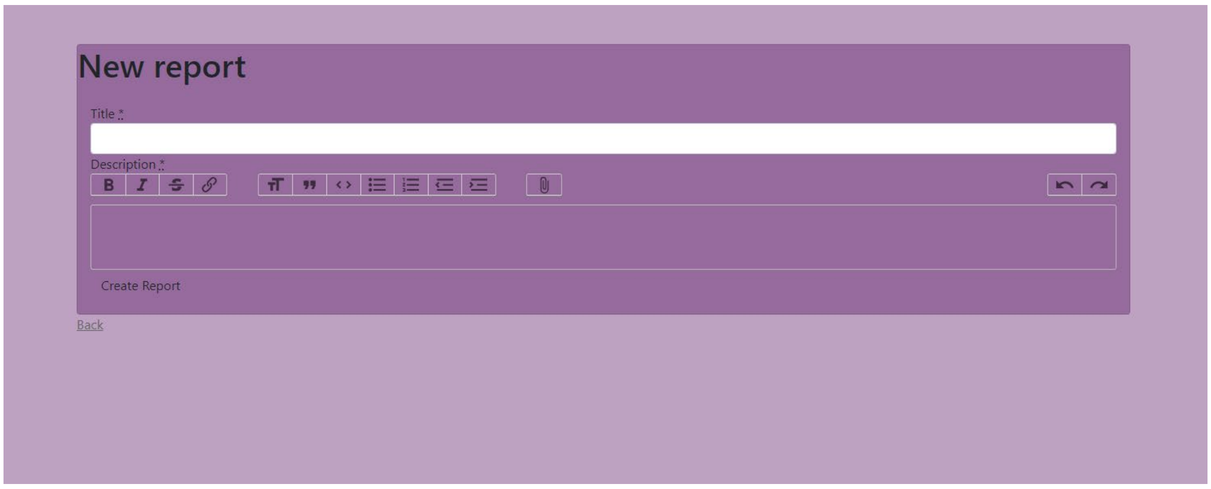
02-Aug-2021

Edit

New medical Report

New Medical Report

New medical report page for users to create their own reports with the title being the medical condition.


New report

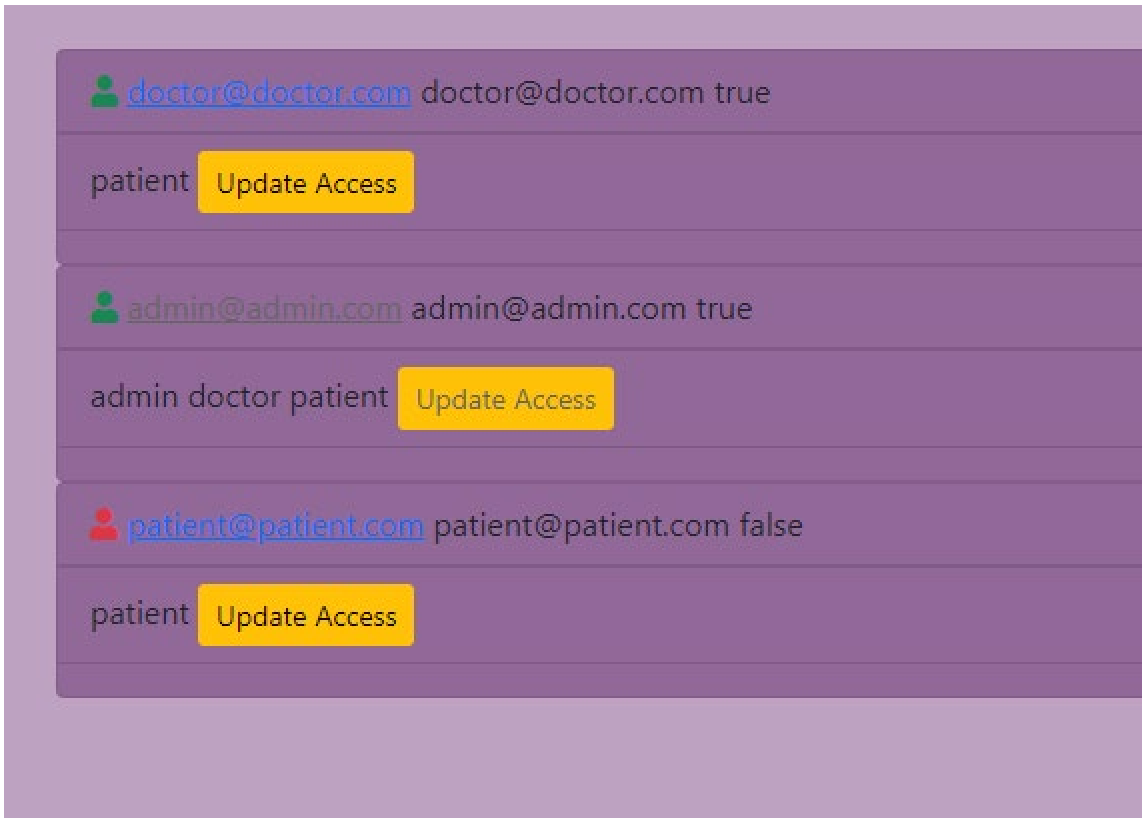
Title *

Description *

Create Report

Back

User section showing who is online. Green is online red is offline. Also shows the update access button if your admin.



doctor@doctor.com doctor@doctor.com true

patient Update Access

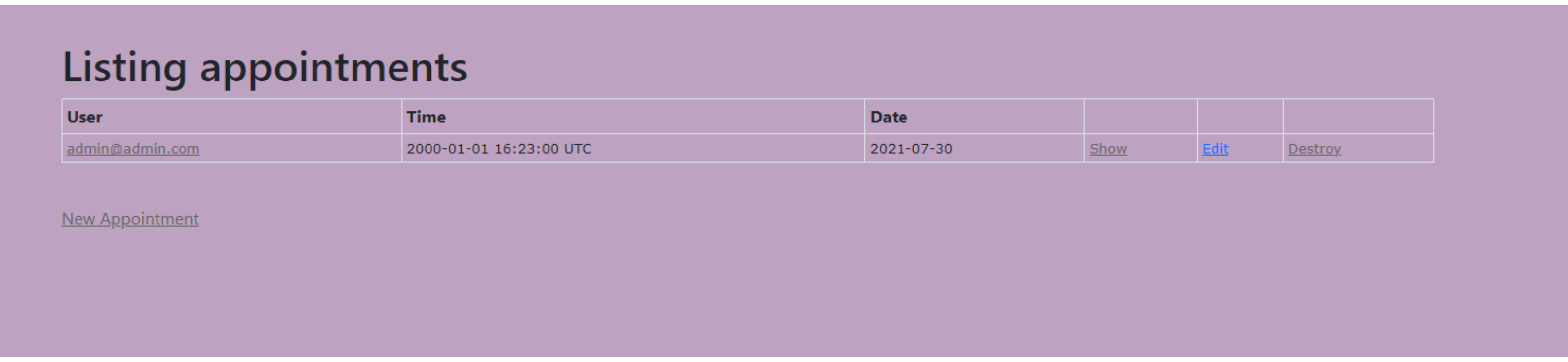
admin@admin.com admin@admin.com true

admin doctor patient Update Access

patient@patient.com patient@patient.com false

patient Update Access

Listing appointments section. Shows a list of appointments scheduled with that user

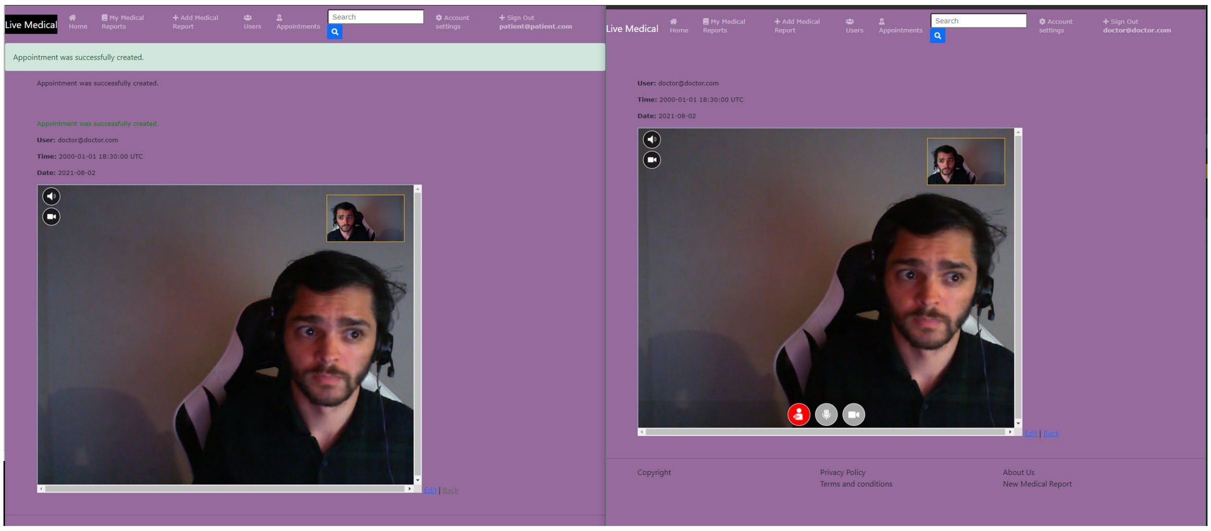


Listing appointments

User	Time	Date			
admin@admin.com	2000-01-01 16:23:00 UTC	2021-07-30	Show	Edit	Destroy

New Appointment

If you click show on the above screen it brings you to the video call screen. Below is a screenshot of it in action. Note: this can only be used in the production environment.



Appointment was successfully created.

Appointment was successfully created.

Appointment was successfully created.

User: doctor@doctor.com

Time: 2000-01-01 18:30:00 UTC

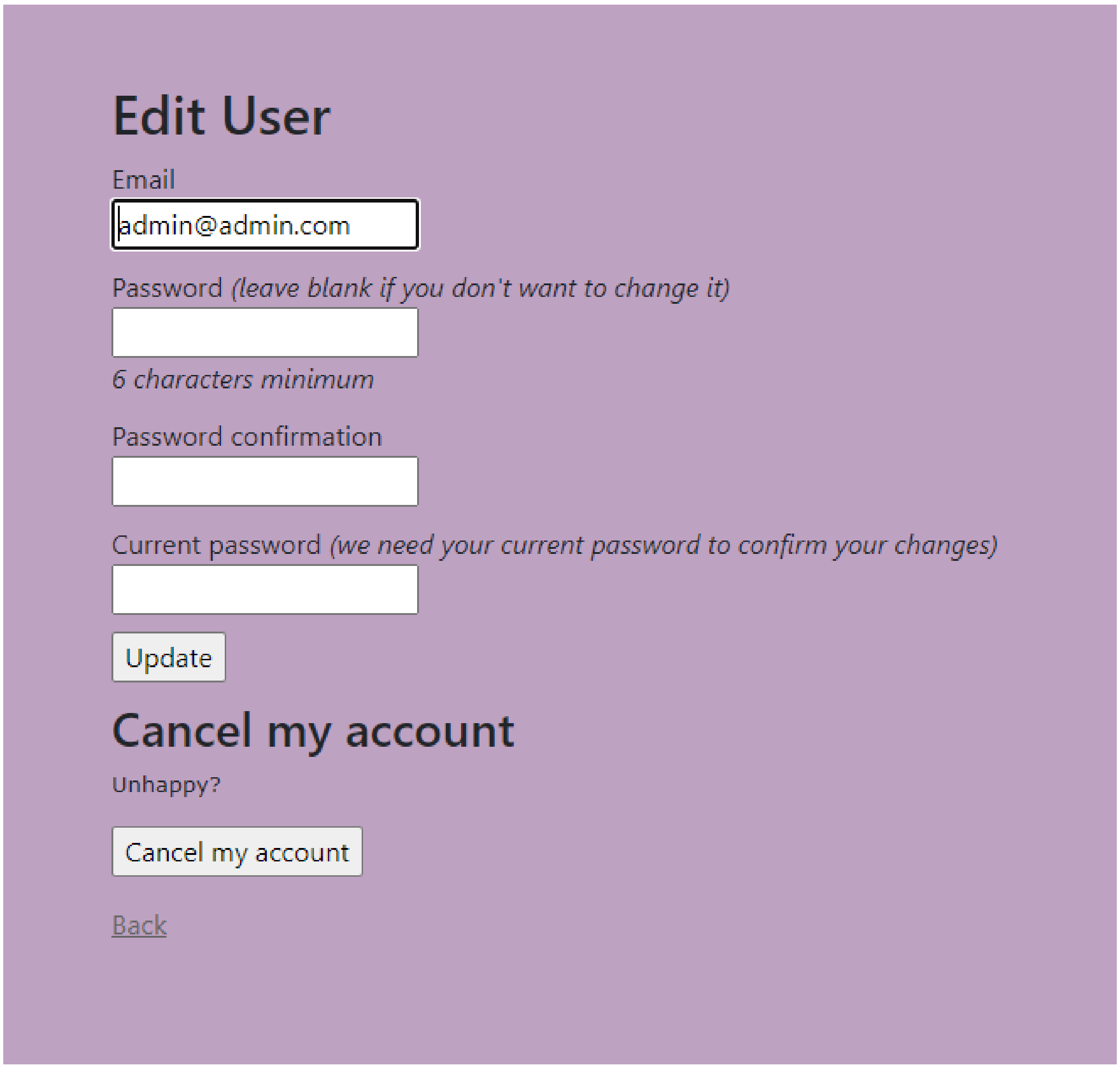
Date: 2021-08-02

User: doctor@doctor.com

Time: 2000-01-01 18:30:00 UTC

Date: 2021-08-02

Account settings section allowing user to update settings.



Edit User

Email

admin@admin.com

Password *(leave blank if you don't want to change it)*

6 characters minimum

Password confirmation

Current password *(we need your current password to confirm your changes)*

Update

Cancel my account

Unhappy?

Cancel my account

Back

3.0 Conclusions

I tried to make this application solve an issue that I had personally while I did complete this to some degree, I did notice some advantages and disadvantages.

Advantages

Dev and production environments

The separation of the development and production environments. This is a great advantage as it allows the webpage to still be accessible to users while I am adding new functions to the application. Once these new functions were completed, I was able to push the changes, and have it auto deploy knowing I would not have any issues.

Cloud

Fully developed and hosted in the cloud. I marked this as and advantage as it was a great learning opportunity for me while simultaneals allowing me to work on the project during times, I would have needed a local machine if I decided to install rails locally instead of in the cloud on an AWS EC2 instance.

Responsive web design

The whole website is completely responsive and will work on any device with internet access and a browser. This allowed the website to be as comfortable as possible no matter the device the end user chose to use.

Disadvantages

Website UI

While the website I feel is easy to navigate I feel I was not able to do the site justice. I originally wanted large pictures of doctors and users to have tags on the side to easily navigate but this proved to difficult and kept causing other display issues when implemented.

Tag system

In its current forms its way too difficult to use properly this issue occurred due to me linking the creation of reports with the creation of tags. So, you cannot create one without the other. While it does serve the purpose, I wanted its cumbersome and unsatisfactory.

Live Medical

4.0 Further Development or Research

While I am happy, I got to create this project once I wrapped up development, I noticed a few more things that I would have liked to have included or changed. Over the coming months in between work, I hope to complete this list further developments

Redesign UI

I am not happy with it overall and it did not come out as I wanted like my wireframes. So, I will be rebuilding the applications front end from scratch.

Rebuild Video Feature

I wanted to give the user way more control over this feature, but I was unable to implement it without it completely breaking the application in production and causing it to slow drastically. I will need to conduct more research on this so I can properly implement an API to allow for video calls.

AI chatbot

This application Would do great with a chatbot. The bot could ask basic questions and maybe point you to the Doctor best to suited for your needs or even prefill a medical report.

Notification system

The only way currently to check if you have a meeting coming up is to click appointments and manually check. This is not practical in the real world. I would like the application to notify you When a meeting is coming up and when a new meeting for you has been scheduled.

Tag system

The currently implementation is just bad. This needs to be redesigned from the ground up. It was supposed to be one of the main fucoses of my website and I failed to implement it properly.

Thew new system will allows users or doctors to quickly add tags to their profiles. These tags will be completed independent and allow you to share them with people you approve.

The above is what I wanted from the start, but I could not get it to work so I tied them to the medical reports which I thought would be satisfactory, but it was not.

5.0 Appendices

5.1. Project Proposal

National College of Ireland

Project Proposal

Live Medical

BSc (Honours) in Computing

Software Development

2020/2021

Anthony Cuddihy

X17103924@student.ncirl.ie

Contents

Live Medical

Objectives

The main object of this project is to build an application that will allow patients to book and attend video appointments with their doctors.

For patients it will

- Allow them to view Doctors schedules so they can schedule a time to be seen.
- Request a Video call with a scheduled doctor at that scheduled time.
- Upload medical history so it is in one convenient place.
- Be able to add blood relatives to their “family”. This will give the user visibility of medical complications that they may be susceptible too due to their family medical history.

For the Doctors it will

- Accept clients request for Scheduled time to be looked at.
- Be able to view all scheduled visits for themselves.
- Request video calls with clients.
- Be able to see client’s medical history (if approved).
- Be able to see client’s family medical history (no names just titles of hereditary conditions)

The application itself should be simple, intuitive & inviting to anyone.

Background

With COVID-19 affecting the world it has become even more apparent that we need a faster, easier and more covenant way to be seen by a doctor. Web videos would facilitate this requirement allowing people to have a video call with their doctor. This web app will also help people how are afraid to go to doctors due to the pandemic or people who may my physically unable to go. These Videos can then be re watched if required.

Being able to login to a website and quickly see if a doctor is free to see you or being able to see when you doctor is next available will be massive for people who would like to manage their time.

My website will also have the ability to view your past medical reports. Having all previous appoints information in one convenient place. Along with this functionality will be the ability to see your family’s hereditary diseases if the family members approve you viewing it.

I myself suffered with a hereditary disease that took almost 2 years to be diagnosed with. If my doctor were able to quickly see a list of health complications that I may be susceptible to I could have been diagnosed earlier.

Technical Approach

To start ill I will create a project plan and outline the project management methodologies I will be using. How the methodologies will be applied along with timelines for all major milestones.
Next ill being outline the requirements and functionality I would like the application to have and adding them to a UML diagram. This UML will then be used to develop the webpage wireframes and also when writing code.

Once the UML is completed ill move onto the logo design and colour pallet. For me designing the logo and settling on a colour palette is extremely important as it lets me get a feel for the product I am developing and helps me settle on how I want the application to look and feel for the end users.

When I am happy with how I want the applications colour palette and how I want the app to feel I can begin working in the mock-ups and wireframes. These wireframes will be created for each webpage I need to complete all requirements in my UML diagram.

Now I move onto the development. For the development of this project, I am going to be using a development and a separate production environment. I chose this to simulate a software a service application. This means once I develop part of the functionality, I will test it in my development environment and once the code is tested and working ill push it to GitHub which will automatically take my website offline and rebuild it with the code I added.

Special Resources Required

The application will be built using ruby on rails and I will be using the AWS Cloud9 development environment for writing the code. I chose AWS Cloud9 as its advantage of being independent from my local machine means no matter what device I am on as long as it has an internet connection and a browser, I will be able to write code. This suits me as I do not have a guaranteed, I will be able to work on my device that has rails installed.

Heroku is where ill publish my application to during the test-driven development and also where my completed application will be hosted.

Since the application will be developed in rails, I will be using several public gems available on rubygems.org. So far, I have identified 14 gems that will be using in my application. During the development this may increase or decrease if I find more gems that I think will work well or if I create my own gems to replace the ones I identified.

Project Plan

Name	Start	Finish	Length (days)
Project Pitch Video	10/10/2020	17/10/2020	7
Proposal	17/10/2020	26/10/2020	9
Requirements Spec	24/10/2020	07/11/2020	14
Mid point	07/11/2020	04/12/2020	26
Application Devolpment	01/02/2021	10/04/2021	71
project Presentation	02/05/2021	17/05/2021	14

Technical Details

I will be using Ruby on rails for this application. So, I will be using HTML/HAML. JavaScript and CSS for the front end. I think I will be settling on HAML as it is a more lightweight than HTML and may help with load times on the website. The rest of the application will be built In C as that is what ruby uses.

As stated previously the application will be built and tested in development environment once I am happy, I can push the changes to GitHub which will auto deploy the changes to my production environment through a pipeline I will setup.

Evaluation

For this application I will be performing specific Unit testing for certain parts of the application. I will be breaking it down it each functionality and will be testing each till it gets a pass.   A pass would be the outcome I expect.

For the Patients

- Can the view and edit their own medical reports and no one else's?
- Can they change or recover login info?
- Can they see doctors who are online?
- Can they book an appointment?
- Can they attend a video call with a doctor?
- Can they update medical tags?

For the Doctors

- Can the view and edit their own medical reports and no one else's?
- Can they change or recover login info?
- Can they see online users?
- Can they book appointments?
- Can they attend a video call with a patient?

For the admin

- Can they update users' access to give or take away permissions?
- Can they change or recover login info?
- Can they view online users?

As the above requires a lot of human intervention I will try to automate some of the testing steps.

5.2. Reflective Journals

Week beginning Monday 5th of October

Started my research into what type of project I would like to do. Settled on a mobile game as I have never done that before also began looking into software, I will be using to build the application.

For AI I decided to learn the more complex moves of the game, so I will be familiar with it for the project.

Data application development required that we have 2 data sources for out CA and advised us to locate them as soon as possible. Found the base data I wanted to build the CA around "Ireland housing prices "but I am having trouble finding a complementary one.

Web services and API, I just completed the lab again to try brush up on it.

Week beginning Monday 12th of October

Finally decided what I wanted my final year project to be which is an Augmented reality multiplayer mobile game once I decided that I decided to research what core software I will need.

Settled on android studio and AR kit for the mobile / Augmented Reality development.
Unity for the game engine / development.
Blender for the character modelling /rigging & animations,
photon (PUN) for multiplayer networking.
& NoSQL firebase database for backend.
Once I settled on the core software needed and the software, I wanted to build for my final year project. I begin creating my script for my project pitch video.

I was called to work in the office for this week so was not able to be as productive as I am when I am working from home.

Week beginning Monday 19th of October

As I did not know if I will be doing the project, I pitched I decided to spend most of this week doing my CA1 for mobile application, started CA1 the chess game for AI and started researching "PESTEL" for Strategic Management.

The chess game is a mess if I am being honest, it is just wall of code. So, I spent my time going over to make sure I understood what the code was doing.

Mobile app came along nicely finished login and main activity.

Strategic Management CA needed peer reviewed papers so was on Google Scholar searching for PEST and skimming them to see if I wanted to read them further. Copied links for ones I liked into a word doc.

Week beginning Monday 26th of October.

Mobile App CA1 was due this week, so I Just focused mostly on that while waiting on an update for my pitch.
Finished the remaining three activities for my application and started working on the documentation that complements the CA

Documentation finished so I combined the docs and got them ready to submit.

Started writing my PESTEL analyse for Strategic management.

I was called to work in the office for this week so was not able to be as productive as I am when I am working from home

As I still had not got a response about my pitch, I contacted my supervisor to see what times they are usually free to meet – we settled on Wednesday evenings for the weeks we are going to meet on Microsoft teams.

November

Covid, death … not a good time.

30th of November to 31st of December

Spent most of this month working on all the CA's due for every other module. AI, Data application, Strategic management, Mobile App and web services and API 16th of December I finished up my prototype and started working on my video.

June2021

Started to rethink my original idea as it was not what I wanted to do anymore I was unmotivated and brought back unwanted memory's therapist recommended a change, so I did.
Beginning thinking of new ideas to pitch and work on my midpoint documentation. Looked through the facility projects and found one by Mohammed that was close in concept to an idea I was already thinking. I combined his Video app with my medical history idea and started to get to work.

in the second last week of June, I was delt a heavy blow as I had just found out that I was not reporting the midpoint submission as I requested but was due to be submitting final project submission.

I started scrambling to find out ways to help me quickly develop a project. I stumbled on Ruby and Cloud9. This was great as it would allow me to be able to code in between shifts at work or on a commute etc and have my data auto save. IT being all in the cloud also reduced the risk of something happening and me losing all my code.

July 2021

First week I spent researching gems that would be compatible for my project. I found 14 that I thought would be a good fit and fingers crossed help me speed up development. Cloud9 is great and if anyone reading this is teaching cloud app, I would suggest looking into using it during classes. It is a phenomenal tool, and I was able to use it to develop much faster than using the native install.

Application is now being developed and, on the side, I am trying to teach myself more about bootstrap and JavaScript.

Also working on my technical documentation which I found is a lot easier if you do it while developing the application
Application is nearly completed its not great looking but it that I could do in this time.

August 2021

Started working on My presentation.

5.3. Application Logo and Thumbnail