1. **Introduction**

The main aim for this document is to illustrate the requirements and specs of the web application. I will give a brief description of functionality of the project and some of the non-functionality of the project. The web application that is being created will have a friendly user experience on the client side who will be able to have control of the material on that web page. I am creating this web application in the Ruby programming language as it is a flexible and very good language to learn.

1. **Scope of project**

The web app I am creating will have multiple users where they sign up and post up music album they like. I will have different Genres and have a general rating system, a review page also you can go on different user’s pages so be more like a social website for music only. It will have easy to navigate with the use of tabs.

The project will be easily implemented under various situation’s; we can add new features as I require also trying to make it a reusable as possible.

1. **System Requirements**

This web application will run on all browsers and to run it locally on your pc you might need to configure the settings depending on the Operating System.

* You will need Ruby on rails CMD prompt or Git Bash but make sure you download the correct version for the OS your using.
* Download Rails installer from Railsinstaller.org
* Install Ruby gems from the Bash Prompt with the code below
* curl http://installrails.com/update\_rubygems.rb | ruby
* This is how to download rails
* $ gem install rails --no-ri --no-rdoc
* Download a text Editor I used Brackets as it has a gitbash plugin
* Go into your folder with GitBash your app folder and run it by typing ……….
* $ rails server
* Than Use the browser and visit <http://localhost:3000>

With all the packages and gems it can difficult to run the app from github on windows. Especially when using Imagmagik I needed to use a Linux based platform, I didn’t have any laptop with Linux so I created it on cloud9 which is an online idk that runs on Linux and has rails environment, to know more about cloud9 go to the details below.

**3.1 Using cloud9.io**

To get started, just do the following:

1. Run the project with the "Run Project" button in the menu bar on top of the IDE.
2. Preview your new app by clicking on the URL that appears in the Run panel below (<https://albumratingapp-heanuea.c9users.io/>).

1. **Tools Used**

**Ruby** is an Object-oriented language which it very like Python language. It has a great number of libraries where you can easily extend. It is also open source so there is a lot of resources out there for it. 

**Rails** is a Software library that extends the ruby language, Ruby is a framework for building web applications/websites it combines ruby and html, css and javascript and runs on a browser. Rails is considered Backend. It is great for for smaller projects but can also do larger projects. It uses gems for more complex libraries using Ruby Gems. (see below)

**Ruby Gems** a package manager for the Ruby programming language that provides a standard format for distributing Ruby programs and libraries (in a self-contained format called a "gem"), a tool designed to easily manage the installation of gems, and a server for distributing them. (see all gems below)

* + **Sass/scss** Sass known as the indented syntax as scss is more a CSS like syntax The SASS gem is an official integration with Sass stylesheet language.
  + **Devise** <https://github.com/plataformatec/devise>

This gem is a flexible authentication solution for rails based applications

It does all the storing passwords by using Http and POSTS requests.

* + **PaperClip**

Paperclip is intended as an easy file attachment library for ActiveRecord. The intent behind it was to keep setup as easy as possible and to treat files as much like other attributes as possible.

* + **Simple\_form** <https://github.com/plataformatec/Simple_form>

Make forms Easy……. It gives all the flexibility of helping you create a simple form and make it easy for the eye.

* + **Sqlite3** <https://rubygems.org/gems/sqlite3>

This module allows Ruby programs to interface with the SQLite3 database engine (http://www.sqlite.org). You must have the SQLite engine installed to build this module. Note that this module is only compatible with SQLite 3.6.16 or newer.

* + **Puma** <http://puma.io/>

(this GEM was in the application when I built it comes with rails 5)

Puma is a simple, fast, threaded, and highly concurrent HTTP 1.1 server for Ruby/Rack applications. Puma is intended for use in both development and production environments. To get the best throughput, it is highly recommended that you use a Ruby implementation with real threads like Rubinius or JRuby.

* + **JQuery-rails** <https://rubygems.org/gems/jquery-rails>

I used this gem as I needed for The rating and I needed to use Jquery for that to get it to work and this gem allows me to implement JQuery into the application.

1. **Architecture of the Solution**

Ruby on Rails uses the Model-View-Controller (MVC) architectural pattern to improve the maintainability of the application. The Model centralizes the business logic, the View manages the display logic, while the Controller deals with the application flow. The MVC allows a clean separation of concerns, in the way that it keeps the business logic separated from HTML views. Additionally, it improves decoupling and testing.

**Model**

The Model layer carries the business logic of the application and the rules to manipulate the data. In Ruby on Rails, the models are used to manage the interaction with their corresponding elements in the database. The Models represent the information in the database and do the appropriate validations.

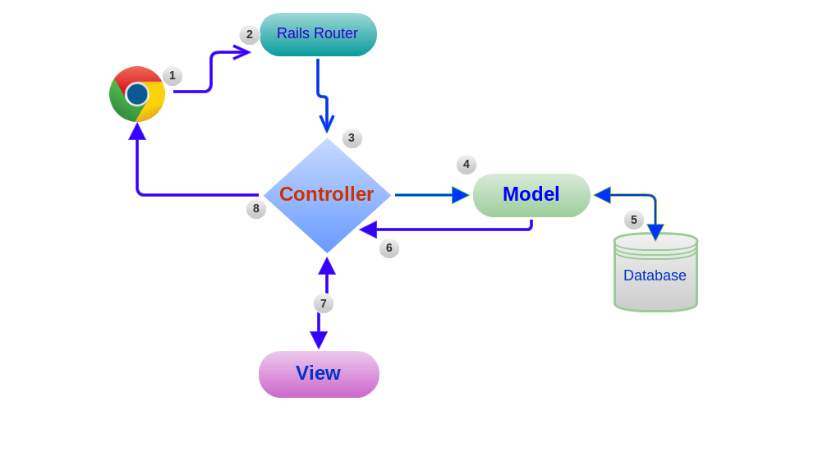
**View**

The view is the front-end of the application, representing the user interface. In Ruby on Rails, views are HTML files with embedded Ruby code. The embedded Ruby code in the HTMLs is simple (loops and conditionals). It is only used to display data to the user in the form of views. Views are used to provide the data to the browsers that requested the web pages. Views can server content in several formats, such as HTML, PDF, XML, RSS and more.

**Controller**

Controllers interact with models and views. The incoming requests from the browsers are processed by the controllers, which process the data from the models and pass it to the views for presentation.

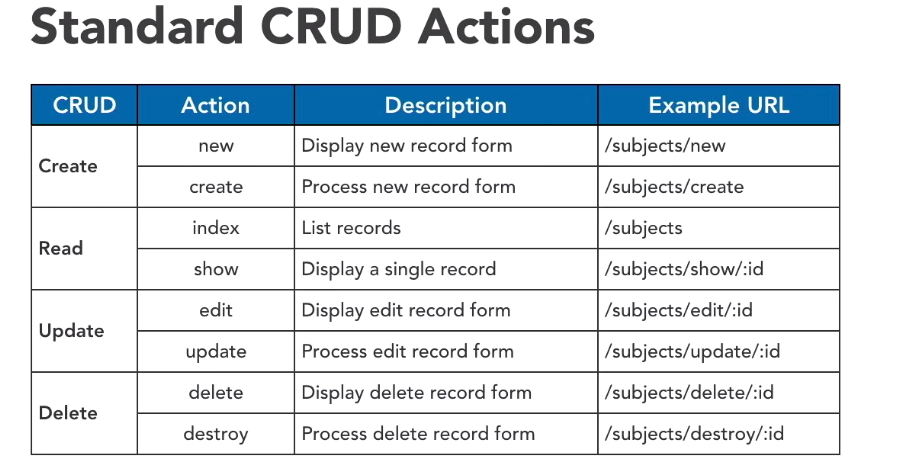
Here is a model.



**CRUD**

First we used controllers and views to create dynamic pages and return them back onto browser. Than we used the Models to interact with our database this is not threw our browser but threw the rails console. We put the two together to make the models work inside our controllers to manipulate the data from the browser. Here I going to decide to use CRUD (create, read, update and delete)

Here are the standard actions for Rail now you don’t have to use the correct names but it makes life easier if you do.

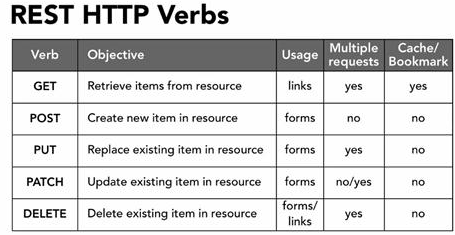
****

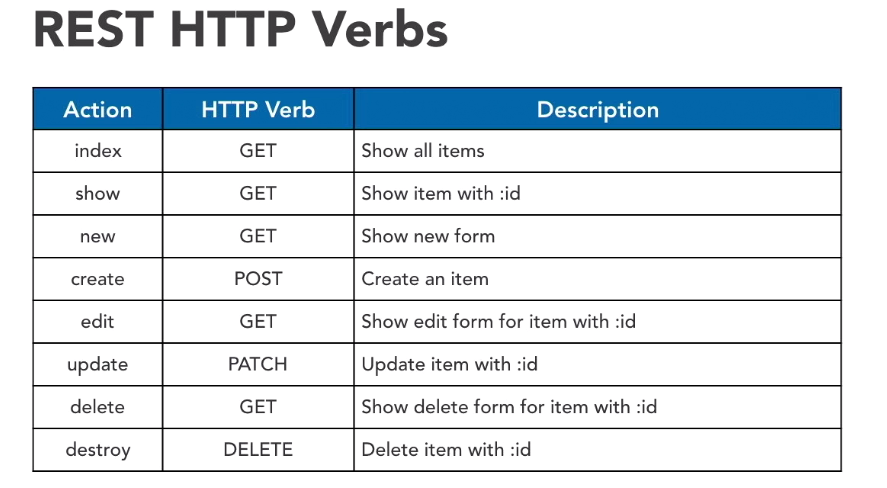
You can see there is two actions to each CRUD element. These are core actions for working with rails they are used all the time to navigate through pages or sections of the application.

I have multiple controllers in my application most of the time you would have one controller per model.

The URLS are used to define the model you are using and the actions are being performed.

**REST**

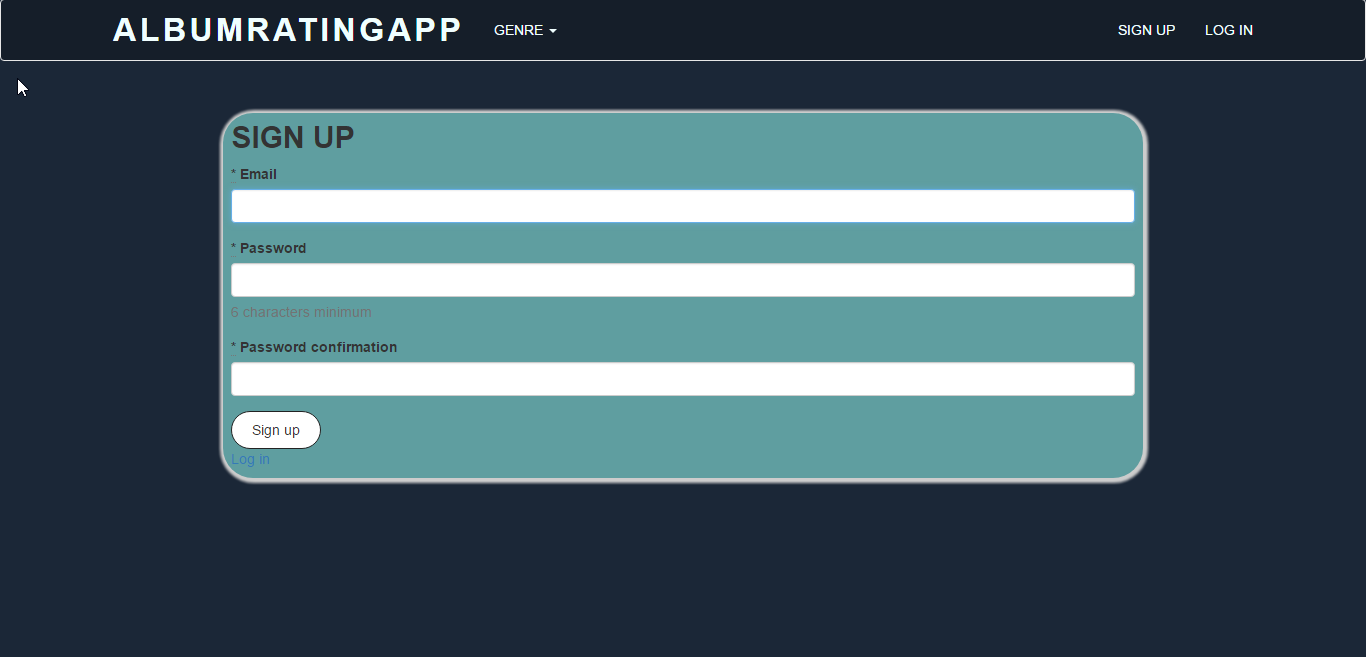
* Representational state Transfer
* Do not perform procedures
* Perform state transformations upon resources e
* The difference with post in ruby is we only used to create a new item as most other web applications they are used for crud. If you want to update an existing item, we use patch and delete is a delete an existing item.
* The verbs above correspond with the crud verbs that we used earlier here is a table to represent how these maps with REST.



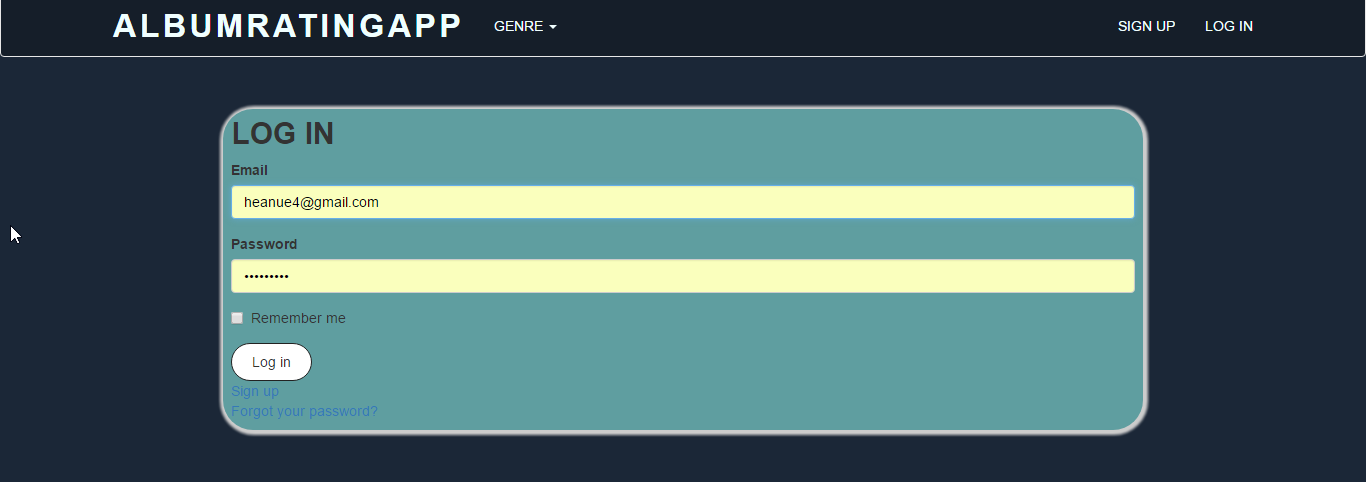
1. **Design Methodology**

* The way I went around this project is more the Agile approach.
* **Brain Storming,** I had very little to pick from I had to learn the basics of ruby on rails and understand the framework but I knew I was going to do some type of web application, first I was going to do a typical blog like app but said I wouldn’t mind trying to upload images or some type of media. I have been looking at some examples online and seen a few rating apps so I decided that a mad music fan and decided to go with an Album rating application with s spotify add on.
* **Design** I wanted to create a nice-looking form where you enter details and album art with some comment box. A header with user sign in and menu with the body of the app filled with the content now with ruby I was only able to use bootstrap, for the first half of the app mostly doing the workings of the app before finishing of the styling. *(I wanted to get a working app before style).*
* **Build** Creating a user friendly with the idea that each user has a log in and a that comments are locked when signed out. But to successfully build the project I had to build it on cloud9 as I was running it on windows10 and it was giving issues.
* **Test/Love it,** running successfully on cloud 9 and having minor issues I have tested the application and it is up to standard for now

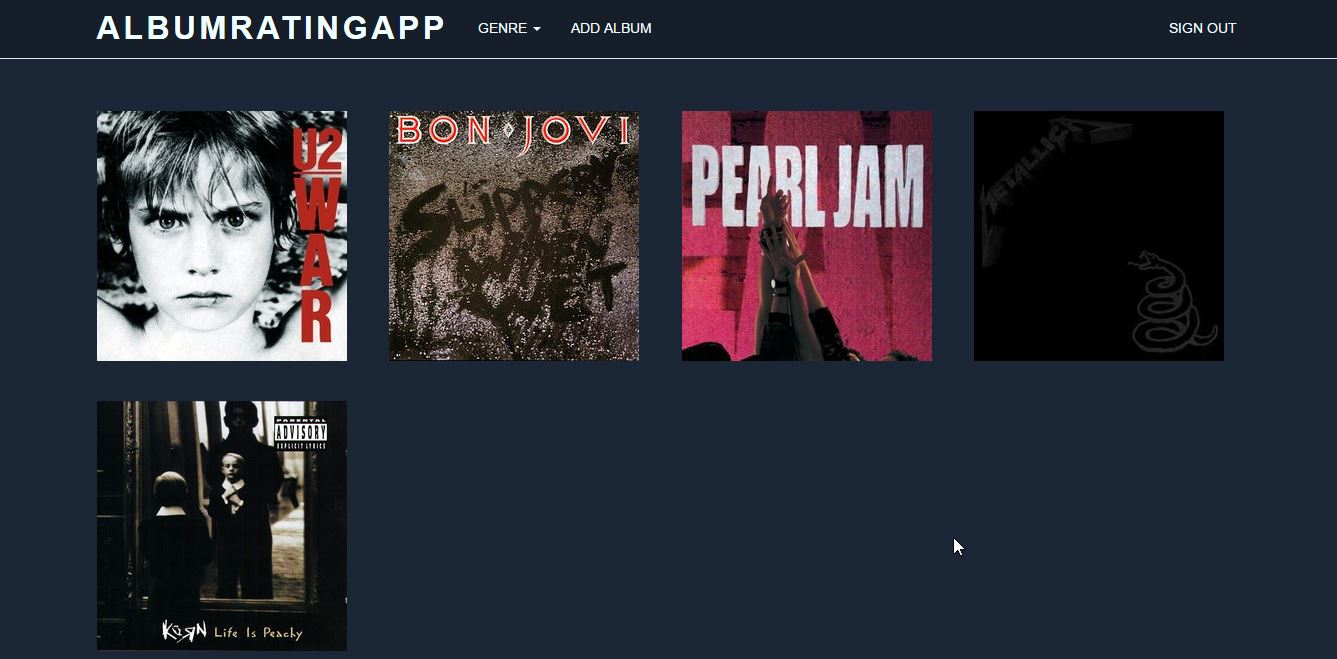
1. **Features** Here is the sign-up page where I can create a user and generate a password with devise this gives great authentication.



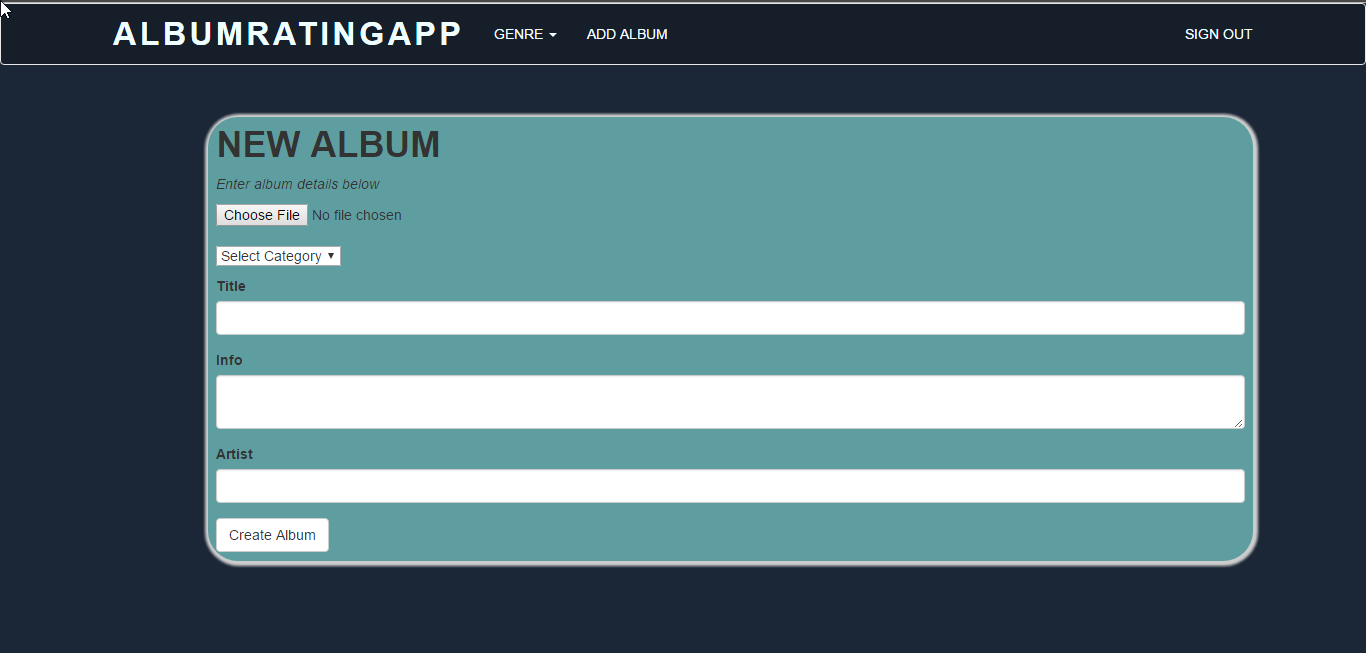
Next we have the Log in page this is after we have signed in you can see that devise has the forget your password and has a cache set up nice little gem very handy for log in forms. With this gem, I had to create a User schema so the application could have multiple users.



Next we have the main page where you have your albums on display, here you can select on the add album tab. This is link that is created in the views/layout/application.html.erb file.



After that you get this screen (below) where we created a form with simple form gem



Here is the paperclip gem where we could upload a file a tricky to get right as paperclip needed another software called ImageMagik an image processing software.

This next page is the Album Info page where you can delete and edit even add a comment and rate it.



As you can see I have an average rating set up but unfortunately is not working now.

Another schema I had to set up which was a mean feature on this app was the genres to put each album on. Than I had to get a dropdown menu from bootstrap which easy enough to put together.



1. **Limitations**

Here are some of the limitations that I had during my time developing this project, I had issues with trying to learn the basic syntax of ruby as well as create a web application. Here are some of the limitations.

* ***Time,*** *I found that time was against me trying to do some research and looking for answers I probably ran out of time to finish my app to where I needed*
* ***Version support*** *I was a long time researching the what ruby and rails versions to download I found a lot of older packages and gems were not compatible with the newer versions. Also, some gems not working at all with ruby 5.*
* ***Rails Support*** *I was writing with the latest version of ruby on rails and found very little help on the newer version even its own documentation was not very helpful. A lot of gems were older so I found that rails 4 was still the main version being used online.*
* ***OS support*** *Rails does not work well with windows as I found most of the time trying to configure a server or platform for rails. Firstly, Git bash was great but didn’t run most of my project so I downloaded a rails console environment which was great, but soon discovered that some gems needed other software to run properly. This was difficult to configure paths and get them to read each other. I started reading that Linux was the Os’ to write rails apps with so I decided to finish my project in a Linux Environment*

1. **Known Bugs** 
   1. The only bugs I found was in the review part of my project that I was running an error when I added an average rating script in my code



This is in the show action in the albums/controller folder when I add a review or 2 I would get an error.

* 1. The JQuery for my rating does not seem to work luckily not getting an error I think it could be cloud9. *(Still working on that).*

1. **Conclusion**

The project was about the overall working of a database and its architecture, a web application created in ruby on rails that pulled data to and from a database. How the whole project should be documented and the delivery of it with a presentation.

Will this be a multi-user system of a single-user system (for one user at a time)?

Are there special business rules and/or calculations that pertain to this industry?

How many data entry screens are required?

What kind of output do the users require?

Will the system produce web pages, printed reports, mail merge files or other types of output?

What is the subject of the database and what purpose does the data stored serve?

Well this project I learned how to better manage my time and research skills.

I also found I have a better understanding in how a database works with a web application especially how CRUD and restful apps work. But as the project had no specific language that we had to cover I’m delighted I picked Ruby on rails, I found it a great little language for this type of project especially on the smaller scale. I’m not the best for picking up on syntax to easily but found it easy to understand there is very little in terms of brackets and is very English based. Routing to in a web app I picked up easily enough. Minor issues like my environment slowed me down as I had awful issues with windows but found this module or project one of the most interesting as I wasn’t restricted to one subject.

One last thing I will keep ruby on my list of languages as a hobby or someday work related.