

**National College of Ireland**

**Project Submission Sheet – 2020/2021**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name:** | Ronan Behan | | |
| **Student ID:** | X19141815 | | |
| **Programme:** | Higher Diploma in Science in Computing Information, Software Development | **Year:** | 2019/2021 |
| **Module:** | Project | | |
| **Lecturer:** | Hamilton Niculescu | | |
| **Submission Due Date:** | 25/4/21 | | |
| **Project Title:** | Contractor Contactor | | |
| **Word Count:** | 11131 | | |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

|  |  |
| --- | --- |
| **Signature:** | Ronan Behan |
| **Date:** | 25/4/21 |

**PLEASE READ THE FOLLOWING INSTRUCTIONS:**

1. Project deliverables should be submitted using the Upload Point from Moodle.

3. **You must ensure that you retain a COPY of ALL projects' assets**, both for your own reference and in case a project is lost or mislaid.

4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**

5. All deliverables must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

National College of Ireland

Higher Diploma in Science in Computing Information, Software Development 2019/2021

Name: Ronan Behan

Student ID: x19141815

Email: x19141815@student.ncirl.ie

Title

**Contractor Contactor Technical Report**





**Table of Contents**

[Executive Summary 7](#_Toc351559315)

[1 Introduction 8](#_Toc351559316)

[1.1 Background](#_Toc351559317)

[1.2 Aims](#_Toc351559318)

[1.3 Technologies](#_Toc351559319)

[1.4 Structure](#_Toc351559320)

[2 System 10](#_Toc351559321)

[2.1 Requirements](#_Toc351559322)

[2.1.1 Functional requirements](#_Toc351559323)

[2.1.2 Data requirements](#_Toc351559324)

[2.1.3 User requirements](#_Toc351559325)

[2.2 Design and Architecture 19](#_Toc351559328)

[2.3 Implementation 22](#_Toc351559329)

[2.4 Testing 3](#_Toc351559330)9

[2.5 Graphical User Interface (GUI) Layout 44](#_Toc351559331)

[2.6 Customer testing 49](#_Toc351559332)

[3 Conclusions 50](#_Toc351559334)

[4 Further development or research 51](#_Toc351559335)

[5 References 52](#_Toc351559336)

[6 Appendix 53](#_Toc351559337)

[6.1 Project Proposal 53](#_Toc351559338)

[6.2 Project Plan 59](#_Toc351559339)

[6.3 Requirement Specification 76](#_Toc351559340)

# Executive Summary

There is a gap in the market. Job boards exist, and target different type’s users for different types of usage. For example Irish Jobs allows users to have a profile and put up jobs. There are usually separated by type of user and subscription. LinkedIn comes from a more social media point of view and Fiverr offers a contractor directory, which you must pay to use. Fiverr is also for the international. Most sites are international and this works for online contractors such as developers. Other sites have notice boards.

There is a gap here. Ireland is not represented well on the contractor directory side in a one stop, open, easy to use Website for free for all trades. The other sites are very international. There is room for a good database of people here for everyone to contact. It also helps people living in the rural areas as they can see who is near and compare rates.

This project seeks to fill this gap. To have a contractor directory for Ireland, free to use for all. It will be heavily focused on the Irish market to make it easy for people in their localities around the island. Customers will also be able to find any type of contractor on this from PTs to Tilers to UX Designers. There will also be a Notice Board for customers to put up jobs for contractors to respond to.

This Website will be free to use, user friendly, open to all types of services, Irish focused, allows people to set up profiles for the contractor directory and there is a notice board. It will be made on the AWS platform in the languages of Html, CSS and Ruby on Rails.

# Introduction

This is a Web Application to connect Contractors and the public. It is a user-friendly interface that allows Contractors to put their profile up along with their portfolio and any other information they deem sufficient. The contractors can then be contacted by the public. This will be for any contractor no matter what domain. It will focus on the Irish market exclusively. A one-stop shop for all services.

The other side of the Web Application is for the public. They will be allowed to put up work on a notice board which only the contractors will have access to. Here they can message each other.

## Background

There are many Website which offer different functionalities to users on the jobs market. Some offer social media sides, more offer major services to the recruitment industry. Many sites are focused on certain professions, for example STEMs. There are sporadic unfriendly Notice Boards with a haphazard category system. A lot of job focused sites are internationally focused.

Taking all these positives and negatives into account it’s clear that there is room for another Irish oriented Job site but one which would give more power to the profile of the contractor, target all trades, is free, easy to use, easily accessible and allows the customer and the contractor easy communication.

## Aims

* Create a Website with HTML, CSS and Ruby on Rails via AWS.
* Allow certain functionalities, with most in the Navegation Bar: Sign in, Sign out, Register and create a profile, edit the profile. Have the profiles up in a Contractor Directory for perusal by the customer. Have a Notice Board Page where you can put up a job notice for public view and allow comments. There will also be a help page and a search bar which allows you to search for the contractor in the directory.
* The Website should be very straight forward and easy to use.

## Technologies

**HTML**

HTML was the main front end language used in the making of this project. It was used for all the pages, helping them form a structure using forms, divs, paragraphs, footers and applying the CSS to this after. Most of the coding is in the HTML.

**CSS**

Css was applied to give the pages a more beautiful interactive, aesthetically appealing and to improve the user experience. It did this through specific calls to the HTML code. The main class for this is the application.css folder

**Ruby on Rails**

Ruby on Rails was used in this project for the backend language. It was used it to create a rails application on AWS. It was adopted to do the scaffolding, create the database, create functionalities, insert gems, implements security and privacy code and allow greater versatility to the front end.

**AWS**

AWS was the IDE upon which this project was built. 3 different languages were implemented in the course of the project. Html, CSS and Ruby on Rails. The code for the project is stored in the Contractor\_Contactor folder within the system. Inside this folder there is a further breakdown of the project. This breakdown went into specific folders assigned by each language. The IDE has great UX and it is easy to shift around the platform from language to language performing MVC functions.

**Canva**

Canva was used for the creation of the logo.

# System

## Requirements

These are based on the original requirements set. This dictates how the Website works. There have been a few changes since the previous Requirements Specification. These are outlined.

### Functional requirements

The main requirements are:

* A User can register.
* A User can log in/out,
* A search bar that helps locate contractors in the Contractor Directory.
* Set up a Profile and have access to a Profile page.
* The Profile will be available on the Contractor Directory.
* The Profile can be edited.
* The User should be able to post on the Notice Board.
* The User can add comments to a post on the Notice Board.
* The User has access to a help page with a Contact Form and FAQs.

There are a few changes here since the previous project requirements:

* The Profile has been have rolled down in how extensive it is. For the purpose of this project the information required is down to what is necessary, minus the profile picture. This is due to time constraints in the project timeline. The ability of the Contractor to message people via the Contractor Directory has been removed again due to time constraints.
* Another change is that the Website asks if the user is a Contractor or a Customer upon entry but all are available to look at the Company Directory and the Notice Board for free. This is to stimulate user growth.
* The final main change is the search bar. It should be more specific and by able to pick out a contractor from the directory but this was left out due to time constraints.

### Requirement 1 <Register>

#### Description & Priority

The user will have the option to register by clicking on an option in the Nav Bar.

Here they will give their information to create a profile

Users do not need to sign in to use the site but will have the option to do so.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the register functionality works.

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when a user goes in to the Web Application and if they have no account and then registers.

**Main flow**

1. The user enters the URL into their search engine.
2. Either user enters into the Web Application.
3. The user will enter the user details to register.
4. The user enters the Website signed in and registered.

**Exceptional flow**

E1: Incorrect User Register Details

1. The user enters the required user details
2. The system tells the user that the details are already in use.
3. The system asks the user to re-enter the details.

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition.

### Requirement 2 <Log In>

#### Description & Priority

User will need a user name and a password to log in.

Users do not need to sign in to use the site but will have the option to do so.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the log in functionality works.

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when a user logs in to the Web Application.

**Main flow**

The user enters the URL into their search engine.

Either user enters into the Web Application.

The customer or the contractor will enter a username and a password to log in.

**Exceptional flow**

E1: Incorrect User Sign In Details

1. The user enters the email and password.
2. The system tells the user that the details are incorrect.
3. The system asks the user to re-enter the details

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition.

### Requirement 2 < Enter Profile >

#### Description & Priority

This is when the user enters their profile on the Web Application. Here they will have the option to edit their account.

The contractor will have the ability to view the messages sent to him.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the Profile works on the Web Application.

**Flow Description**

**Precondition**

The user is logged in.

**Activation**

This use case starts when the user clicks on “Profile” in the nav bar.

**Main flow**

1. The user logs in.
2. The user clicks on “Profile”.
3. In the Profile the user can see all the profile details.
4. Some of these details are on show in the Contractor Directory.

**Exceptional flow**

1. On Entering the Profile the user can edit their profile.
2. Here they have a form of details they are allowed to edit.

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition

### Requirement 3 < View Contractor Directory >

#### Description & Priority

This is the ability to see all the contractor profiles.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the “View Contractor Profiles Work”

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when

**Main flow**

1. Enter the Contractor Directory section.
2. View the Contractor Profiles and the contact details.

**Exceptional flow**

E1:

1. Enter you search query in the search bar.
2. The search bar will take you to the Contractor Directory.

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition

### Requirement 4 < Enter Notice Board >

#### Description & Priority

This functionality allows the user of the Web Application to enter the Notice Board where they can see the Job Posts on the notice Board. This is open to the public.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the Notice Board section works.

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when a user clicks on the notice board section

**Main flow**

1. A user enters the web application
2. A user clicks on the notice board section.
3. They enter into this section.
4. The user can see the posts by the public looking for a contractor.
5. The user will have the ability to enter that post and leave a comment.

**Exceptional flow**

E1:

1. A user can write a notice
2. That notice will have the same functionality as the other posts.

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition

### Requirement 5 < Help >

#### Description & Priority

Here the user will have the ability to send the administrator a message and to view the FAQs.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the Administrator Help section works.

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when a user clicks on the Administrator Help section

**Main flow**

1. The user enters the Administrator Help section.

**Exceptional flow**

E1:

1. The user can send the administrator a message, they must enter their email here.

E2:

1. The user can see the FAQs.

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition

### Data requirements

For the purpose of this Website the user needs to give certain personal information about themselves to set up an account.

These are:

* Email
* Username
* Phone Number
* Address
* Bio
* Profession
* Date of Birth
* Are they a contractor/customer
* Password

This information is all stored in the user table and is allowed to be shown on certain pages. Secret info such as the password is never on show.

On the Notice Board, the user can put up a notice, but this is for public view and so the public can put up any information they deem fit.

### User requirements

The user should have the data requirements above to be able to set up a profile.

The site is free to use for everyone so the main user requirements would be for the user to have access to the internet upon which to look up our website and an internet connection.

## Design and Architecture

The Project was created on the AWS IDE. This allowed for a well formed documentation repository that was smooth and user friendly.

In the end most of the user data, which is the main body of the data, went into the user table in the schema.rb database as it provided a cleaner code and a more straightforward approach to coding.

The Website was designed to have a home page as the main page as in the site map below. Here the user would have a welcome page with some introductory information.

The nav bar here would be the same for all pages.

Next the user could go into the Contractor Directory from the nav bar and this allows the user to view the Contractor Profiles.

The next button on the nav bar is the Notice Board. Here the user can see notices and put up new ones. A further functionality of this page is that the user can enter the posts and leave a comment in reply. Here the Job Posts are stored on the topics table in the Database and the comments are stored in the comments.

The next class is the help page. Here the user can send the admin an email and view the FAQs.

The other options are to register, where the user will give info to form a profile or login/out.

Once a profile is created the user can edit the profile. Certain aspects of the profile will be visible on the Contractor Directory.

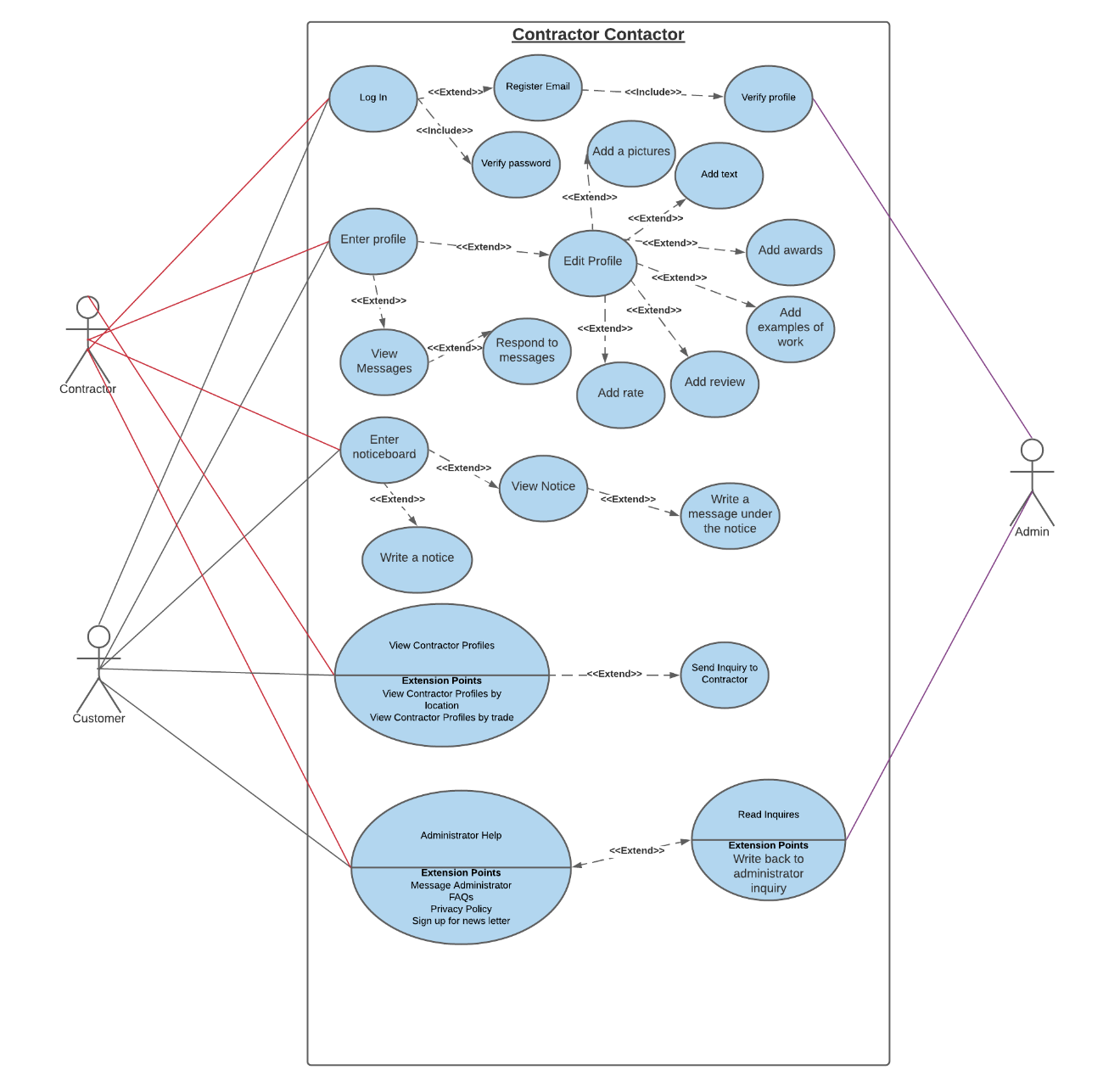


Fig 1. This is the original Use Case Diagram. Some of the functionalities have changes but it mainly the delivered project.

From this diagram the only changes are:

* That the Profile has a much more streamlined approach to details and no profile picture yet due to time constraints.

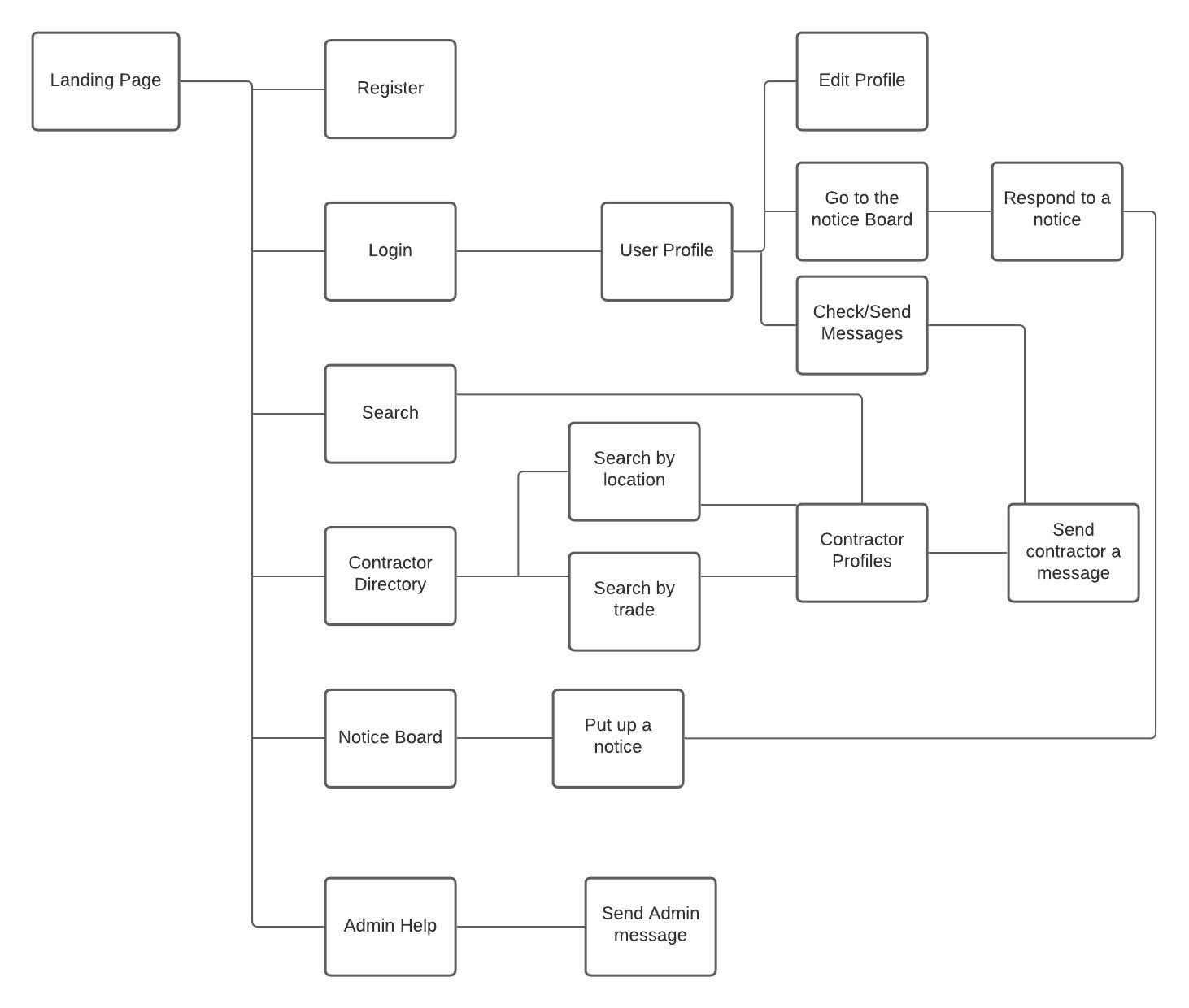


Fig 2. This is the original Contractor Contactor Site Map. Much of this map has remained the blueprint for the actual Project end goal.

The changes are as follows:

* The user profile can only edit but it is linked to the Contractor Directory.
* The user can only search for the term put in the search bar and it will bring the user to the Contractor Directory.
* The contractor can respond to the notice but not from the profile page.

This was all done on the AWS IDE, using its fantastic documentation broken into different classes. Manipulation of the project was rather easy due to this fact.

The languages were HTML, CSS, Ruby on Rails and a small bit of JavaScript.

## Implementation

We started this project by creating a new Directory in AWS. This is done with the following code.

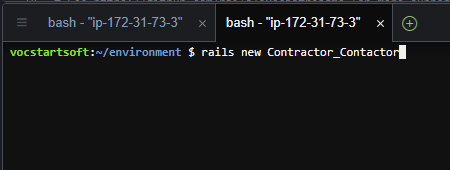


Fig 3. Creating the repository.



Fig 4. This is an image of what the browser shows after you set up initially on Ruby on Rails.

A gem was inserted here followed by a bundle install to update the gem. “'sqlite3', '~> 1.3.6'” was to allow login and out functionality to the Website.

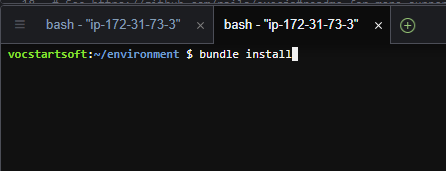


Fig 5. This is the command to install the gem.

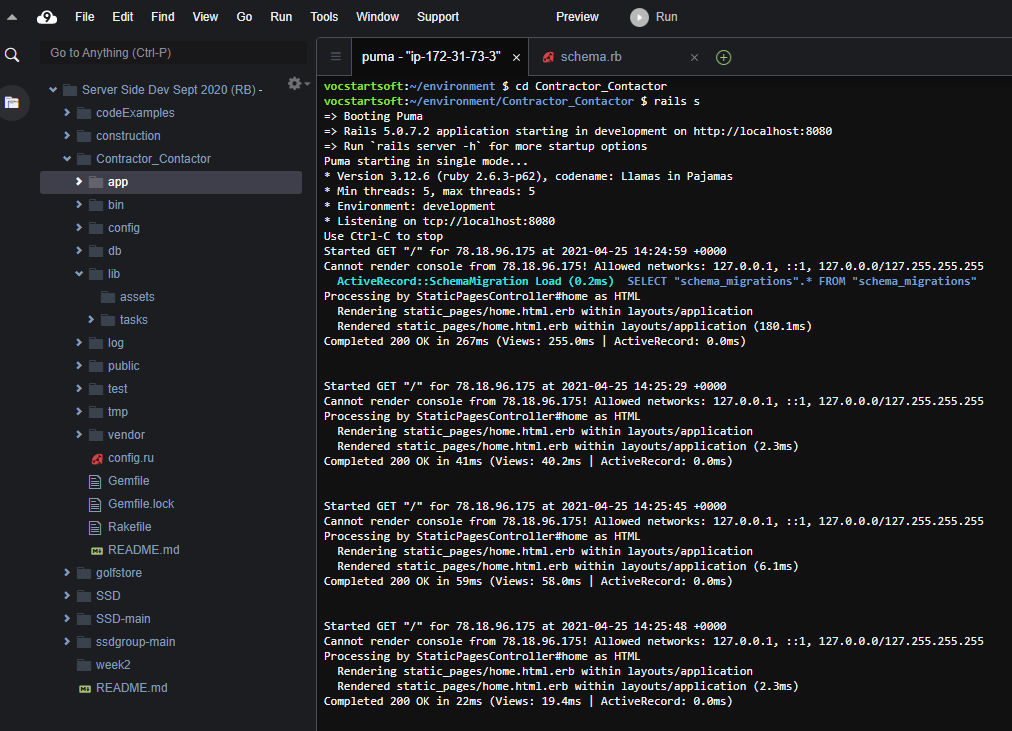


Fig 6. This is a view of the AWS Project Folder layout of the Contractor\_Contractor repository.

There are the stylesheets where the CSS is written. These provide the front end with colour, location, font size, and a more human touch. The application.css is the main class here and contains nearly everything. It appeals to the differently named tags in the HTML. Certain functions in here involve the layout of submit buttons and colour changes when a mouse is dragger over a certain button as in the submit or nav bar.

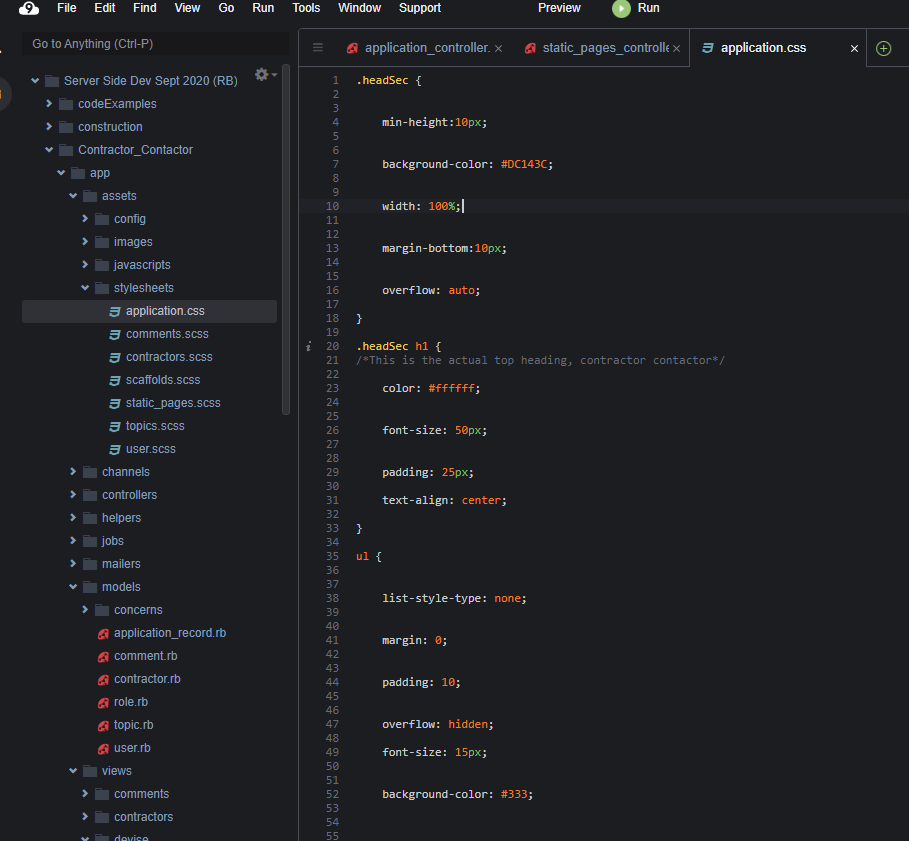


Fig 7. This is a picture of the application.css class where most of the css is stored.

The controller classes (Learn Ruby on Rails - Full Course, freeCodeCamp.org, 2020) were the crux of the application controlling how the client would communicate with the rest of the system. There was also functions here which restricted users to certain information. One key controller here is the application\_controller.rb which permits certain columns from the tables in the schema.rb to be used by the users when registering, for the profile and for the notice board for example. Other key tables are the user\_controller.rb which allows a login and out function and allows all users to see their profile information in profile. The topics controller here is for the Notice Board. The static pages controller allows the nav bar names to route to other pages. The comments\_controller.rb allows comments from the comments table in schema.rb.

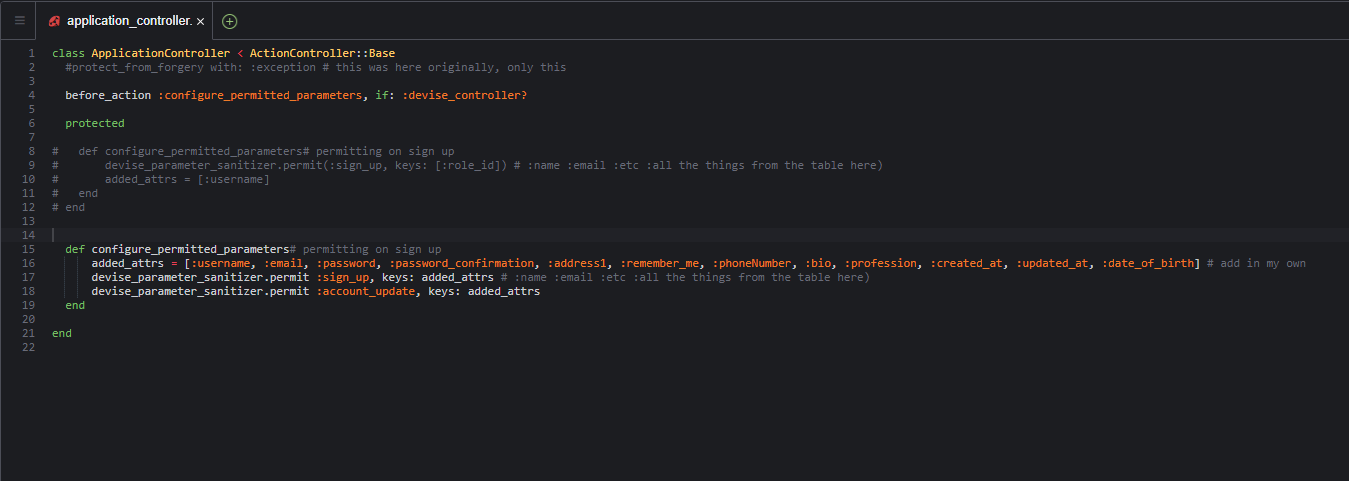


Fig 8. This is the application\_controller class.

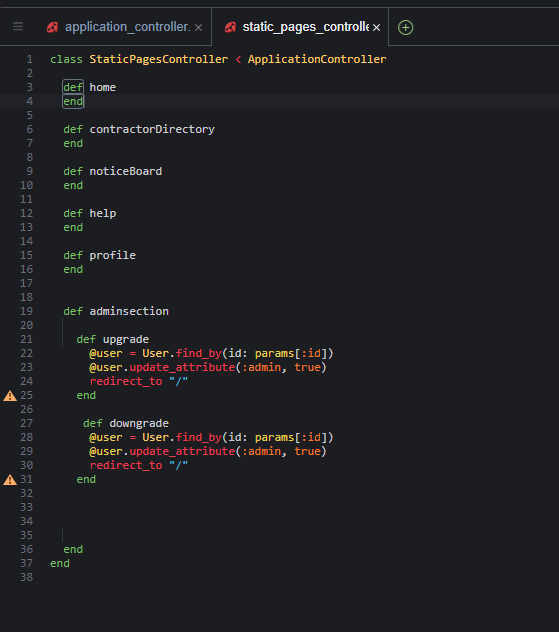


Fig 9. This is a photo of the Static Pages Controller.

Next are the model classes where information on the tables is stored and declared. You can make changes here to which users have access to what information. Declaring info about classes, assigning data from table to table. (Learn Ruby on Rails - Full Course, freeCodeCamp.org, 2020)

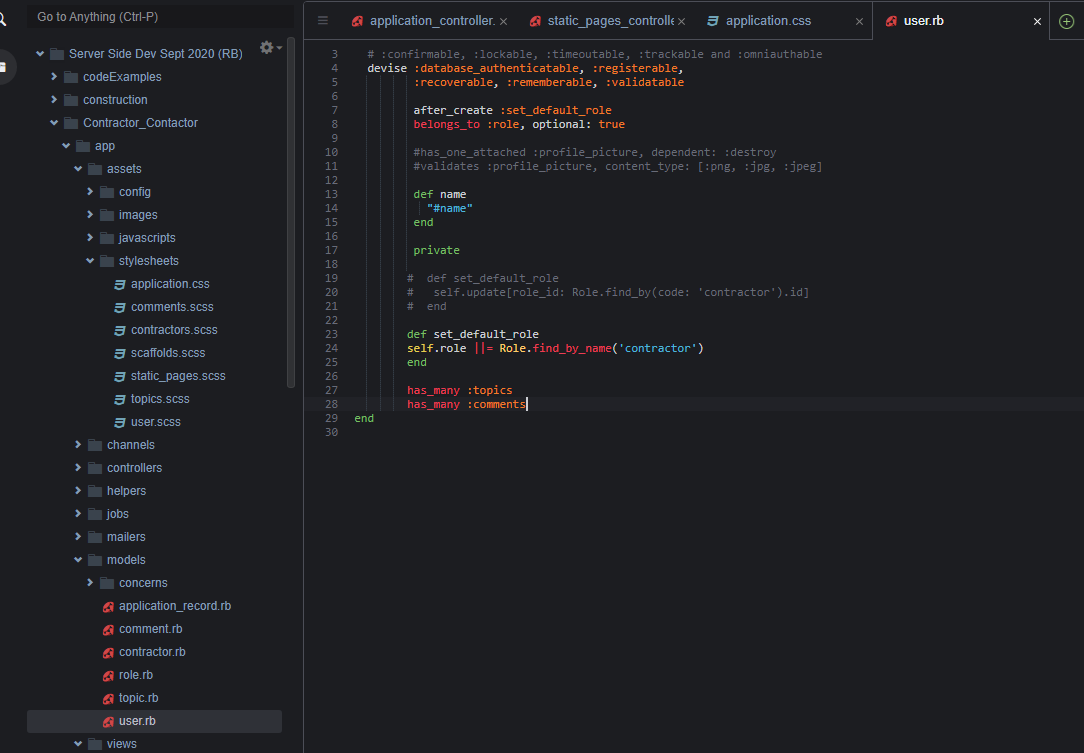


Fig 10. This is an image of the user.rb in the models folder.

Following this is the layouts. Here application.html is the main body of work. There is HTML in this as well as some javascript. This is the ultimate HTML page for the site. It has the layout of the entire website. It has the heading, the nav bar, with their links to the routes.rb file. There is a searchbar here which reverts back to the Contractor Directory. At the end is an extensive footer split into boxes for the divs of the info included there. There is an about us, social media links and a contact form. (Fully Responsive Footer Section using HTML CSS, CodingNepal, 2020)

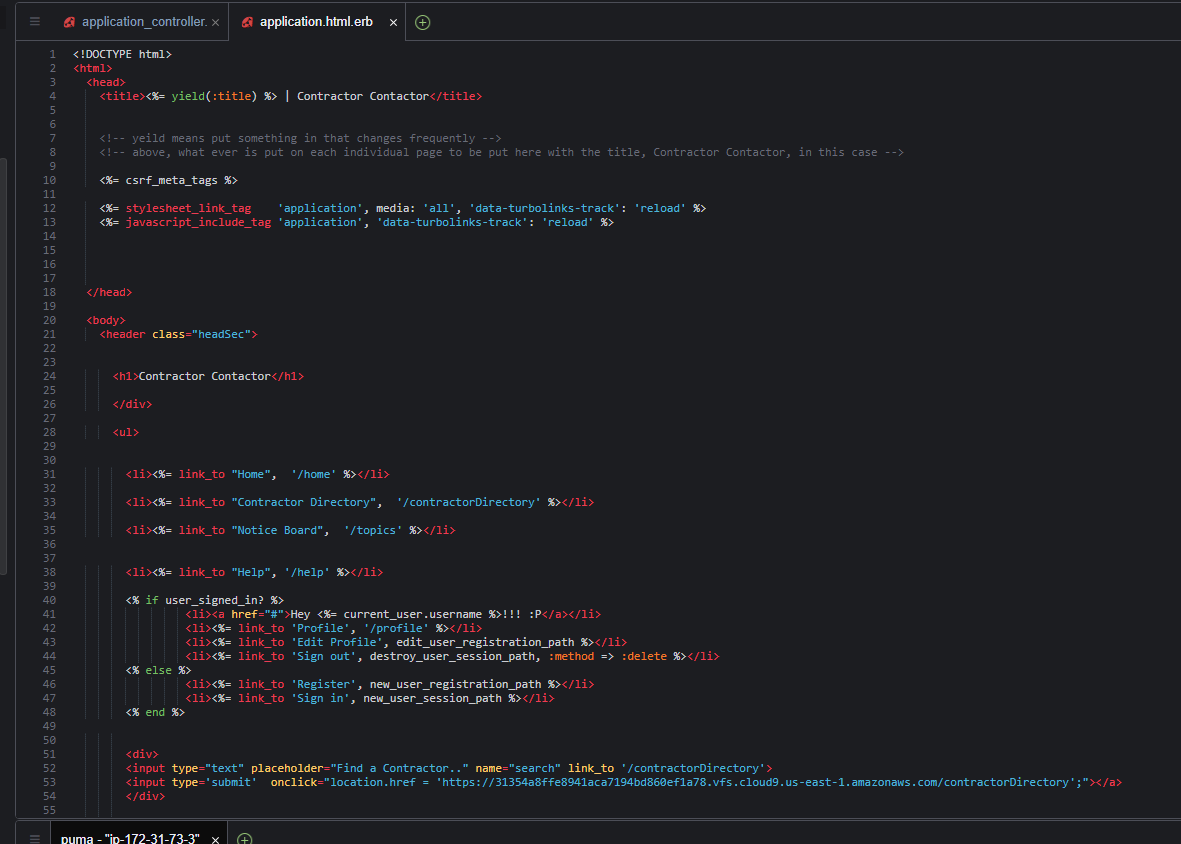


Fig 11. This is a picture of the top section of the application.html.erb class.

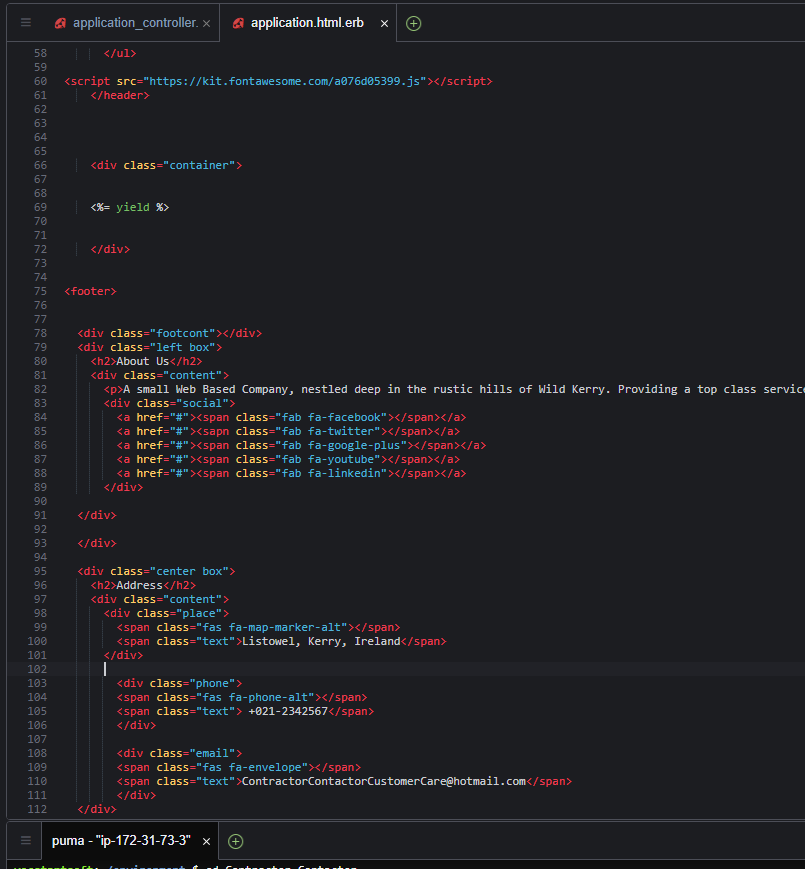


Fig 12. This is a picture of the bottom section of the application.html.erb class.

Next are mainly the static\_pages and topics folders for the HTML. In the static pages the html was created to form all the pages. One of the most extensive pages here is the [home.html.erb](http://home.html.erb). Here the user can get a good sense of the website and how the other pages fit into the structure of the application.html.erb. The structural information is spread out in these classes. All of these pages are mainly body for the different classes. They have a link connected to the routes which allows them to be accessed by the user who clicks on the nav buttons. The contractor contactor allows the user to see certain allocated information, from the database which is the schema.rb, in the “Contractor Directory” class. (Allow Users to View Profile and List All Users, GitHub.com, 2019) Next is the Notice Board. Here the user is allowed to put up job notices with info from the person who posted and there is the option to comment under these posts. Next is help with a contact form to allow the user to contact the customer care/admin, underneath this is the FAQs. Next is the register and the sign in. The user must insert info into the users table which is in the schema.rb. To allow this to happen the developer needs to go into the application controller, and allow certain columns to be accessed by the user. Once registered and signed in the user will see the username in the nav bar as well as a new option, a Profile. (AWS + Rails: How to Add Devise Users and Admins to a Rails App, BrainTrust Digital, 2020) Here the user will see certain information about the profile which will be shown in the contractor directory. From here the user can edit the information, there are two options for this in the nav bar and underneath it which will bring the user to an edit form. (Learn Ruby on Rails - Full Course, freeCodeCamp.org, 2020)

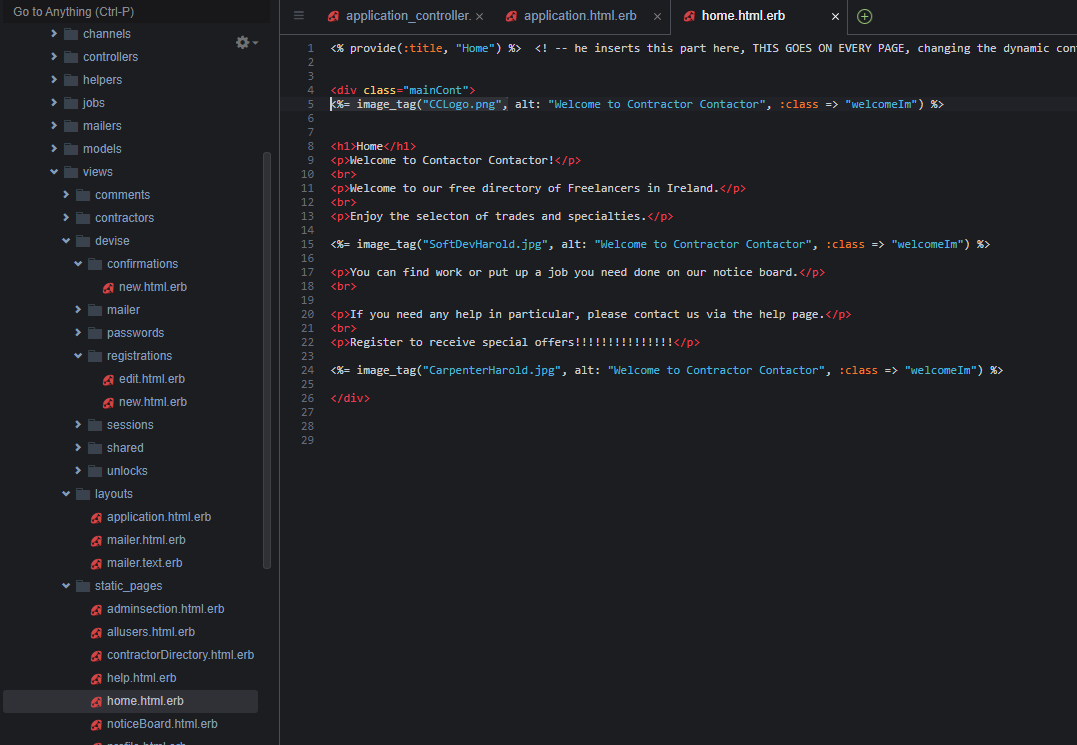


Fig 13. This the Home.html.erb class

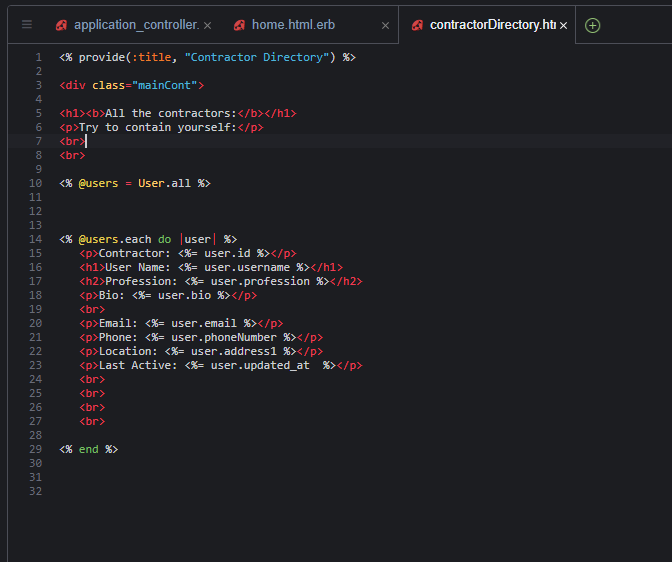


Fig 14. This the the contractorDirectory.html.erb page

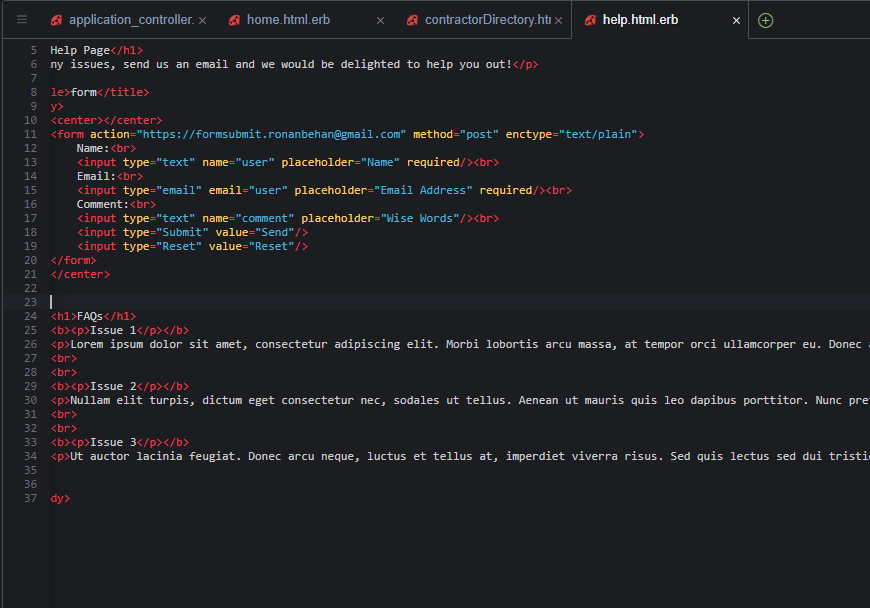


Fig 15. This is the help.html.erb page

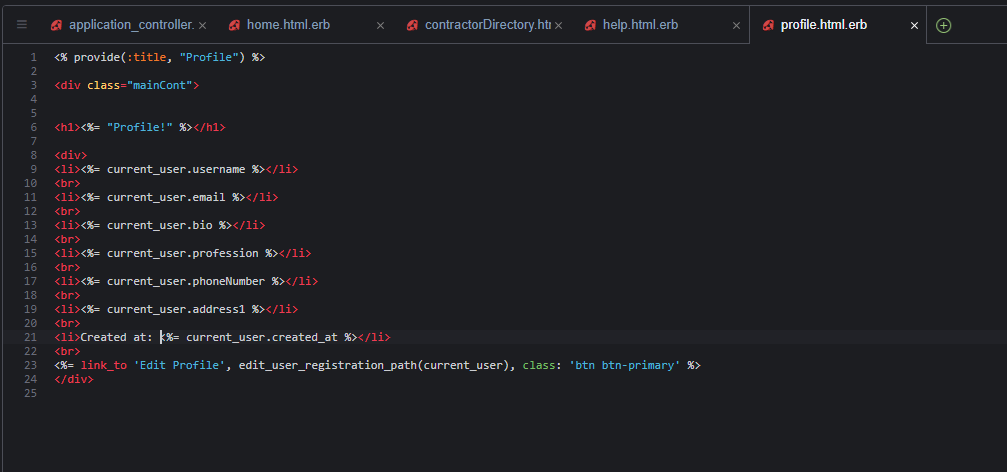


Fig 16. This is the profile.html.erb class for the profile page.

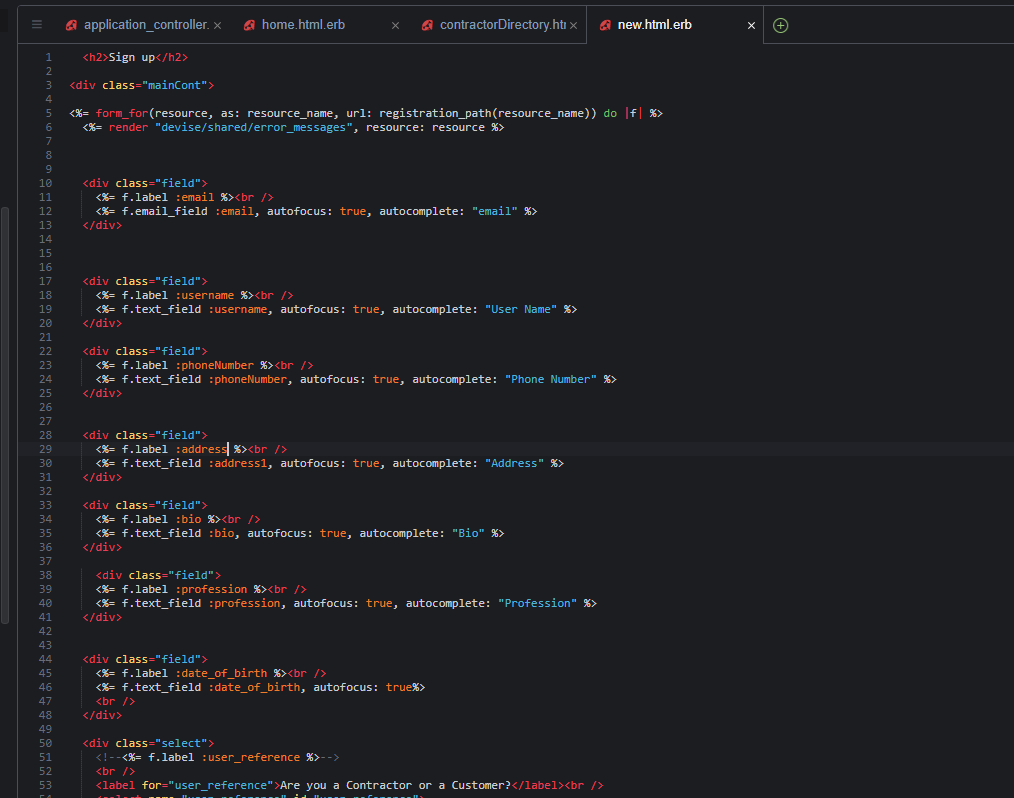
Fig 17. This is the top of the new.html.erb class. This is the registration page.



Fig 18. This is the bottom of the new.html.erb class. This is the registration page.

The index.html.erb was the main class of the topics folder. This was made for the functionality of the Notice board which allows people to come and leave a job on the noticeboard with information on the role and through the topics folder the user can edit a post, the user can leave a comment to a post and put up more for the public to see. The user can only see what the person who posted wanted the public to see.

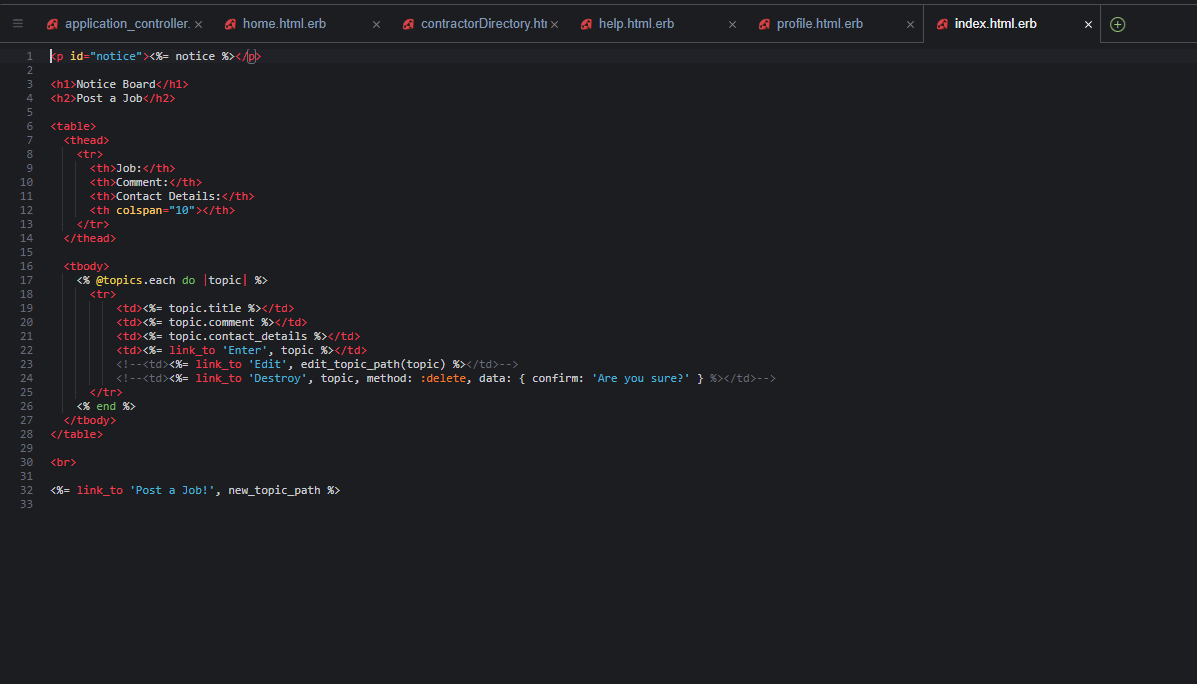


Fig 19. This is the index.html.erb page. This is where the notice board is.

In the routes.rb file there are all the routes. This is in the config folder. There are lots of get requests here for options to take on the Html pages as the nav bar routes. There is also the root to the home page which makes this the started page for the Website as well as resources which allow certain table columns to be accessed by the user for example.

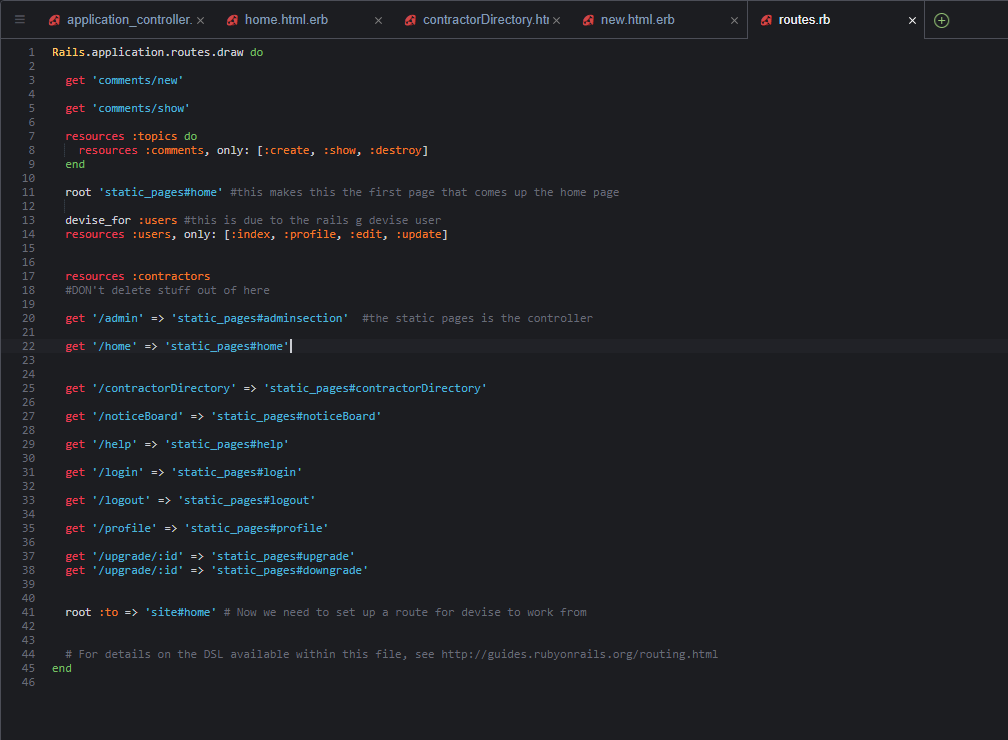


Fig 20. This is the routes.rb class.

Next there is the migrate folder. This is in the db folder. This is where the developer added to the tables in the database schema.rb.

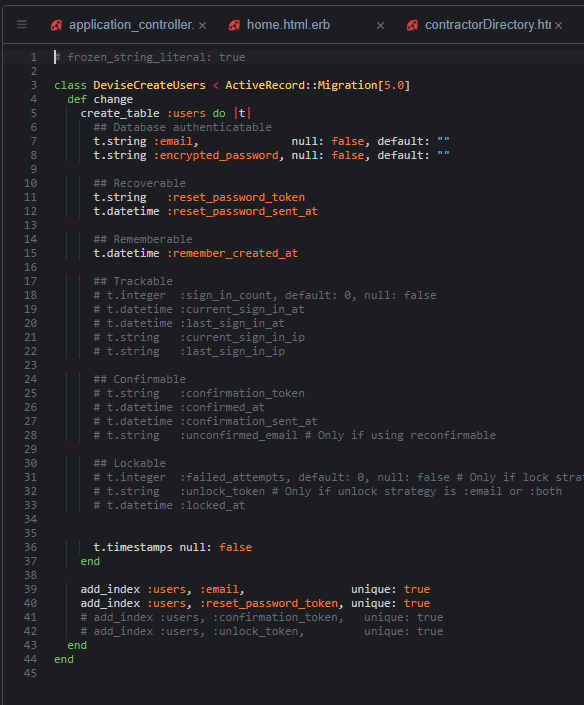


Fig 21. This is the 10210409165617\_devise\_create\_user.rb class in the migrate folder.

Here the developer created tables. This was done by adding to them by the Terminal. This was done through through Ruby by ordering a scaffolding to be created in the following code example.

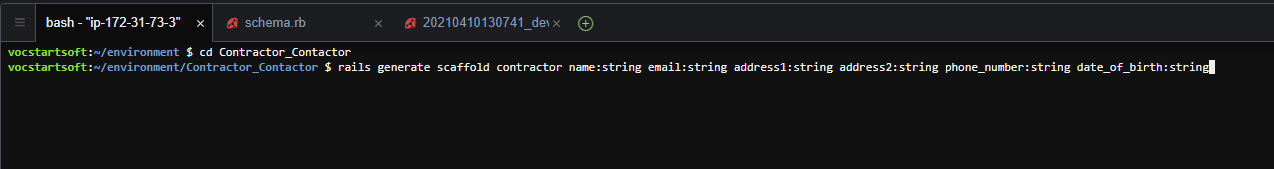


Fig 22. Scaffolding of a table in the terminal.

That’s followed with the next code example.

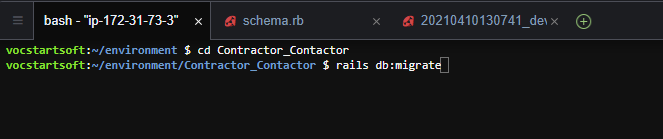


Fig 23. Migration of the scaffolding to create the table.

This db:migrate is to create a database by producing a model representing a database table for our entries. Here is how we add a column to that table.

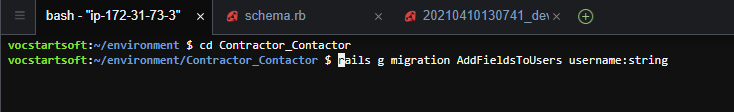


Fig 24. Adding fields to the User table

This is proceeded by the next rails command.

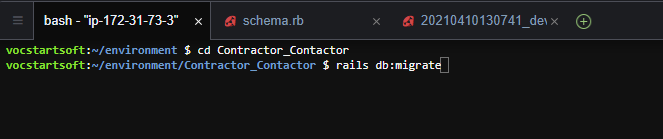


Fig 25. This is to migrate the new fields to the User Table.

This adds the column to a table.

The schema.rb class is one of the most important for this project. Without it, it would be severely limited. Here is where all of the tables are stored. In the end the most used tables are the “users” which has an allows data to be stored for use on the Contractor Directory, Profile, Register, Sign Up, Login/out by the user. The other main tables are the “topics” table where the topics, the job posts, are put up for public view. The other table is the “comments” table used for commenting on the topics put up in the Notice Board, known there as job posts.

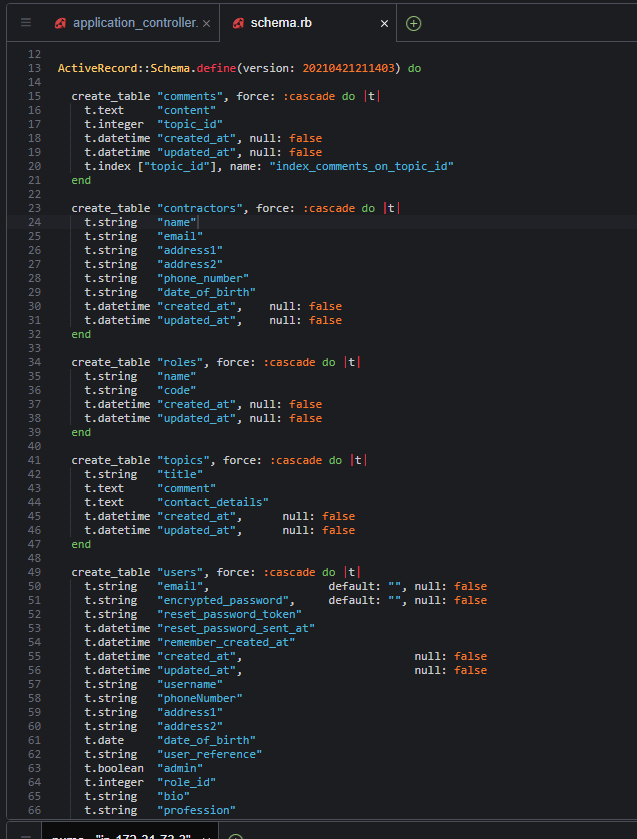


Fig 26 This is the schema.rb file. This is where are the tables are stored.

Gemfile. Here we started by changing the Gemfile to “gem 'devise', '4.6.0'” and this allowed the project to have the login and out functions for the application. Gems are added here to ruby to allow for further greater functionality. (Using Devise in Your Ruby on Rails Application, Miguel Angel Dubois, HackerNoon, 2020)

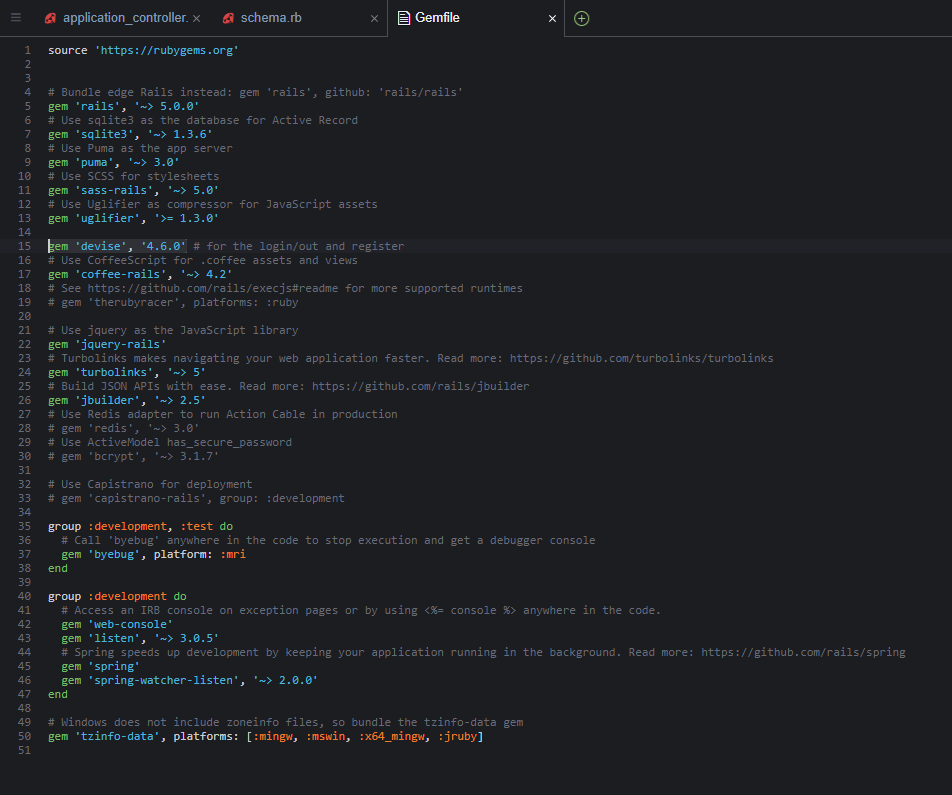


Fig 27. This is the Gemfile. Here is where we add gems for better functionality.

The Terminal. The terminal is essential to AWS and the running of this application. Access the file in the repository with the following command: “cd Contractor\_Contactor” and then run it with a “rails s”. This must always be done, this to get access to the correct project in the right repository.

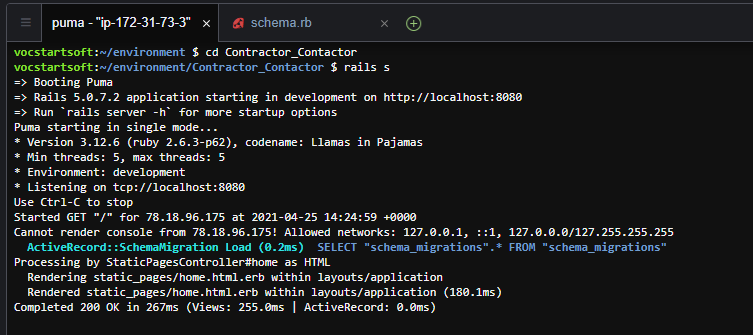


Fig 28. In the terminal is also where the scaffolding, added tables and further columns get added to the schema.

## Testing

Throughout this project there was an extensive amount of testing. Testing was conducted after anybody of new code in any language was entered. This amounts to some principals of Test Driven Development.

A copy of the Project was made before any major alteration. When progress and alterations were needed to be made the coding was done piecemeal bit by bit to make sure that every aspect worked in tandem and didn’t cause any unnecessary errors.

Much of the project was done with Black Box and White Box testing in unity. Code would be written. Checked for errors. If everything looked ok the program would be run. The browser would be checked to see if anything had altered. If the changes were non-existent the code would be checked again and tested again.

Here are some examples:

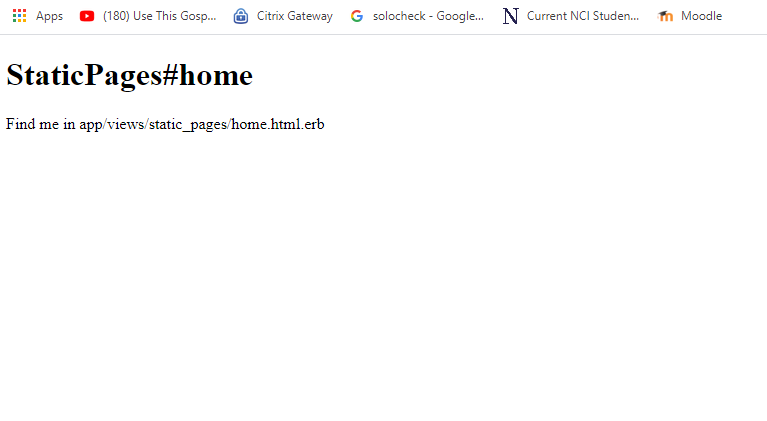


Fig 29. Here is the first static page.

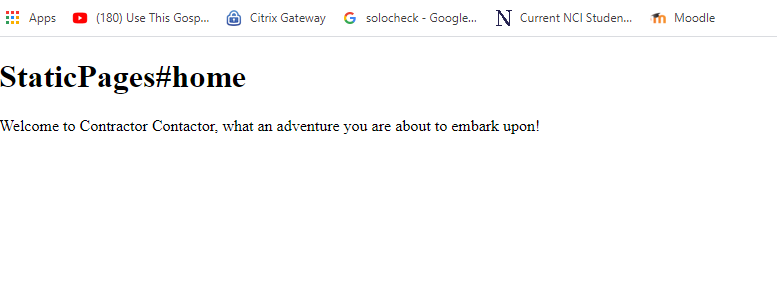


Fig 30. Here is the first change to this page. The writing has altered. In this manner the code would be written and double checked on the browser then.

Eventually errors would be encountered.

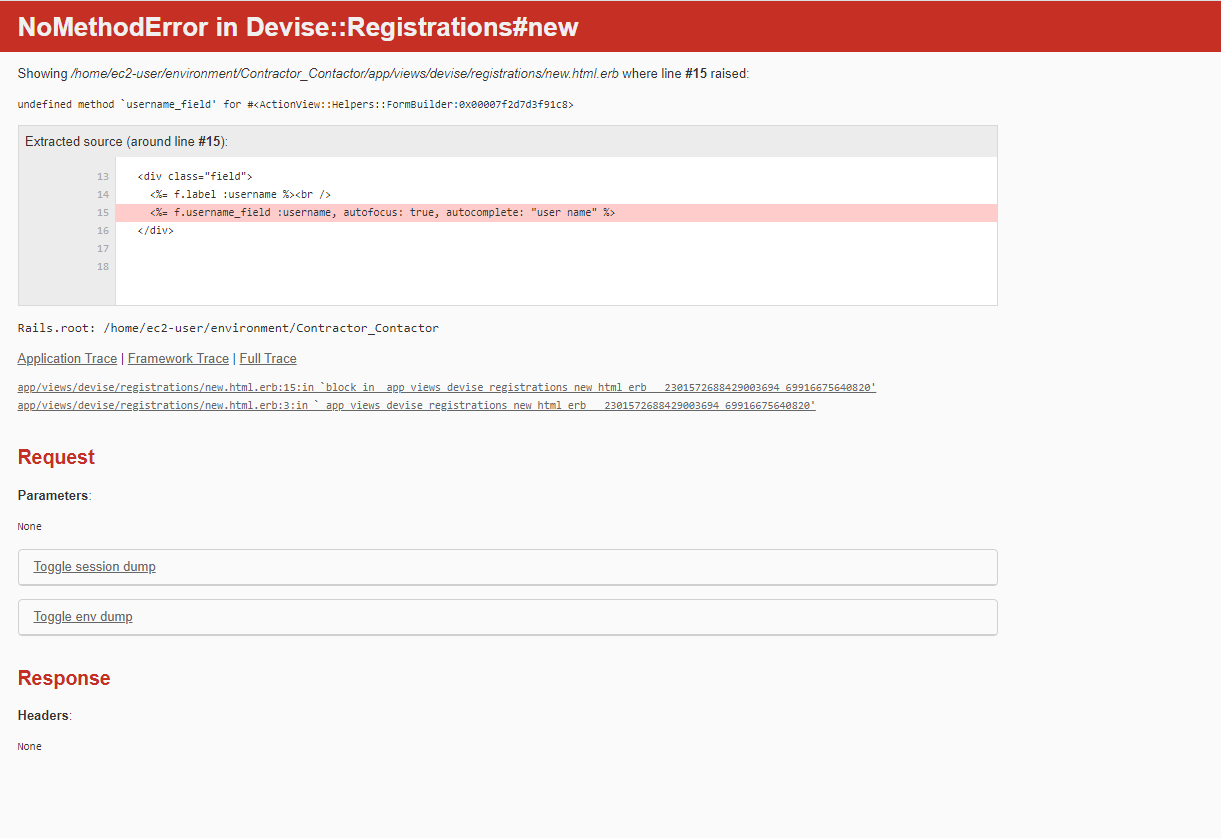


Fig 31. Error in the Code.

Errors called for a backtrack to AWS. Information would be taken from the error page to see what might be the issue. The code was checked with great care and precision following the MVC routes. Some errors were small and easily surmounted others called for a change in direction and a reload of an old save if the code was heavily resistant to corrections.

Database errors proved the hardest to correct. Some possible choices for tables have been inserted in the schma.rb and are visible however attempts to removed them were unsuccessful and attempts were rolled down due to time contraints.

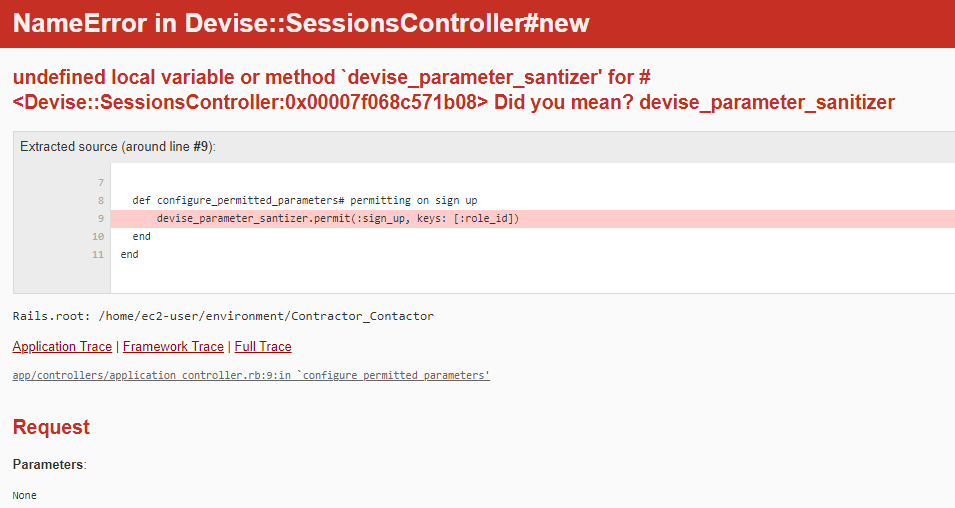
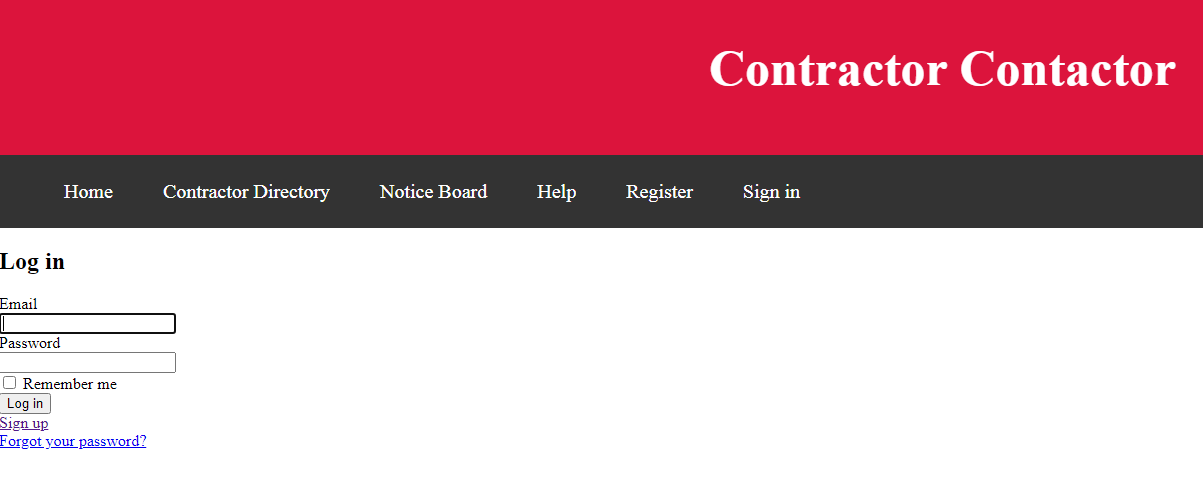


Fig 32. Syntax error regarding data information.

This was the process of testing as the project continued. Minor changes followed by saves. This went on to a run function. The browser was checked to see if the code was correct. If the code was incorrect AWS would be opened again and gone through with a fine tooth comb. Checked again. Saved again. Run again. The functions on the browser were check regularly. This was the manner of testing throughout the project.

In the end white box testing took over as the project was largely created. The usability and the functions of the website were checked over and over to make sure that it was user friendly. That the website functions worked quickly, that it was uncomplicated and straightforward and aesthetically pleasing. Every aspect was checked and reviewed.

Here is an example:

  
Fig 33. In this example the developer of the website is trying out the login function to see if the nav bar changes to look as intended previously.

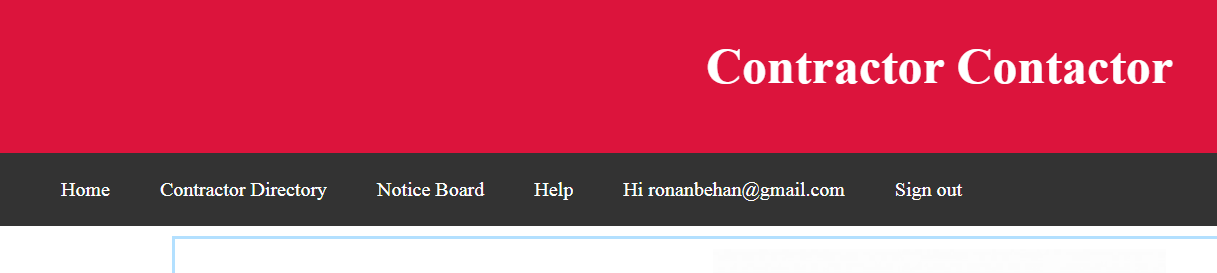


Fig 34. This is the nav bar looking as intended by the test above.

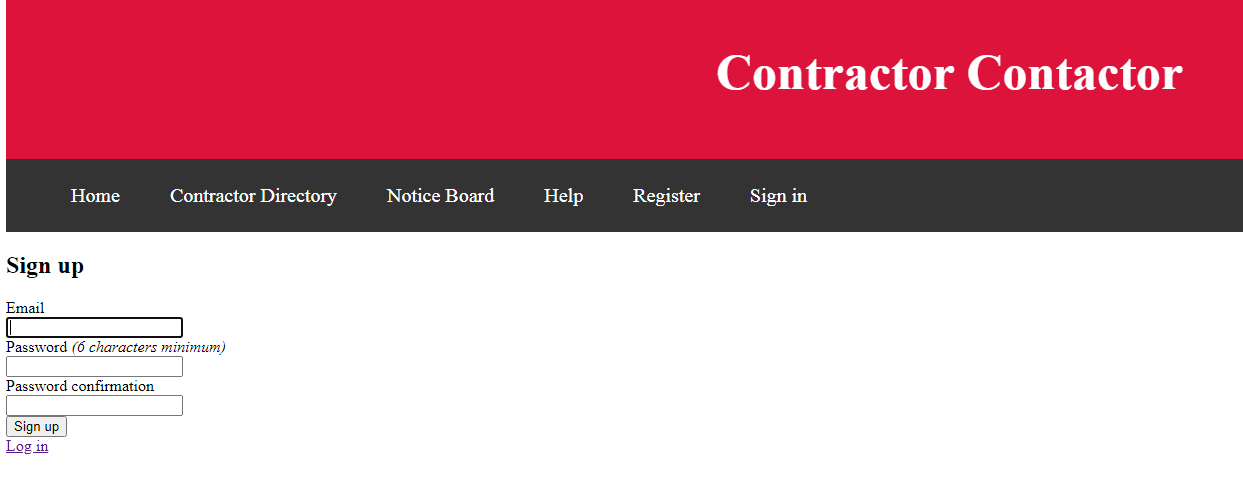


Fig 35 This is a test of the registration. The early registration form.

## Graphical User Interface (GUI) Layout

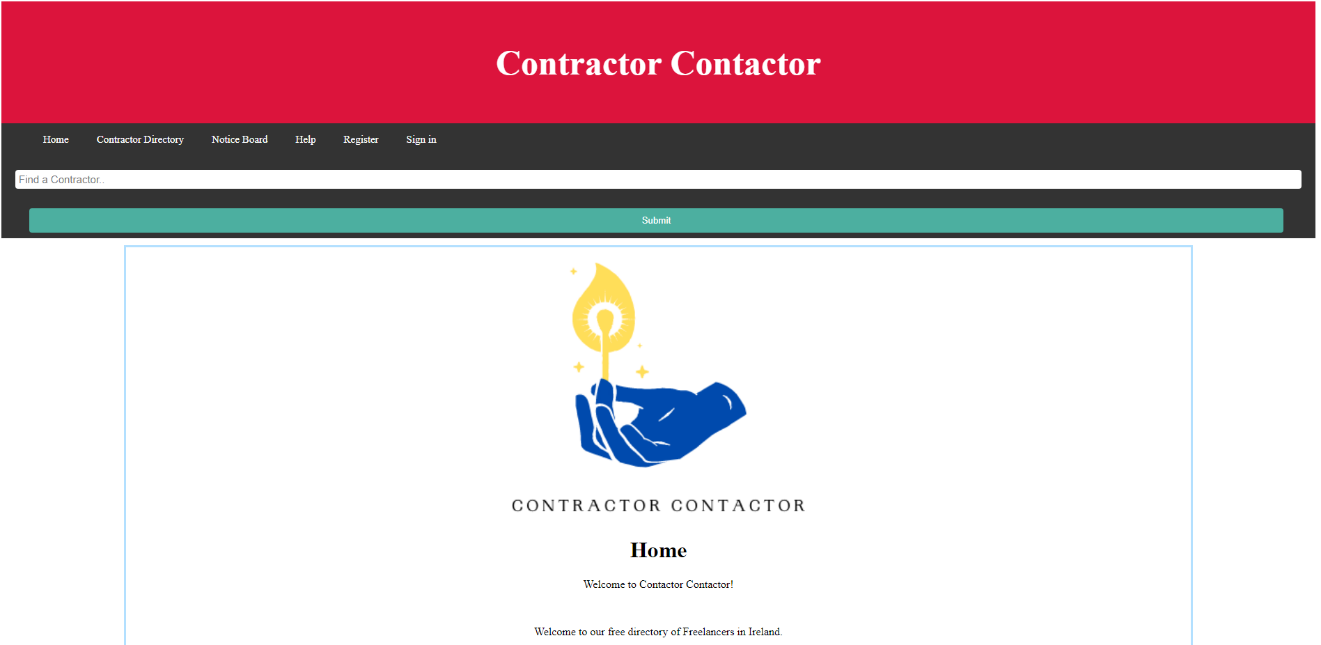


Fig 36. This is the top of the home page.

We have the name of the Website in the header. You can see the nav bar with most of the fuctionality there for access to the other pages. Underneath the nav bar is the Search bar which will revert you to the Contractor Directory.

Under this is the body for the Home page. The Logo plus some welcoming information.

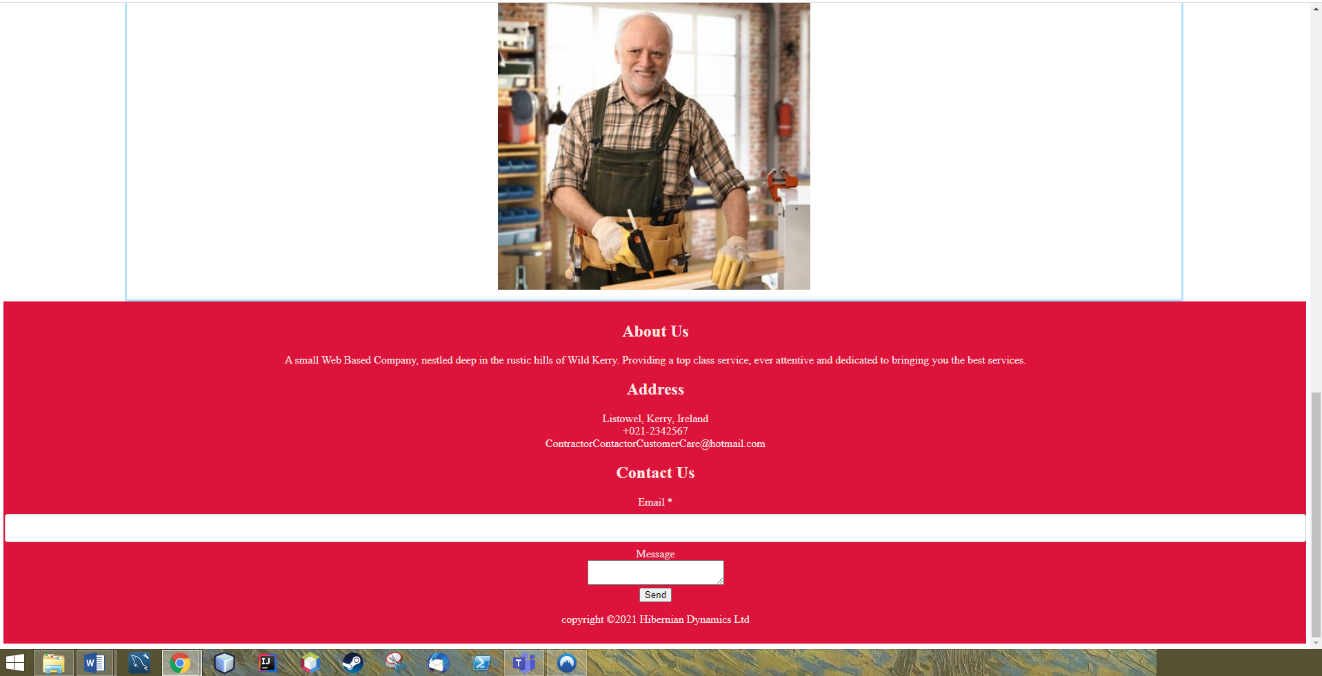


Fig 37. This is the bottom of the Home page. There are a few pictures in the body followed by the footer. Here there is some information about the company as well as a contact form.

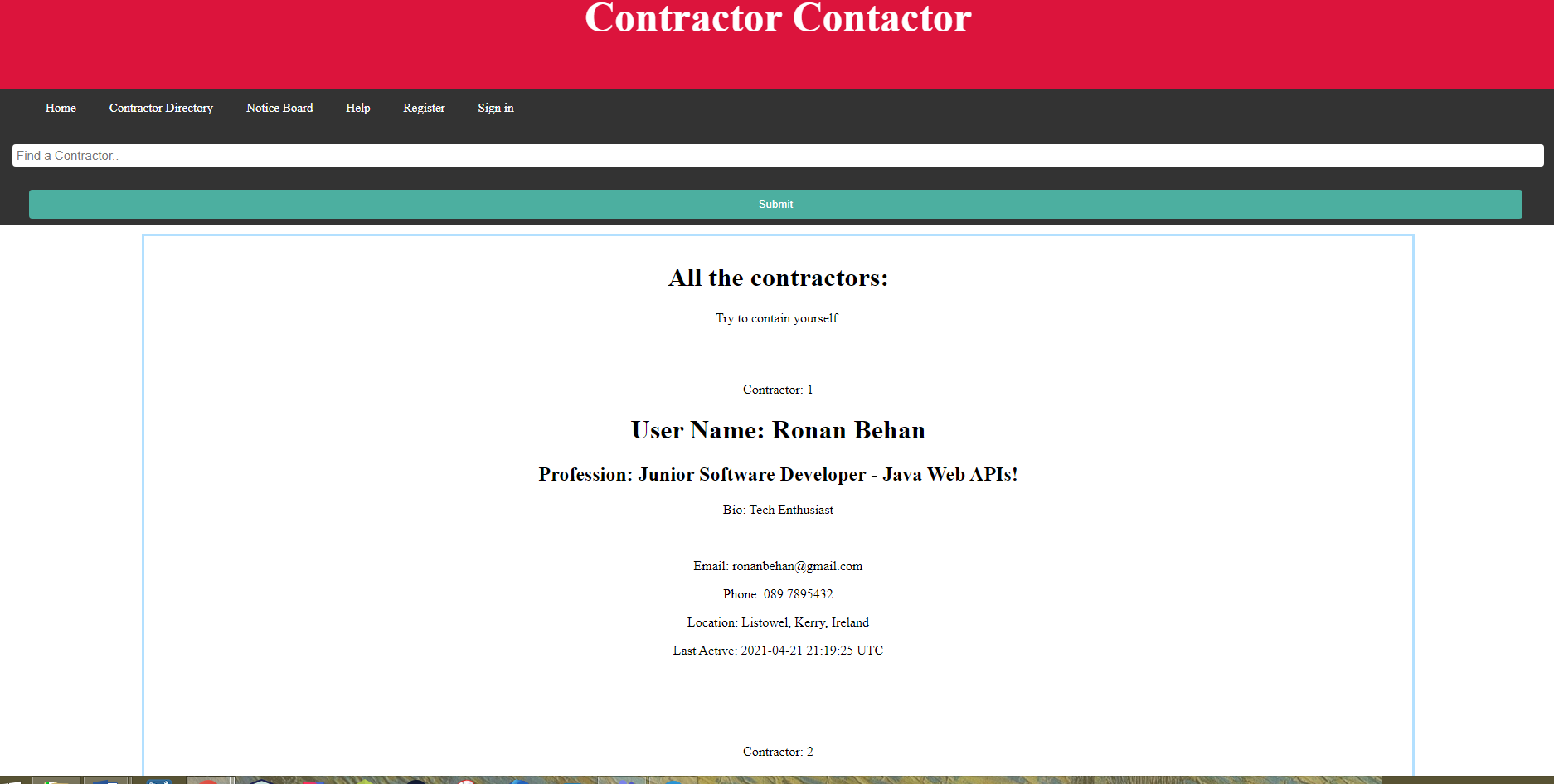


Fig 38. This is the Contractor Directory. Here is where certain information of the profiles go for public view. This can be edited in the profile.

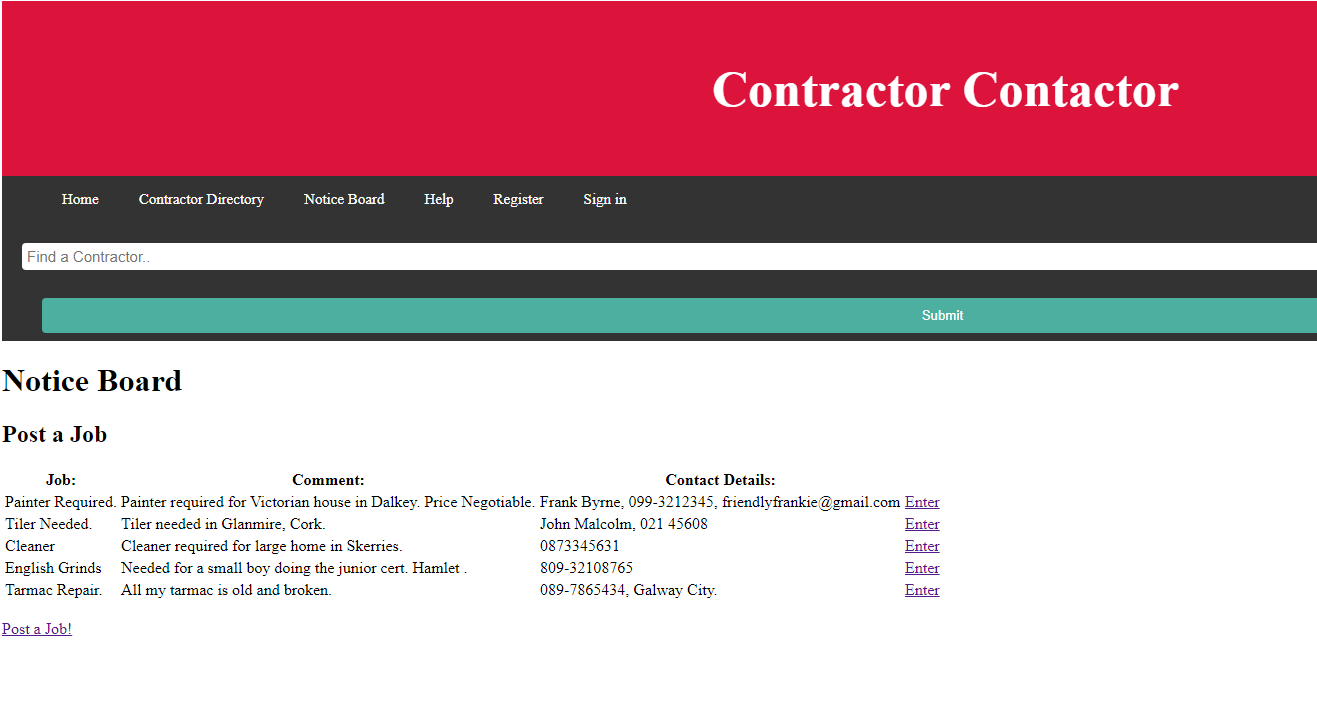


Fig 39. The Notice Board

This is the Notice Board where new notices can be put up for public view. You have the ability to enter into these posts. The user can leave some information that they want a contractor to see. See the next Fig.

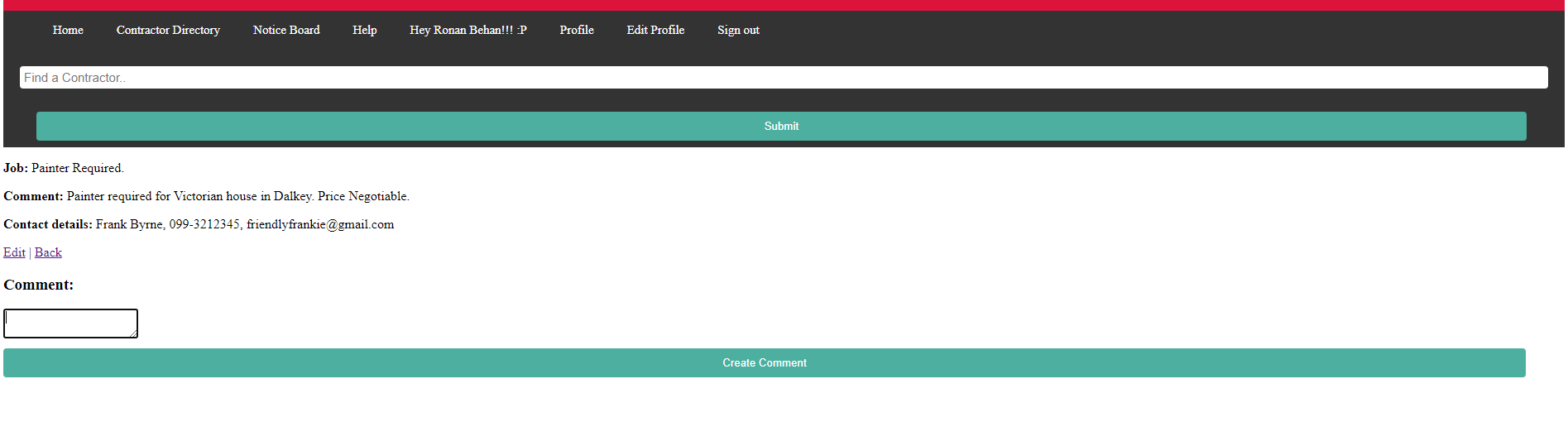


Fig 40. Here we have the opportunity to post a comment under the Job Post.

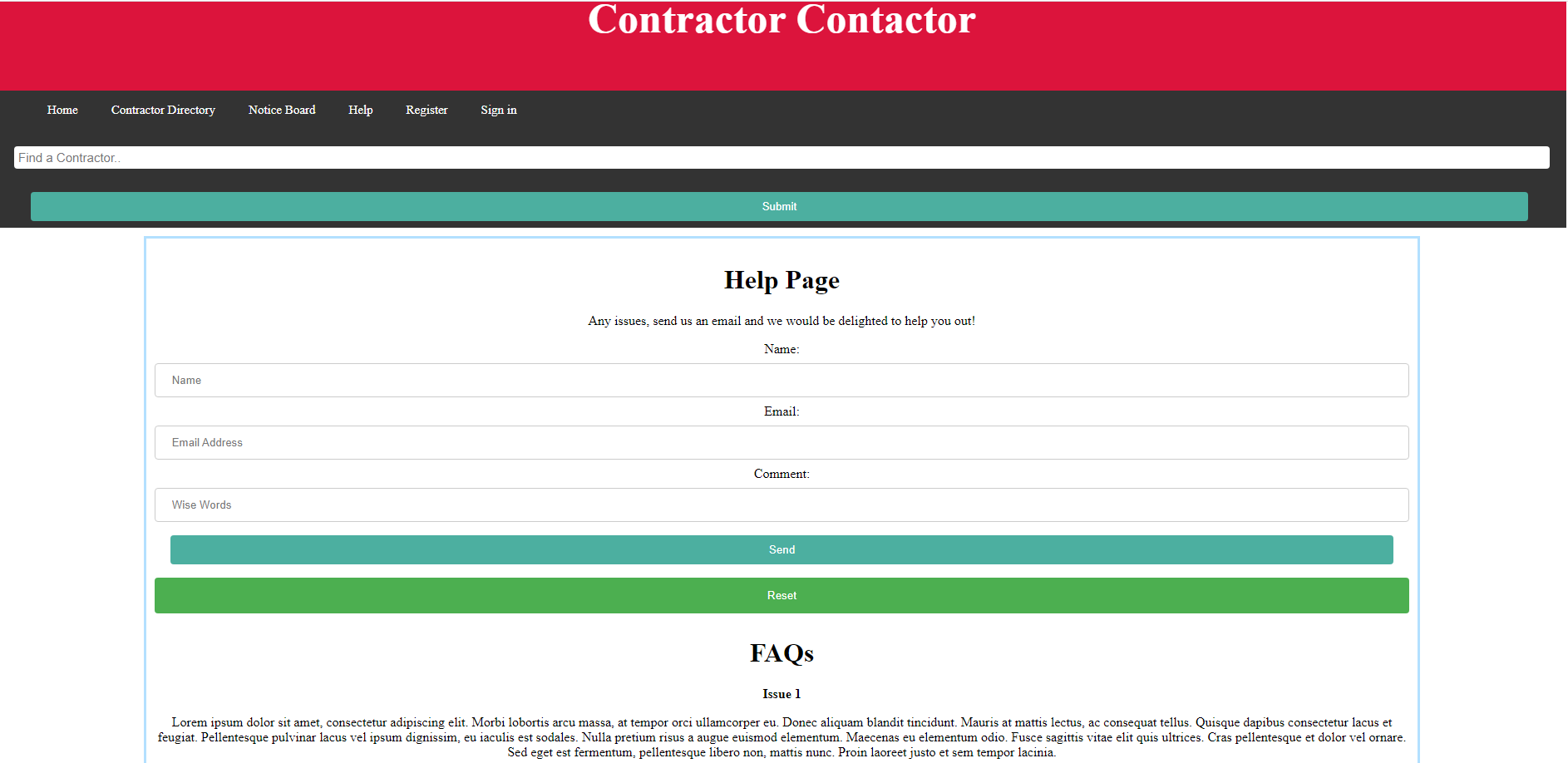


Fig 41. This is the Help Page. Here is another contact form for customer care. Under this is a set of FAQs.

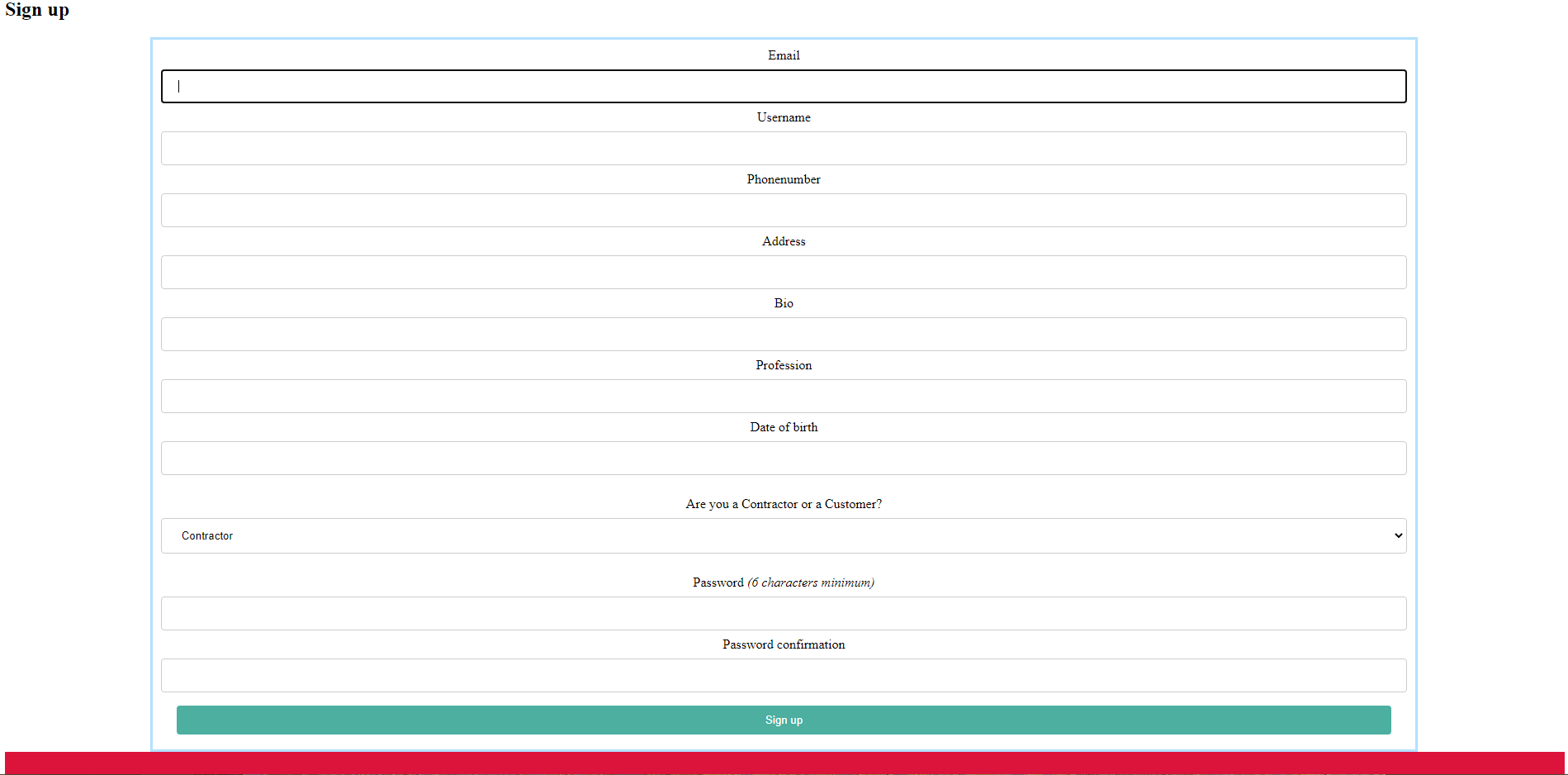


Fig 42. This is the register page. It takes all of the information and stores it in the user table for use in other pages when the user is logged in.

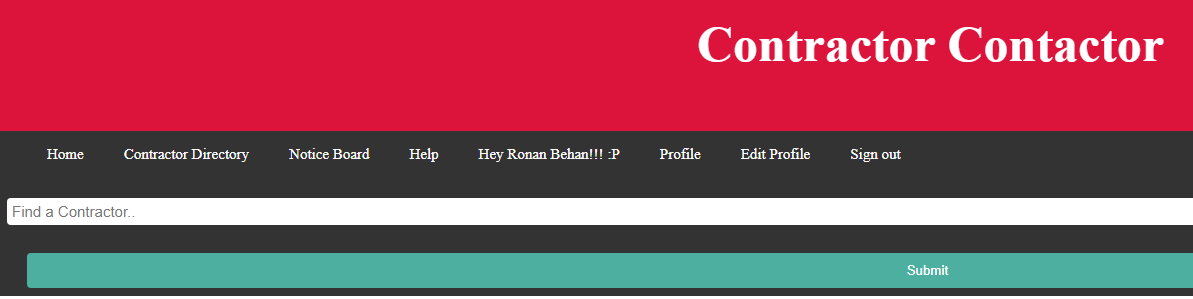


Fig 43. This is a view of the Nav Bar.

See the changes since the user has logged in. It now greets the user by the username. There is a profile option where you can set and edit your profile. There is an edit Profile and a sign out.

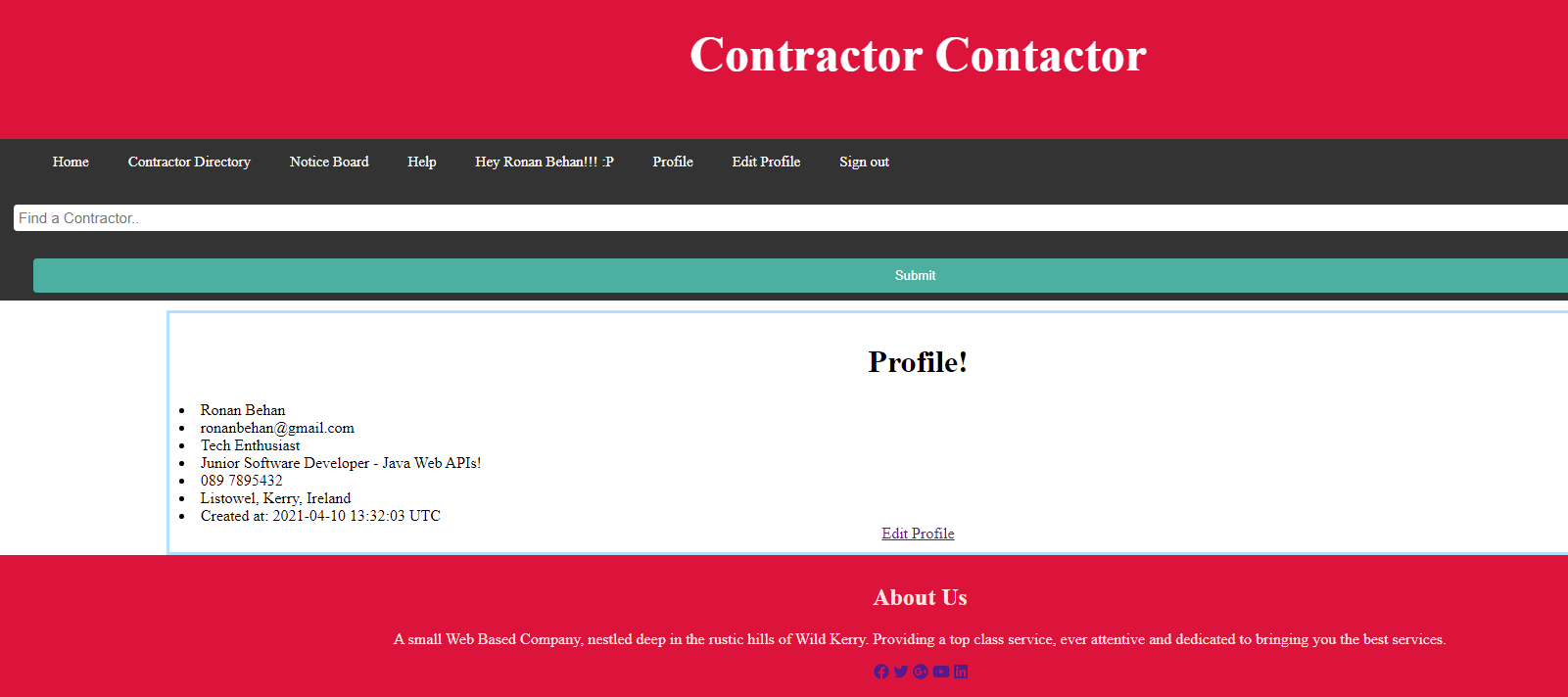


Fig 44. This is your Profile. Here you have all the info that the user is allowed to see. They can edit this information. Most of which is on view in the Contractor Directory.

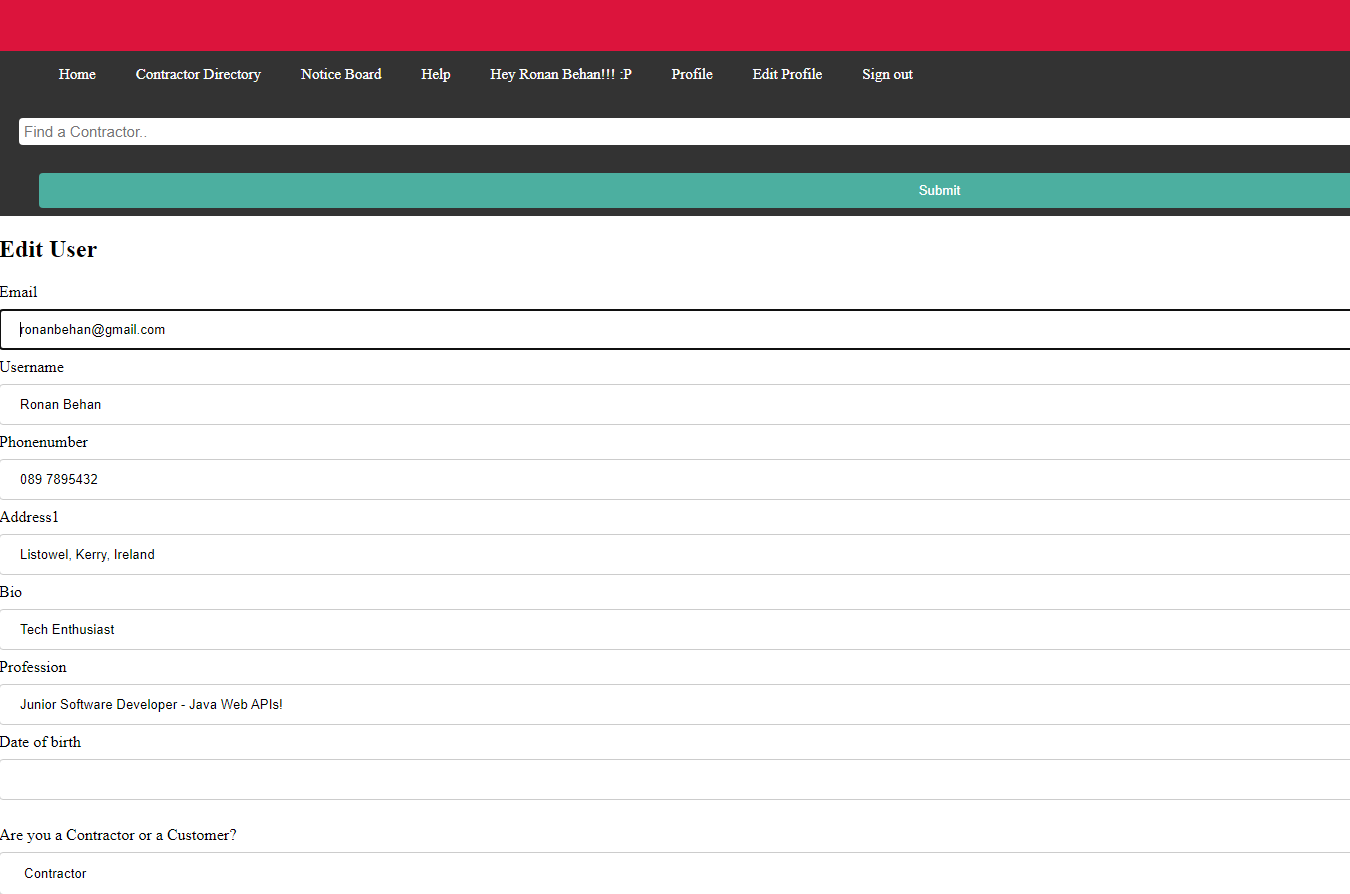


Fig 45. This is the edit page for the Profile information stored in the user table.

## Customer testing

The Website was tested by four customers.

The first test customer reviewed the site and found it to be ok but thought that the font should be larger and that the site could be more aesthetically pleasing.

The second test customer thought the site was very user friendly but thought it could have used more imagery and bigger font.

The third user thought the usability of the website was very good, that it had great UX but called for more images and more categorization in the contractor Directory.

The fourth user thought the font ought to be bigger, more images were required, that the search bar be more useful in searching, that categorisation was required in the contractor directory along with locations, that there be a profile picture and that the Home page have more information about the company.

# Conclusions

Contractor Contactor is an online Contractor Directory. This alone if done well has a market opportunity.

The website has some work to be done with the GUI to make the website more immersive, interesting and appealing by users. This could be done with more imagery and imagery that changes regularly.

The contractor Directory could use categorisation and for a maps function to allow for locations to be checked. This would reduce the amount of unnecessary information viewed by the user which would make it a lot more user friendly.

The profile needs a profile picture, it also could use an internal messaging system.

Finally security needs to be better, on which user can see what and what they can have access to.

There are a few changes that are required to make this website market ready but it has made great strides towards that goal. It’s straightforward, user-friendly, and free. A one stop shop to find a contractor in Ireland along with a notice board. There is a gap in the market clearly shown by other major companies, and the Irish market is ripe for such and opportunity. There is great potential here for a real design for future projects.

# Further development or research

This project has been an excellent incubus for a commercial website of the same type. It stands that a Website of this description with better security and functionality, possibly through Java instead of Ruby, could be an able service for contractors in Ireland.

This project will inevitably lead to more Websites being created in earnest or for research value.

# References

freeCodeCamp.org (2020) Learn Ruby on Rails - Full Course, 9 Nov 2020. Available at: <https://www.youtube.com/watch?v=fmyvWz5TUWg&t=1866s>

HackerNoon (2020) Using Devise In Your Ruby on Rails Application, Miguel Angel Dubois, June 21st 2020. Available at: <https://hackernoon.com/using-devise-in-your-ruby-on-rails-application-a-step-by-step-guide-m92i3y5s>

BrainTrust Digital (2020) AWS + Rails: How to Add Devise Users and Admins to a Rails App, 3rd of May 2020. Available at:

https://www.youtube.com/watch?v=8UIG9Ggu9Q4

GitHub.com (2019) Allow Users to View Profile and List All Users, Yosef Benny Widyokarsono, 27th of October 2019. Available at:

<https://github.com/danweller18/devise/wiki/Allow-Users-to-View-Profile-and-List-All-Users>

CodingNepal (2020) Fully Responsive Footer Section using HTML CSS CodingNepal, 9th of April 2020. Available at:

https://www.youtube.com/watch?v=f4J3uaNNYTE

# Appendix

## Project Proposal

1. Objectives

The objectives for this project are co-ordinating and delivering a Web Application which will display many functionalities

First a concept of a project must be created to follow. This will be a Web Application for Contractors and Contracts for Contractors. A two sided app.

On one side there will be the contractors. They can join the Web App and then they can build up their profile and add a portfolio of their work and other credentials. They also can put up their prices per hour/day/job. Here the people who avail of their services can leave a review. The purpose of this will lead to a more customer focused outcome in regards the skills offered. It will be open to anyone who is a contractor in any domain.

On the other side of the Web Application there will be a page where people looking for jobs to be done can put up a posting along with some details for the contractors to peruse and contact them if the offer seems good enough for them. This is the noticeboard.

For this project there are the following objectives:

* This will be created through a server side programming language to maintain a complex persistent data storage for the Web Applications functionality.
* There will also be a client side programming language to present a GUI for the Web App.
* As project continues, detailed academic documentation will be created with correct structure and precise formatting.
* Project Management skills will be implemented to complete the project in a fair and timely manner.
* Coding skills, recently learned will be put to test and implemented to build this Web Application to a high level with a high quality user experience.
* As the project proceeds there will be regular testing and documentation of the test plans and the results.
* Finally once the project is complete there will be a video recording, a report, code submittal and a presentation to the lecturer.
* The code, report and cover sheet ready to submit for the 21st of April 2021. I will present the video and the project presentation submission on April 21st by midnight.

The project will be presented at a time specified thereafter.

1. Background

This project will be based around a Web Application in which any type of contractor can put up their portfolio and contact details on the site. Here they can be contacted by anyone who wishes to avail of their services.

This site will also allow the lay people to put up contracts in the hope of a contractor coming to contact them and offering them a good deal. Here the customer can ask them a question and wait for a response.

The reason for this idea is because there is a gap in the market here.

There are a few sites like this one already on the internet such as Fiverr, Up-Work and Tradesmen.ie, however they are more focused on specific areas of work. These sites which are either specific to certain domains, such as IT or Trades, or have a poor user friendly interface. This Website will focus on general contracting for anyone who wishes to create a profile and advertise their services. From PTs to Architects.

The users of this Web App will have the opportunity to put a lot of their info up for the potential customer to see. Their profile.

This site will be attractive too, so that it is a pleasure to look at as well as easy to use.

The sites mentioned earlier; UpWork and Fiverr are for a more international audience. This site will focus exclusively on the island of Ireland. This focus on this market will be a key element of the Website. This way it can serve its customers more directly and focus on the needs of a familiar jurisdiction. Irish people always want to buy home brands so this will provide them with another opportunity here.

Another different aspect of this site will take is that the contractor’s details as well as the contracts will be available to anyone who looks for free, they will not be required to become a user. This will be a fast one stop shop for our customers.

Finally this will allow the people of Ireland to have an easier time to find work and to find the kind of specialist they need through a user friendly Web App.

The mix of these functionalities is wider in spec than then the Website’s competition mentioned above. It allows both the customer and the contractor a greater level of control. Both can post and both can search.

1. Technical Approach

For this Web Application there will be a planned approach to be followed categorically.

Throughout the background research, Work Sites like Fiverr, Upwork and Tradesmen.ie keep coming up as the sites most like this idea, further research has led to the decision that use of a different mix functionalities could lead to a more user friendly site.

As the project continues web resources such as StackOverflow, Youtube and previous learning materials from the NCI will be used.

Amazon Web Services will be the platform that the Web App will be built on and the languages will be Ruby on Rails, HTML, CSS and SQL.

When they put up their profile on the site, there they can put any information on themselves that they see fit such as rate, location, previous awards or successful projects. I want to create a really user friendly interface that encourages the user to search.

This Web application will have the following requirements:

* Login for Contractors and Customers.
* There will be a nav bar for better functionality and allow the user to go to all the different sections easily.
* There will be an about us page which explains our origin and vision.
* An area for the contractor to put up his profile, there will be a few options here for them to put up more information on themselves.
* Sections in which the contractor’s profile will be stored for example electricians in electricians.
* There will be the ability for the customer to see the contractors by skill or area.
* The ability or someone on the site to send a contractor an email.
* There will be another section for a customer to put up their own ad for the contractors to find.
* The site will have a search bar for specific terms.
* Anyone can contact the admin if they have any special request or query.
* There will be social media links at the bottom.

Testing strategies will be implemented as the project progresses so that the end product is up and working in good order.

1. Special resources required

* Ebooks
* Web Sources – For research, websites and videos via Youtube mainly.
* Previous NCI Documents and education – There is a lot of material from the NCI to help complete the project.
* SQL – to help build the database.
* AWS – to develop the Application on AWS.

1. Project Plan

Gantt chart using Microsoft Project with details on implementation steps and timelines – project deliverable



Fig 46. Gantt chart of the Contractor Contactor Project Lifecycle

1. Technical Details

This project will be implemented on AWS through the languages of:

* Ruby on Rails
* HTML
* CSS
* SQL

1. Evaluation

* Going through the project each item, a page or a functionality or an object, will be tried to make sure it runs. This will be done systematically in piecemeal fashion to make sure that it works effectively. The code will be clean, indented and documented.
* Validation will be carried out to make sure the web app fulfils the requirements set and to make sure it runs.
* Manual testing will be very important going through this project, as well as record keeping for of the testing.
* Black box testing will be implemented, checking the source code and keeping the objectives in mind.
* White box testing will be done to test the structure and design, the security, paths, inputs, outputs, loops and statement and objects by using test cases, statement and branch coverage.

In the end test customers will try out the Web Application to make sure that it runs smoothly. If needs be amends will made until the project is done and ready for submittal.

## Project Plan

Table of Contents

[Version History 1](#_Toc528342615)

[1. Introduction 3](#_Toc528342616)

[1.1. Purpose of The Product Design Specification Document 3](#_Toc528342617)

[2. General Overview and Design Guidelines/Approach 3](#_Toc528342618)

[2.1. Assumptions/Constraints/Standards 3](#_Toc528342619)

[3. Architecture Design 4](#_Toc528342620)

[3.1. Logical View 4](#_Toc528342621)

[3.2. Hardware Architecture 6](#_Toc528342622)

[3.3. Software Architecture 6](#_Toc528342623)

[3.3.1. Server-side 6](#_Toc528342624)

[3.3.2. Client-side 6](#_Toc528342625)

[3.4. Security Architecture 6](#_Toc528342626)

[3.5. Communications Architecture 6](#_Toc528342627)

[3.6. Performance 7](#_Toc528342628)

[4. System Design 7](#_Toc528342629)

[4.1. Use Cases 7](#_Toc528342630)

[4.2. Database Design 7](#_Toc528342631)

[4.3. Data Conversions 7](#_Toc528342632)

[4.4. Application Program Interfaces 7](#_Toc528342633)

[4.5. User Interface Design 7](#_Toc528342634)

[4.6. Performance 11](#_Toc528342635)

[Acronyms, Abbreviations and Terms 11](#_Toc528342636)

# 1. Introduction

## 1.1. Purpose of The Product Design Specification Document

This document has been created to give the reader an overall idea of the system required to run the Contractor Contactor Web Application. This document has been created to help me come up with an organized approach to building out the Web Application’s system and architecture. It is to help a developer and the project manager to completing the goal of this project.

**2**. **General Overview and Design Guidelines/Approach**

## 2.1. Assumptions/Constraints/Standards

For this Web Application to be fully online, running and to provide the great service it intends; there will be a few requirements.

The user of the Web Application will need an internet connection. If it is offline, the user will be informed of such by their own system.

The extent of the system and how much it can offer to the user will be defined by the amount of work that can be put into it by the end of the project.

This Web Application is a straightforward system and will be available for all devices who can attain internet connection. It will be developed for the latest edition of Chrome and on a Windows 10 PC.

This application is intended to be available for any device and screen size and will be highly user friendly.

# 3. Architecture Design

## 3.1. Logical View

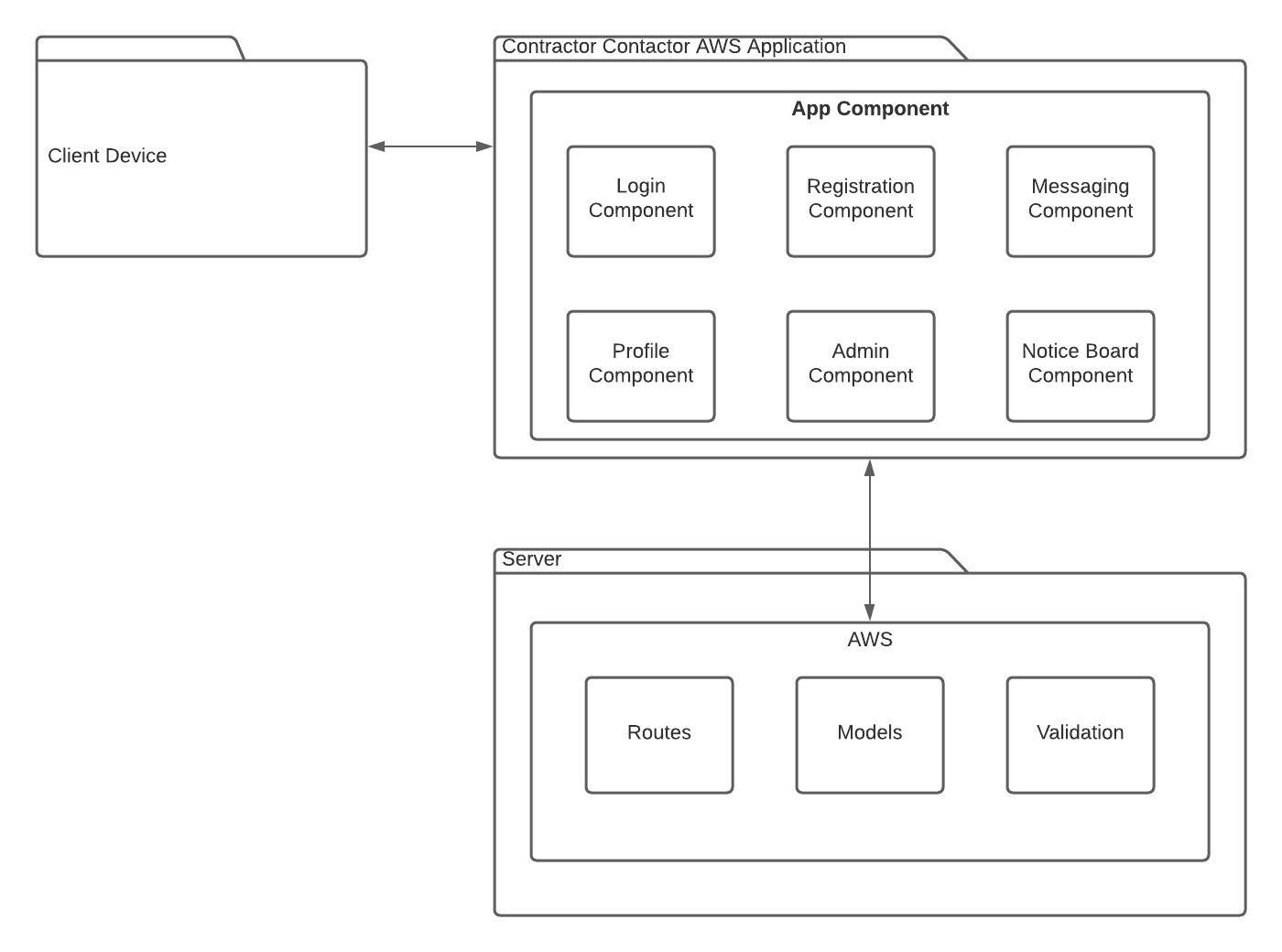


Fig 47. The picture above displays the overall system of the Contractor Contactor.

In this representation we start with the user’s browser. They access this on the device that they wish to do so on.

They will get to our application via a URL request. This is a website it will be programmed through ruby, HTML, CSS and JavaScript on AWS. It will be a single page website which functions as a multipage. It will be very quick and very user friendly. As indicated there are several pages which the user can navigate through such as the login and the Profile. The flexibility of the website will be handled by routes and the information will be stored in the models. There will be some validation required such as validation from the Admin of new profiles.

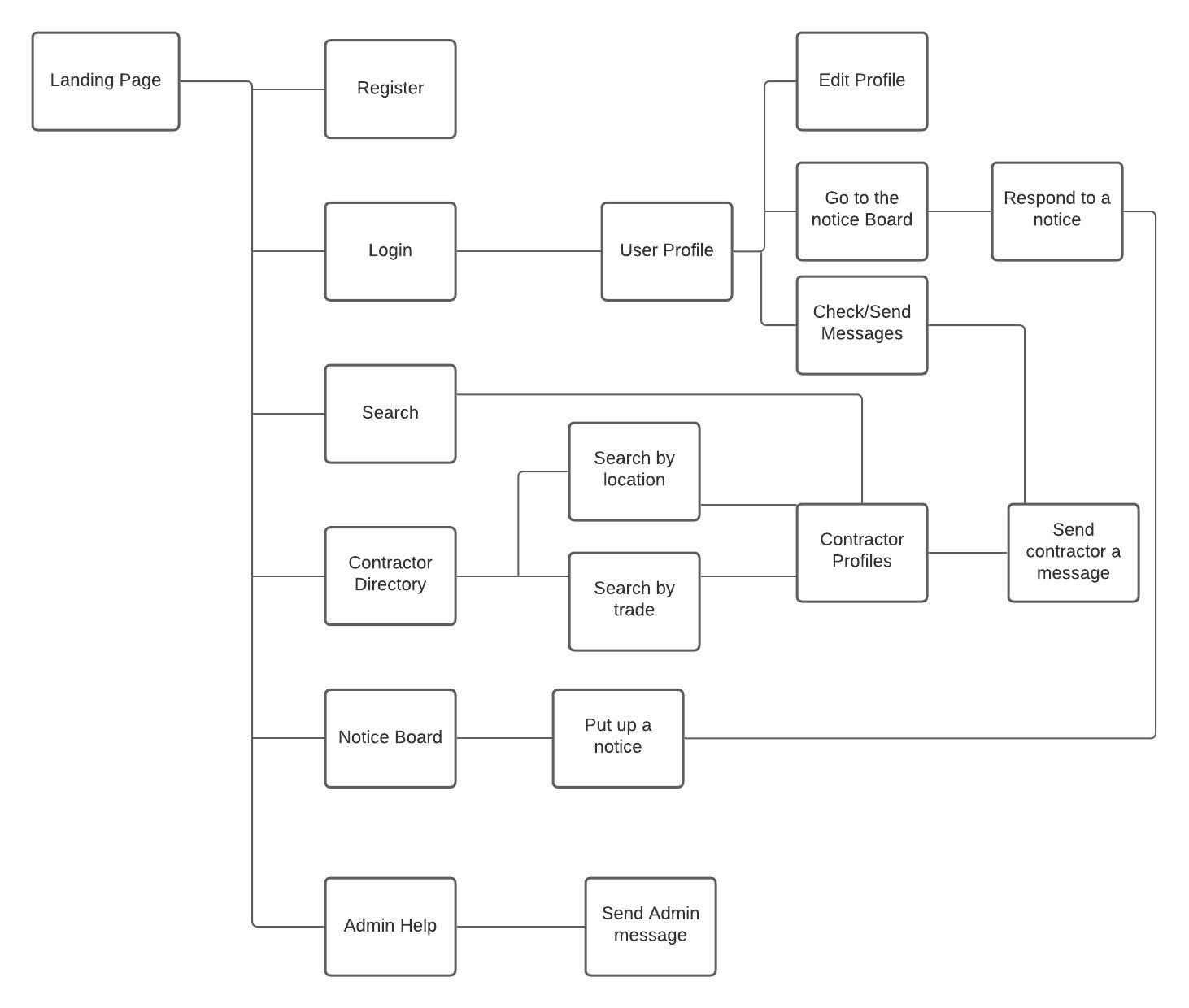


Fig 48. The site map.

This is the site map. The user starts on the landing page where there are several options. The customer can register an account, login into the account, search for a type of contractor, enter the contractor directory, view the notice board and enter the admin help section.

After log in, there are a few further choices. There are still the previous user functionalities but if the user has a profile option then they can edit their profile.

There is the notice board where the user can put up a notice and respond to notices.

The search bar will bring the user into the contractor profiles.

There is the contractor directory where you have a directory of Contractor Profiles.

Finally there is the help section. Here the user can see specify sections like the FAQs etc. and most importantly the user can communicate directly with the admin for better customer support on the app.

**3.2. Hardware Architecture**

**This Website will be set up on AWS.**

**3.3. Software Architecture**

Contractor Contactor Web Application Software Architecture will use the following software:

## 3.3.1. Server-side

HTML

CSS

JavaScript

## 3.3.2. Client-side

Ruby on Rails

# 3.5. Communications Architecture

# This Web Application will make use of HTML CSS and some JavaScript on the front end. The user will interact with the front end when using the application. This front end will sent the data to the back end.

# Data will be required, such as login information to access entry to the Web Application, profile information for the contractors to be able to use the site as a contractor with all the benefits that entails.

# The tables in the database in AWS will be planned on the SQL workbench and then scaffolded into the AWS system. Here all the data will be stored for the website.

## 3.6. Performance

AWS and Heroku are very well made systems which provide excellent and efficient service to many users at once. We can expect speed, flexibility and no problems with the functionality with these reputable systems.

# 4. System Design

## 4.1. Use Cases

Use cases for the Contractor Contactor application can be found in the Requirements Specification documentation.

## 4.2. Database Design

At first I will use the SQL workbench to design my database as I am very familiar with this too. I will create all my tables, primary keys, foreign keys and relationships.

After this I will test it out several times to make sure that it runs correctly.

When this is working to an optimal level and I am satisfied that it will serve the Web Application to a high impeccable level I will reassess it for the Ruby on Rails code. After this the code will be scaffolded into the AWS system so that the database can be accessible for the Web Application.

## 4.3. Data Conversions

Data will be moved around the system at high speed. It will go from the front end to the back end to the database, where it is stored for later use. If it is requested it will be sent to the client for review by the user of the Web Application.

The data will be stored in AWS on the tables we have scaffolded in.

## 4.4. Application Program Interfaces

The API is the way for the user to communicate with the Web Application. The user sends a HTTP request to the Web Application, for example to get access to the Contractor Profiles. The Contractor Profile section will receive the request and respond with the correct type of contractor profile.

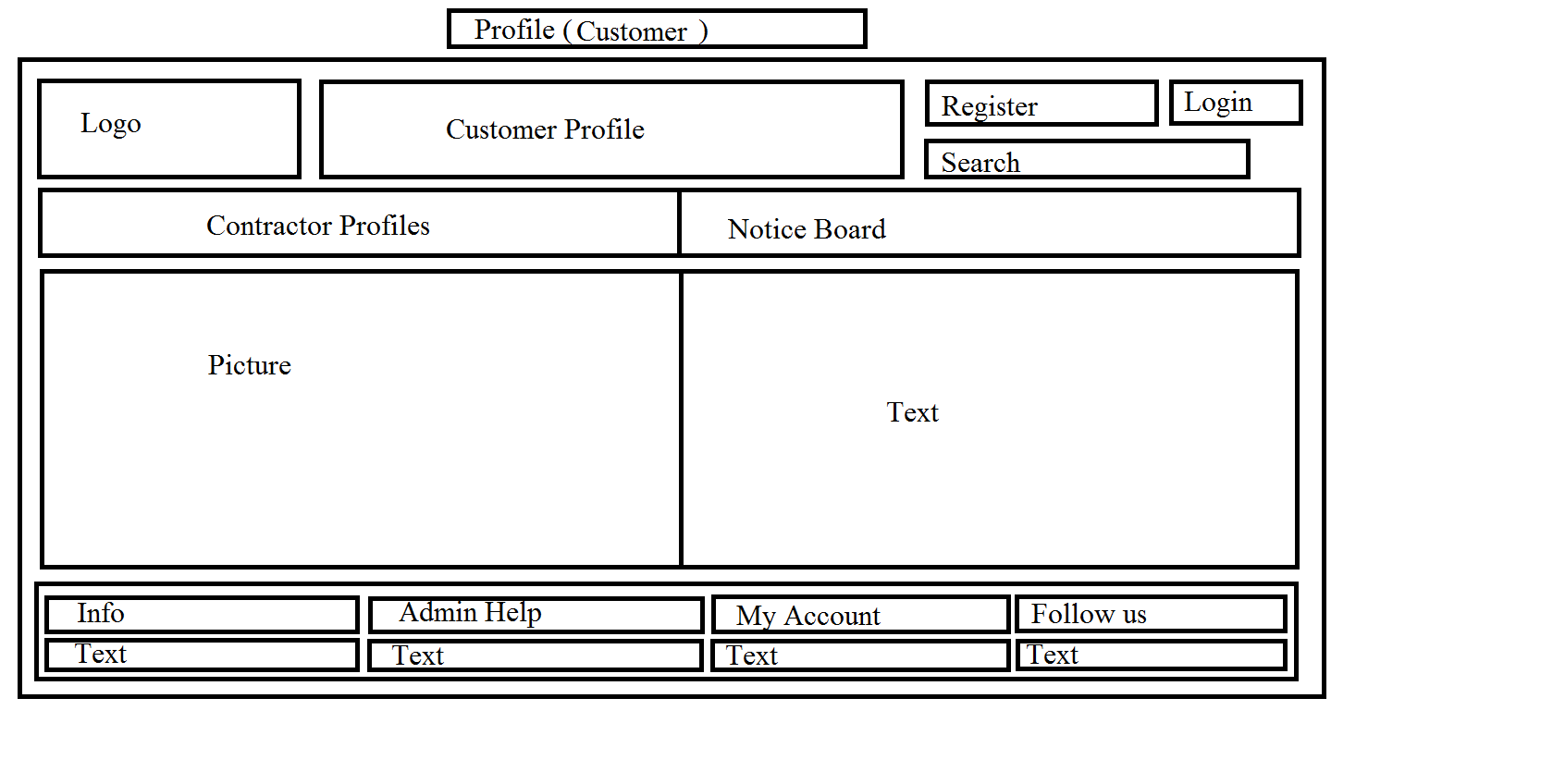
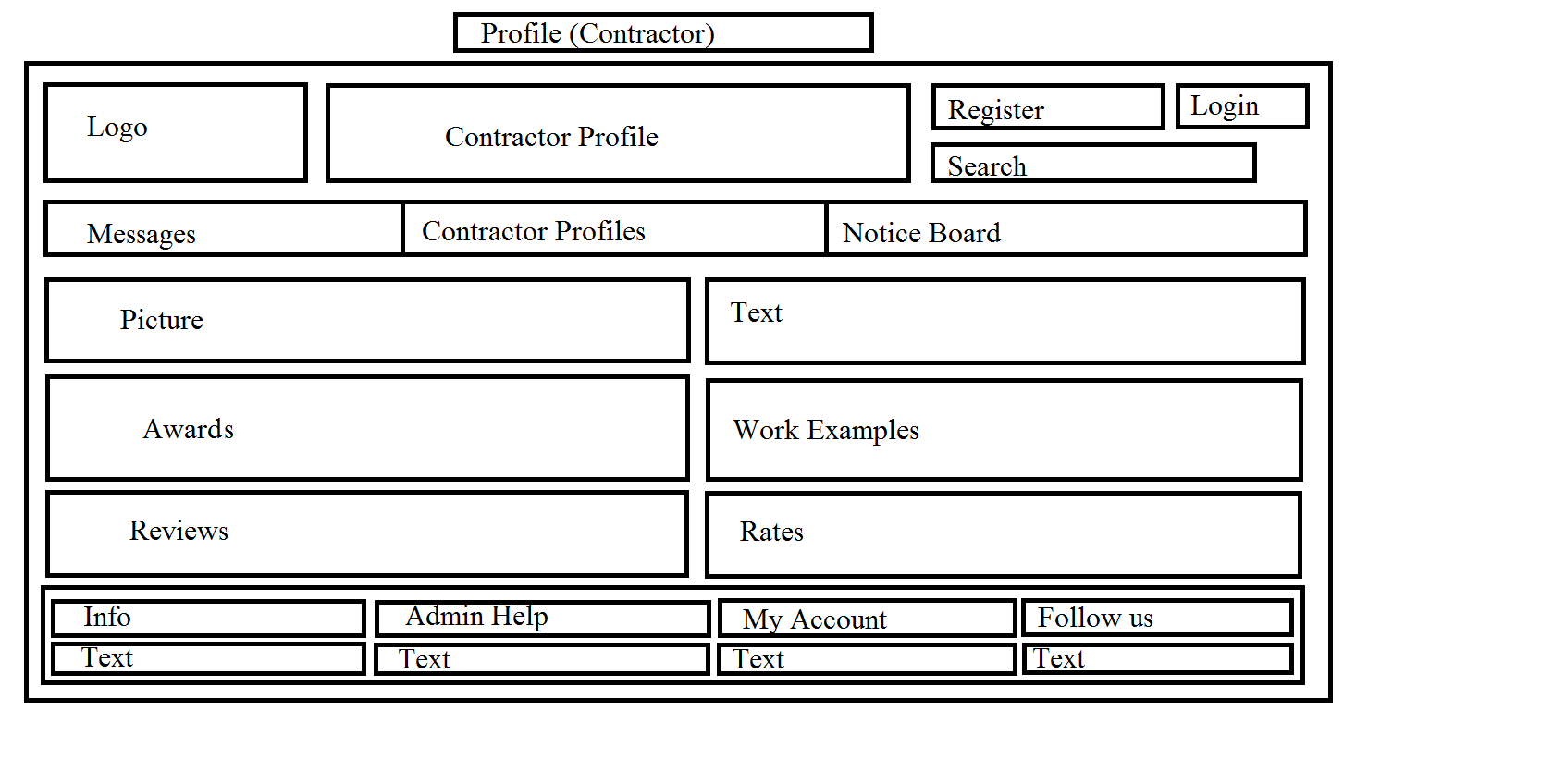
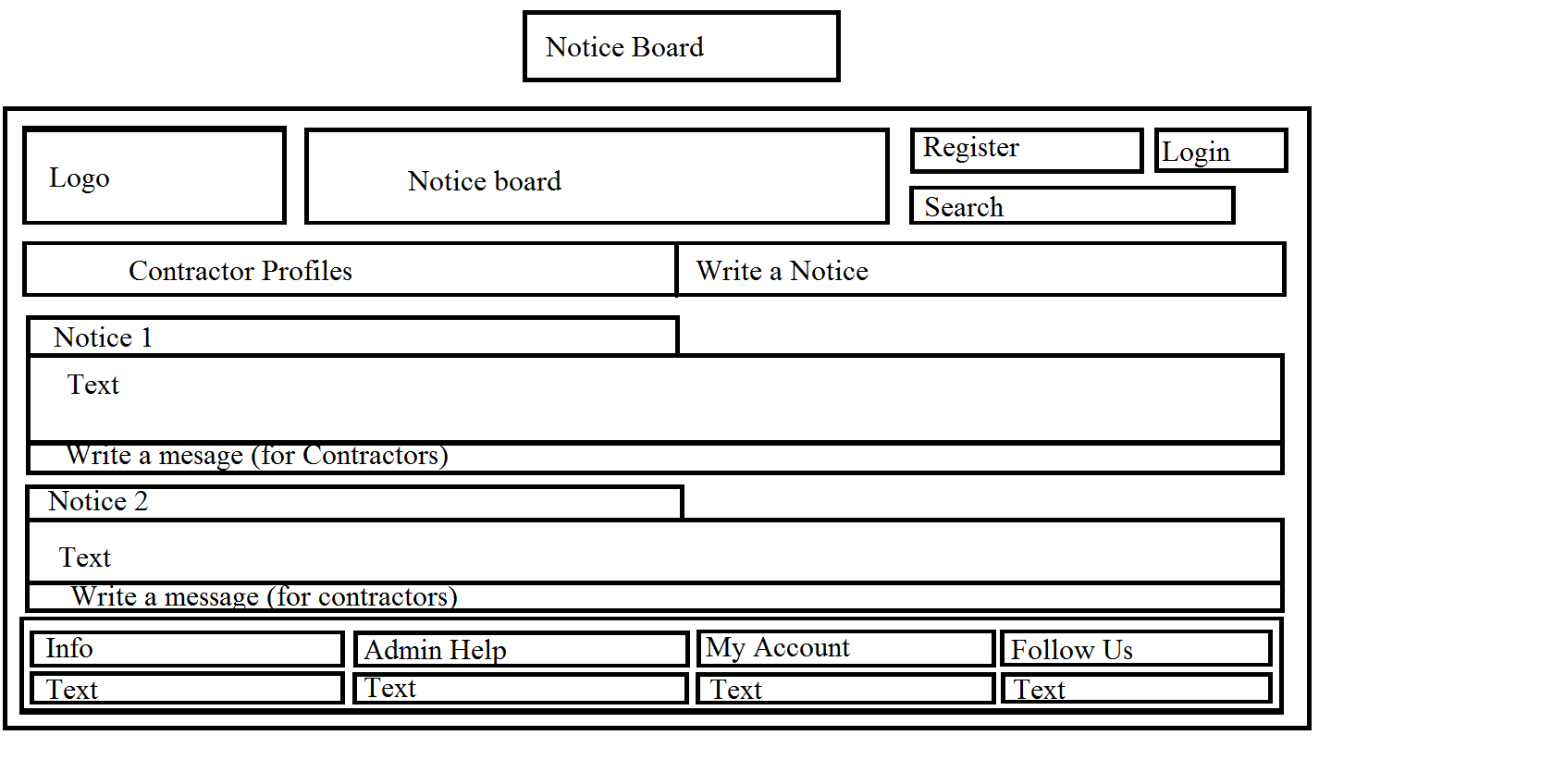
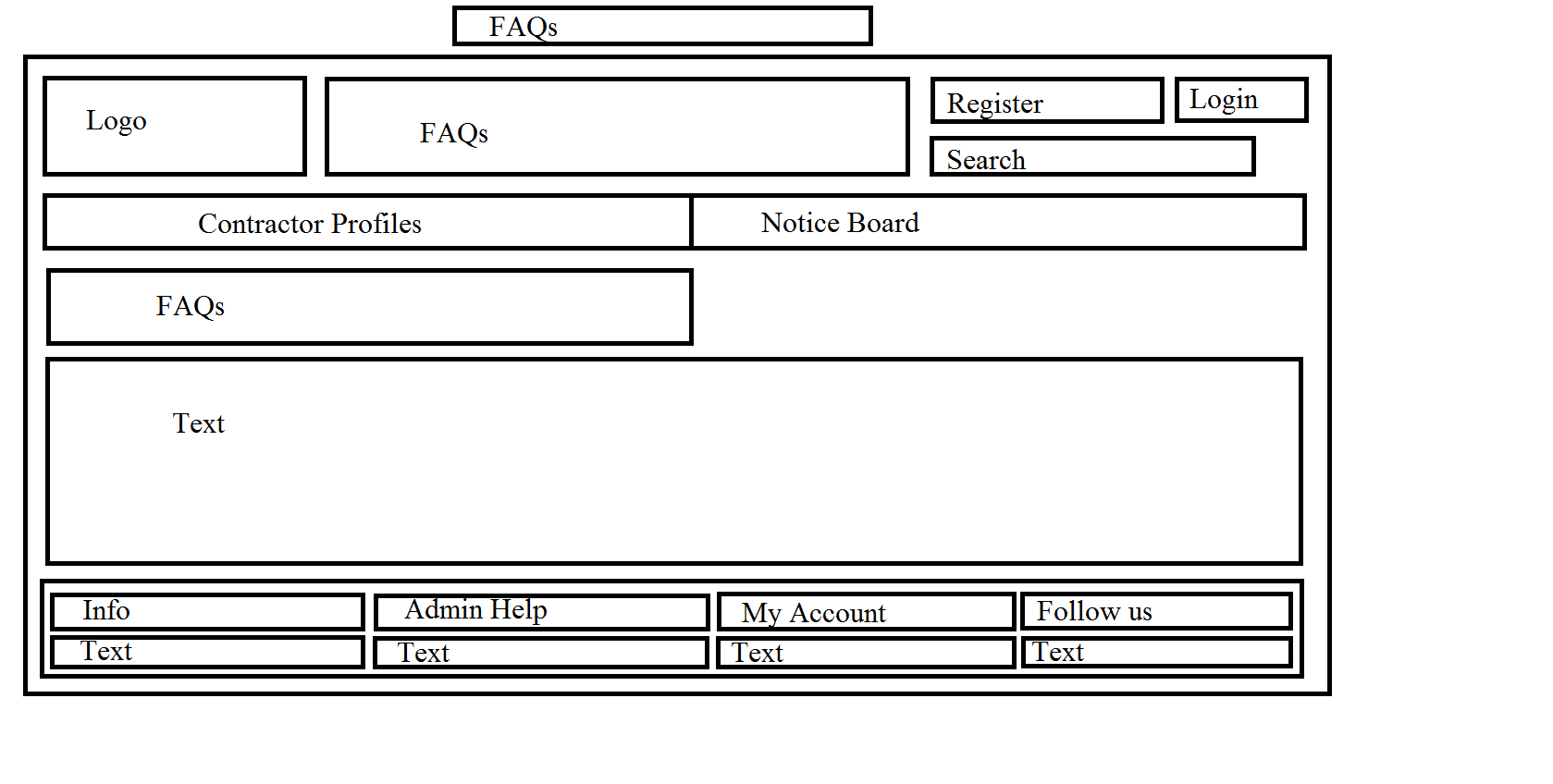
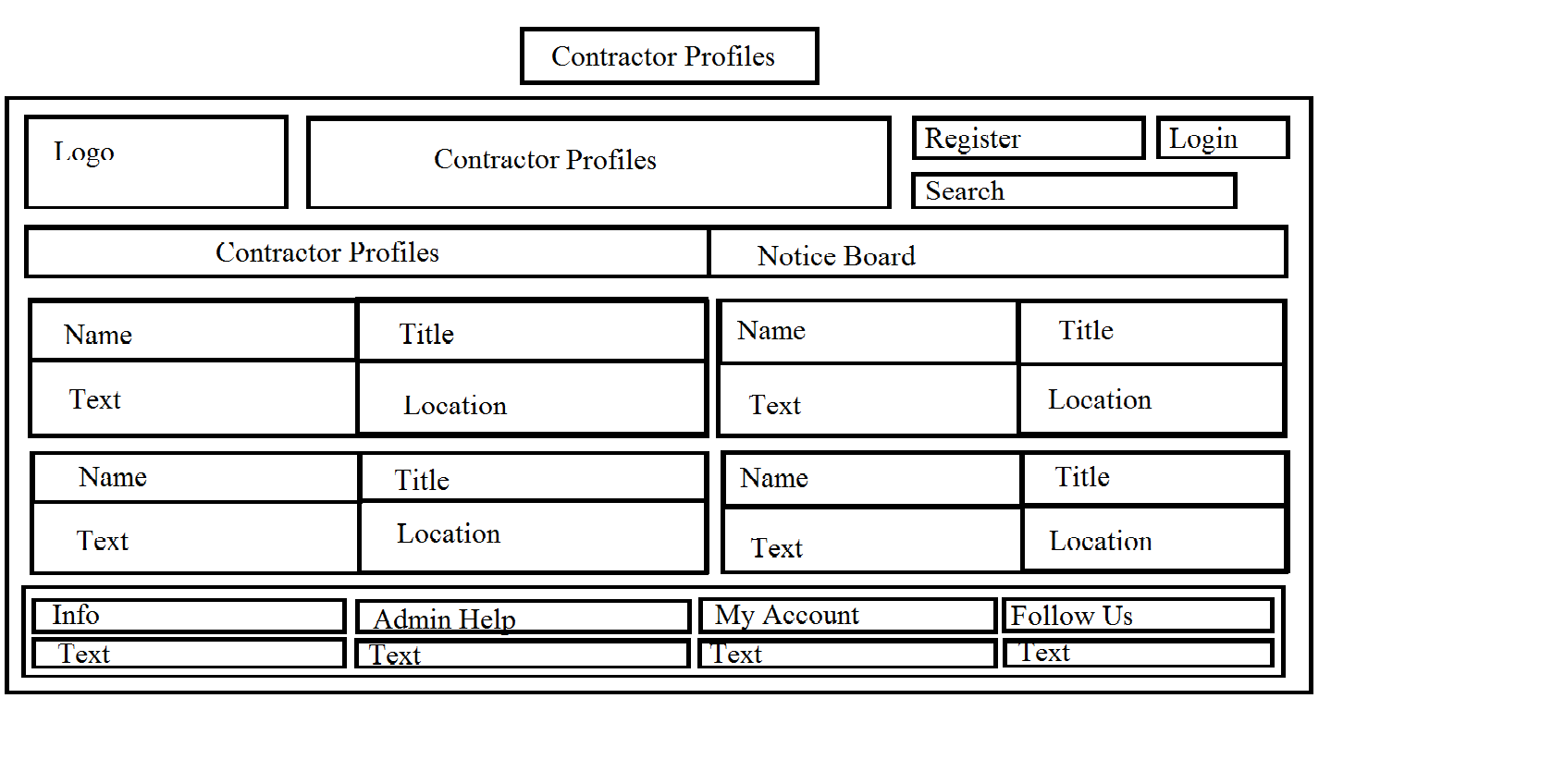
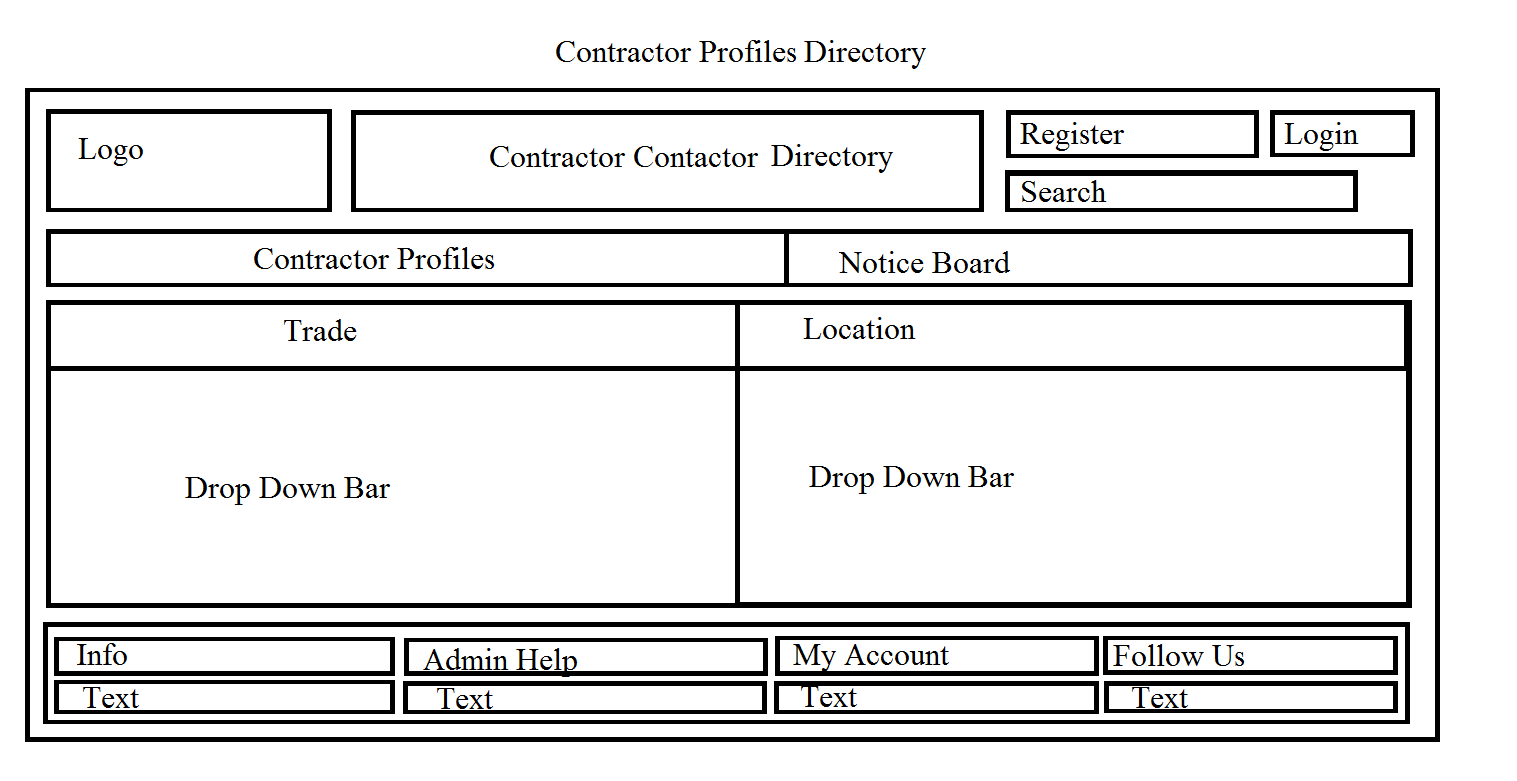
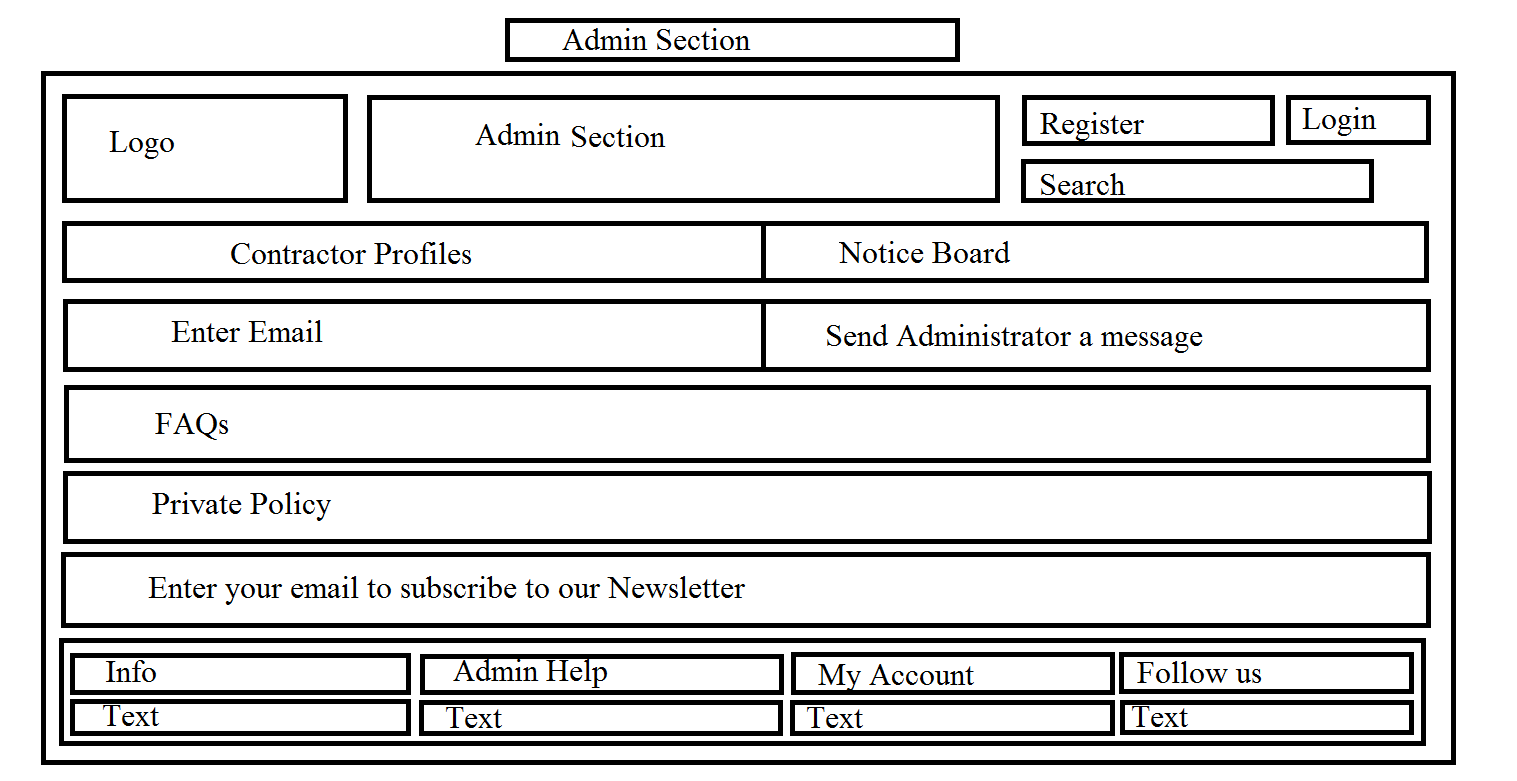
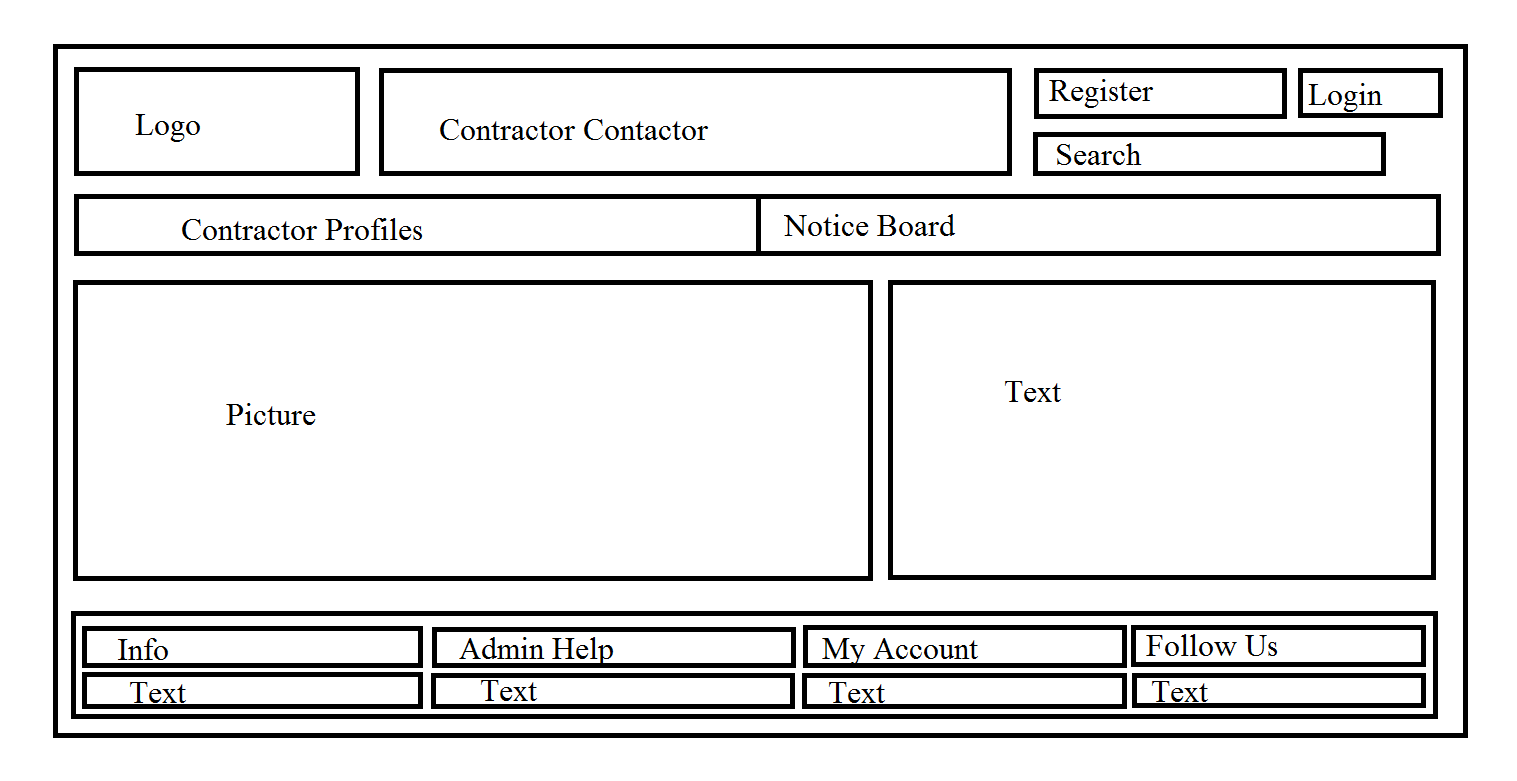
## 4.5. User Interface Design

The User Interface Design will be created with 3 technologies. On the front end we will make a very user friendly interface using HTML, CSS and JavaScript. These technologies shall be written into AWS.

All of these technologies are the core languages for any Web Application. They will create a well-structured website with an enticing aesthetic feel and a high level of responsiveness. All of this will create a unique and enjoyable experience for the user which will encourage them to return.

Here are some wireframe examples of what the user interface will look like:

This is the main page.



## 4.6. Performance

If someone accesses our page they will find it to be very user friendly, aesthetically pleasing as it will work very fast. This is due to the high level of code, the particular coding languages, Heroku and the simplicity of the Website.

The only thing that may hold back this website would be the internet connection of the user, whether Heroku or the Contractor Contactor We Application is undergoing updates.

**13. Acronyms, Abbreviations and Terms**

|  |  |
| --- | --- |
| HTML | Hyper-text Mark-up Language, the language that provides structure to web pages. |
| CSS | Cascading Style Sheets, adds styling to web pages. |
|  |  |

## Requirement Specification

**Table of Contents**

[1 Introduction 4](#_Toc316977392)

[1.1 Purpose 4](#_Toc316977393)

[1.2 Project Scope 4](#_Toc316977394)

[1.3 Definitions, Acronyms, and Abbreviations 6](#_Toc316977395)

[2 User Requirements Definition 6](#_Toc316977396)

[3 Requirements Specification 7](#_Toc316977397)

[3.1 Functional requirements 7](#_Toc316977398)

[3.1.1 Use Case Diagram 9](#_Toc316977399)

[3.1.2 Requirement 1 <Log In> 10](#_Toc316977400)

[3.1.3 Requirement 2 < Enter Profile > 11](#_Toc316977401)

3.1.4 Requirement 3 < Enter Notice Board > 13

3.1.5 Requirement 4 < View Contractor Profile > 15

3.1.6 Requirement 5 < Administrator Help > 16

[3.2 Non-Functional Requirements 18](#_Toc316977402)

[3.2.1 Performance/Response time requirement 18](#_Toc316977403)

[3.2.2 Availability requirement 18](#_Toc316977404)

[3.2.3 Recover requirement 18](#_Toc316977405)

[3.2.4 Robustness requirement 18](#_Toc316977406)

[3.2.5 Security requirement 18](#_Toc316977407)

[3.2.6 Reliability requirement 19](#_Toc316977408)

[3.2.7 Maintainability requirement 19](#_Toc316977409)

[3.2.8 Portability requirement 19](#_Toc316977410)

[3.2.9 Extendibility requirement 19](#_Toc316977411)

[3.2.10 Reusability requirement 19](#_Toc316977412)

[4 GUI 20](#_Toc316977414)

[5 System Architecture 25](#_Toc316977415)

[6 System evolution 26](#_Toc316977416)

**1. Introduction**

This purpose of this introduction is to outline to the reader the requirements specification of the Contractor Contactor web application. It will provide a thorough analysis of the steps involved in the creation of this web application from conception to the end result.

## Purpose

The purpose of this Requirements Specification document is to elucidate the requirements for the Contractor Contactor Web Application. This application is for contractors and the general population. The Contractors will have a profile where they can be contacted and the general public will put up contracts for the contractors seek out. The user must be a contractor to see these contracts. This will be for all types of contractors. A general directory.

The target audience will be the contractors of Ireland and the people who seek them out. It will be free and easy for people to seek out the contractors.

## Project Scope

The project scope here is to create a Web Application for Contractors to have a profile on, which they can display their portfolio and be contacted by the general public, initially by a message and then they can choose to respond or not. The other side of the Web Application will be a page where we will have a message board where the general public can put up contracts and here the contractors can send a message to the person who put it up. This Web Application will have a focus on Ireland and it’s localities to facilitate the different types of contractors. It allows the people in Ireland to support the home market.

This Web application will be developed with HTML, CSS and Ruby on AWS.

**Motivation for this project**

* The main motivation for this project is that there is a gap in the market here. There are sites for specific contractors and there are international contractor sites. There is no web application which is user friendly, for contractors of all types and for the general public to put up their own contracts with a focus on the domestic Irish market. Irish people like to support home brands especially if there is a focus on their locality.
* Another major motivation is to create a major directory of contractors. This will boost competition and therefore quality of service, it will also allow juniors to compete with seniors by offering lower rates, if that is what they wish to do.

**Project objectives**

* The primary objective for this project is to develop a user friendly Web Application for a general directory of Contractors and for the general population to put up contracts to be viewed by contractors.
* The Website will be fully functional.

**Project expectations**

* This Web Application will be available online and will be functional.
* After this, there will be continued work on the Application and marketing for it to gain an online presence.

**Project suppositions and restrictions**

* There will have to be significant work on the marketing to get a presence on the web application.
* When this project is submitted there will be further work to create a mobile application for this Web Application.
* The will be a major focus on data and personal protection to keep all parties safe while using the Web application.
* Initially the focus will be on one county such as Kerry.

**Project Risks**

* There is a risk that the security features will not be perfect upon completion.
* There may be an issue with time, in finishing the project on time.

## Definitions, Acronyms, and Abbreviations

Contractor: Contractor, someone who wants to put up a contract on the Web Application.

Customer: The general public.

Administrator: Person who runs the Web Application.

User: The Contractor or the Customer

# User Requirements Definition

* The main user requirements are that this site be available online as a web application.
* Contractors will have to create an account and a profile to receive inquiries from the public, access the contracts from the public and to display their portfolio.
* The general public will not have to create a profile to use this site. They will have free access to view the profiles, make inquiries, and leave reviews and to put up contracts for the Contractors to see them.

# Requirements Specification

All requirements will be fully functional. This will be a very user friendly Web Application. It will be easy for anyone to use. The contractor only needs to create an account and a profile to be up and running. The customer doesn’t need to create a profile but will have the option. They can freely use the site. We will have an FAQ section and the ability to contact the admin if there is any issue.

## Functional requirements

This part of the Project Requirements will state all of the functional requirements that the contractor.

**The contractor**

* When a contractor comes onto the Website they will have the option to log in or sign up.
* If they log in, the contractor can go to their profile. Here they can add a picture of themselves, their awards, their work, reviews and rates if they wish. They can also add the areas that they will go to for work.
* The contractor can then check a messages section. These messages are inquiries from the general public. The contractor can respond here.
* The contractor will have the ability to check the notice board for contracts put up by the general public. The contractor will be able to write a response here to the person who put it up.
* The contractor will have the ability to put up contracts here too.

**The general public**

* They will have the option to sign up and log in.
* If they have a profile they can put up a picture, text and their reviews.
* They can view the contractors in a certain area or by trade.
* They can then go into the contractors profile and view their portfolio.
* They can send the contractor a message via the internal messaging system.
* The general public can then put up their own contract on a noticeboard and wait for a contractor to come and put a message under the post, then they can put a message under the post.

### Use Case Diagram

Fig1: Use Case Diagram of the System “Contractor Contactor”

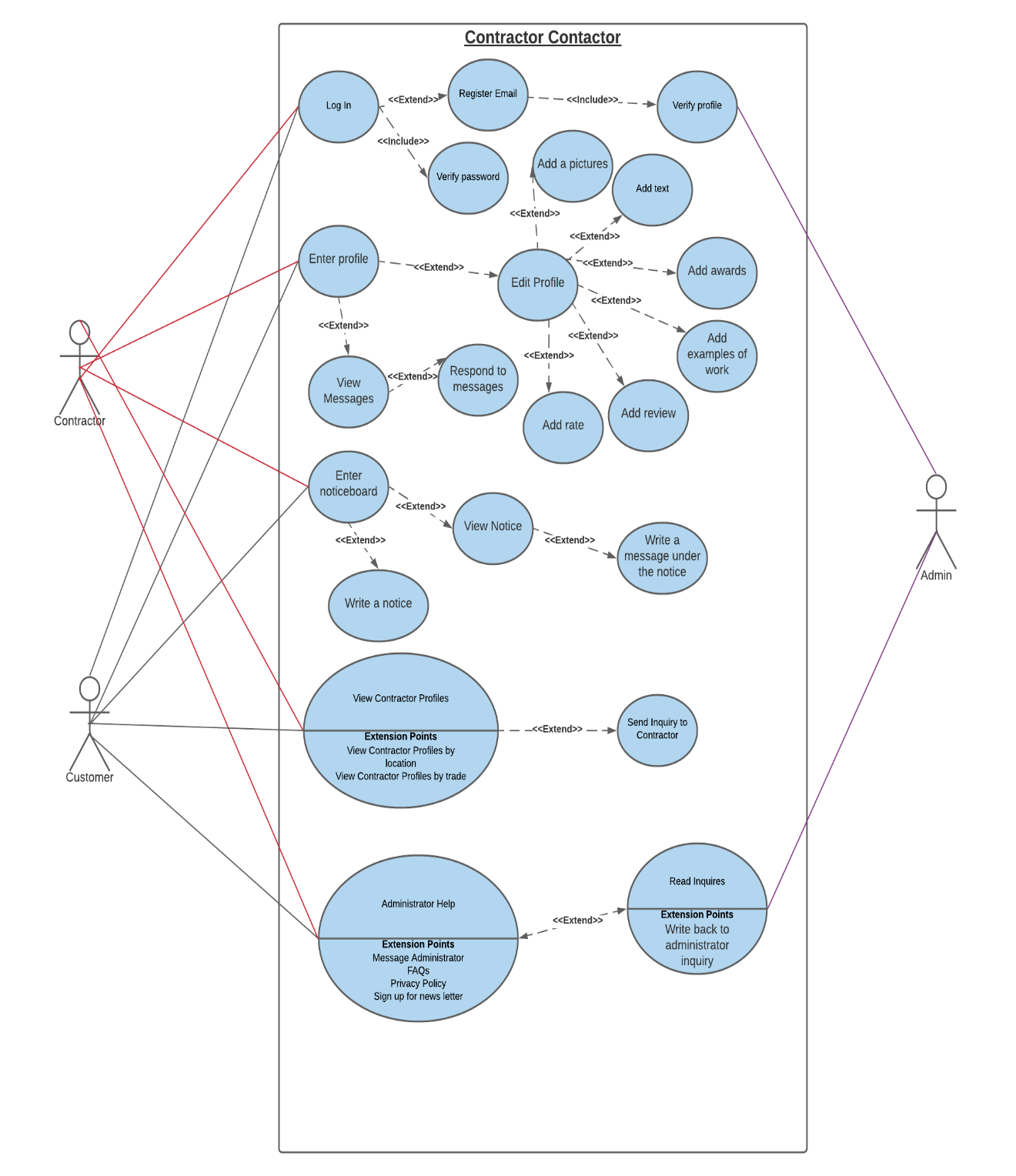


Fig 51. The class diagram.

### Requirement 1 <Log In>

#### Description & Priority

Contractor will require a profile to use the site and put up contracts. They will need a user name and a password. The Contractor will have to be verified by the admin.

Customers do not need to sign in to use the site but will have the option to do so.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the log in/register functionality works.

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when a contractor or a customer logs in to the Web Application or if they have no account signs up and then logs in.

**Main flow**

1. The customer or the contractor enters the URL into their search engine.
2. Either user enters into the Web Application.
3. The customer or the contractor will enter a username and a password to log in.

**Alternate flow**

A1: Register a new Account

1. The customer or the contractor will enter an email address and a password to register.
2. The Administrator will accept the user if they deem the user fit.
3. The use case continues at position 1 of the main flow

**Exceptional flow**

E1: Incorrect User Sign In Details

1. The user enters the username and password
2. The system tells the user that the details are incorrect.
3. The system asks the user to re-enter the details

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition.

### Requirement 2 < Enter Profile >

#### Description & Priority

This is when the user enters their profile on the Web Application. Here they will have the option to edit their account.

The contractor will have the ability to view the messages sent to him.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the Profile works on the Web Application.

**Flow Description**

**Precondition**

The user is logged in.

**Activation**

This use case starts when the user clicks on “Enter Profile”.

**Main flow**

1. The user logs in.
2. The User clicks on “Enter Profile”

**Exceptional flow**

E1:

1. On Entering the Profile the contractor can edit their profile.
2. The contractor can edit the profile picture.
3. The contractor can edit the profile text.
4. The contractor can edit the profile awards.
5. The contractor can edit the profile examples of work.
6. The contractor can edit the profile reviews.
7. The contractor can edit the profile rate for work done.

E2:

1. The Contractor can view the inquiries from the customers.
2. The contractor can respond to these messages.

E3:

1. The Customer can edit their profile picture.
2. The Customer can edit their profile text.

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition

### Requirement 3 < Enter Notice Board >

#### Description & Priority

This functionality allows the user of the Web Application to enter the Notice Board where they can see the contracts. Only the contractors can write responses on the Web Application.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the Notice Board section works.

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when a contractor or a customer clicks on the notice board section

**Main flow**

1. A user enters the web application
2. A user clicks on the notice board section.
3. They enter into this section.
4. The user clicks on a locality.
5. The user can scroll up and down and view the contracts availably in this area of Ireland.

**Exceptional flow**

E1:

1. A customer can write a notice
2. A Contractor can write a notice

E2:

1. A contractor will be able to write a message under the contract notice. You must be a registered contractor to write responses. This will generally be contact details or profile information.

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition

### Requirement 4 < View Contractor Profile >

#### Description & Priority

This is the ability to see all the contractor profiles by location or trade or both.

#### Use Case

**Scope and Description**

The scope of this use case is to show how the “View Contractor Profiles Work”

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when

**Main flow**

1. Enter the contractor profile section.
2. View Contractor by trade.
3. View Contractor by location.
4. View contractor by trade or location.
5. Click on the contractor profile.
6. Enter into the profile.

**Exceptional flow**

E1:

1. Enter your email.
2. Send the contractor a message

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition

### Requirement 5 < Administrator Help >

#### Description & Priority

Here the user will have the ability to send the administrator a message. View the FAQs, see the privacy policy or sign up for the news latter

#### Use Case

**Scope and Description**

The scope of this use case is to show how the Administrator Help section works.

**Flow Description**

**Precondition**

The system is not in use.

**Activation**

This use case starts when a contractor or a customer clicks on the Administrator Help section

**Main flow**

1. The user enters the Administrator Help section.

**Exceptional flow**

E1:

1. The user can send the administrator a message, they must enter their email here.

E2:

1. The user can see the FAQs.

E3:

1. The use can see the privacy policy.

E4:

The user can enter their email to get a newsletter from the Administrator.

E5:

The Administrator can send an email back to the user.

**Termination**

The system stops all activity when the user signs out.

**Post condition**

The system goes back into an idle condition

## Non-Functional Requirements

The non-functional requirements are also required for the system to run effectively.

### Performance/Response time requirement

This will depend on the speed of the internet connection.

### Availability requirement

This shall be up and running for the 25th of April 2021.

### Recover requirement

The Web Application will have a backup system set up. This will back up the data files daily. This will keep the data safe and the website in good working order.

### Robustness requirement

The Web Application must be able to run despite errors and malfunctions in the code during the running of the program.

### Security requirement

All Contractors must create accounts and these profiles will be verified and made accountable by the administrator.

The customer can create an account too, this will be verified by the administrator too.

In both cases they must use the correct details to access the accounts.

Neither user’s info shall be available to anyone except the administrator.

### Reliability requirement

When the Web Application is available online it will run at full capacity for the user.

### Maintainability requirement

The code shall be clean, indented and very legible. This is for future updates and amendments.

### Portability requirement

This Web Application will be available on smartphone, laptop and PC.

### Extendibility requirement

Later down the road I may create another app exactly like this for the UK Market.

### Reusability requirement

This web application will be reusable for all different types of Contractors by trade and location. It will always be able to add new users and new data to the tables.

# GUI

Basic model

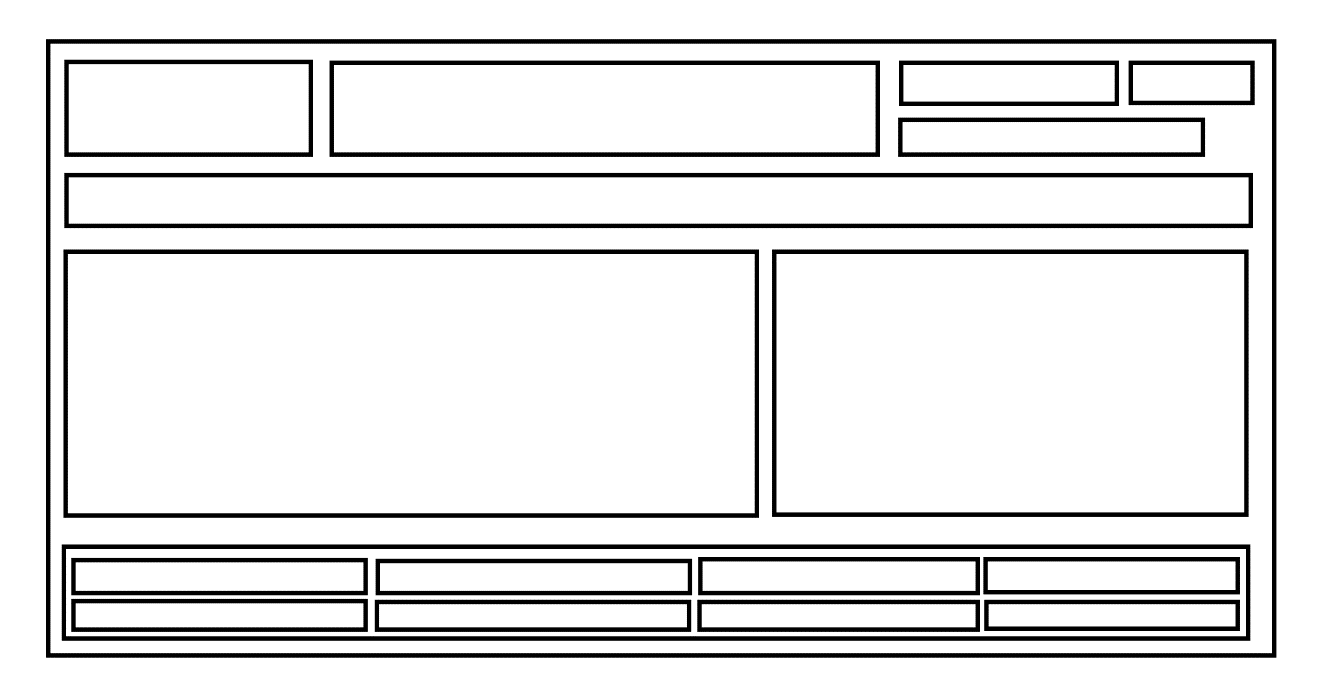


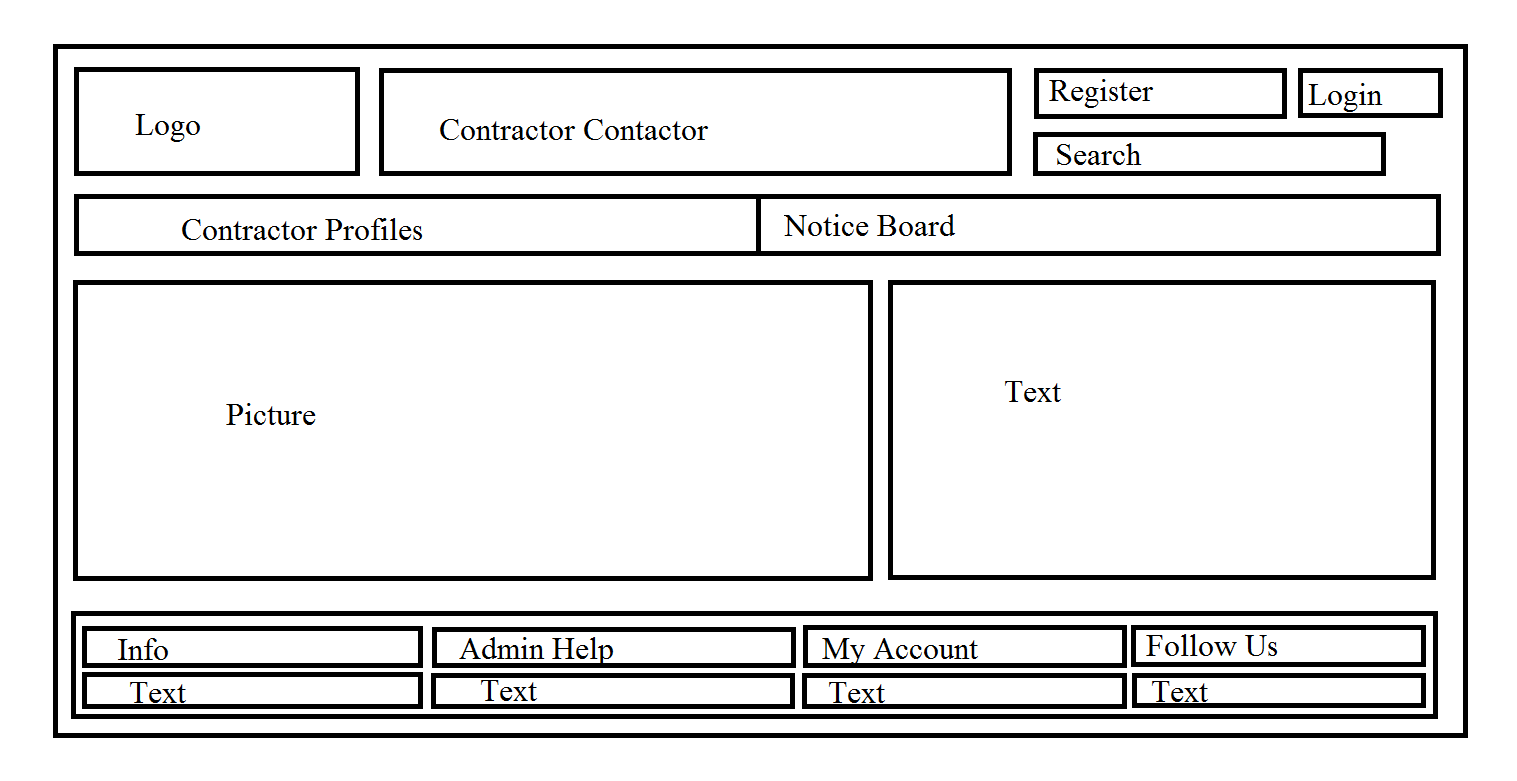
Fig 52. Main Page

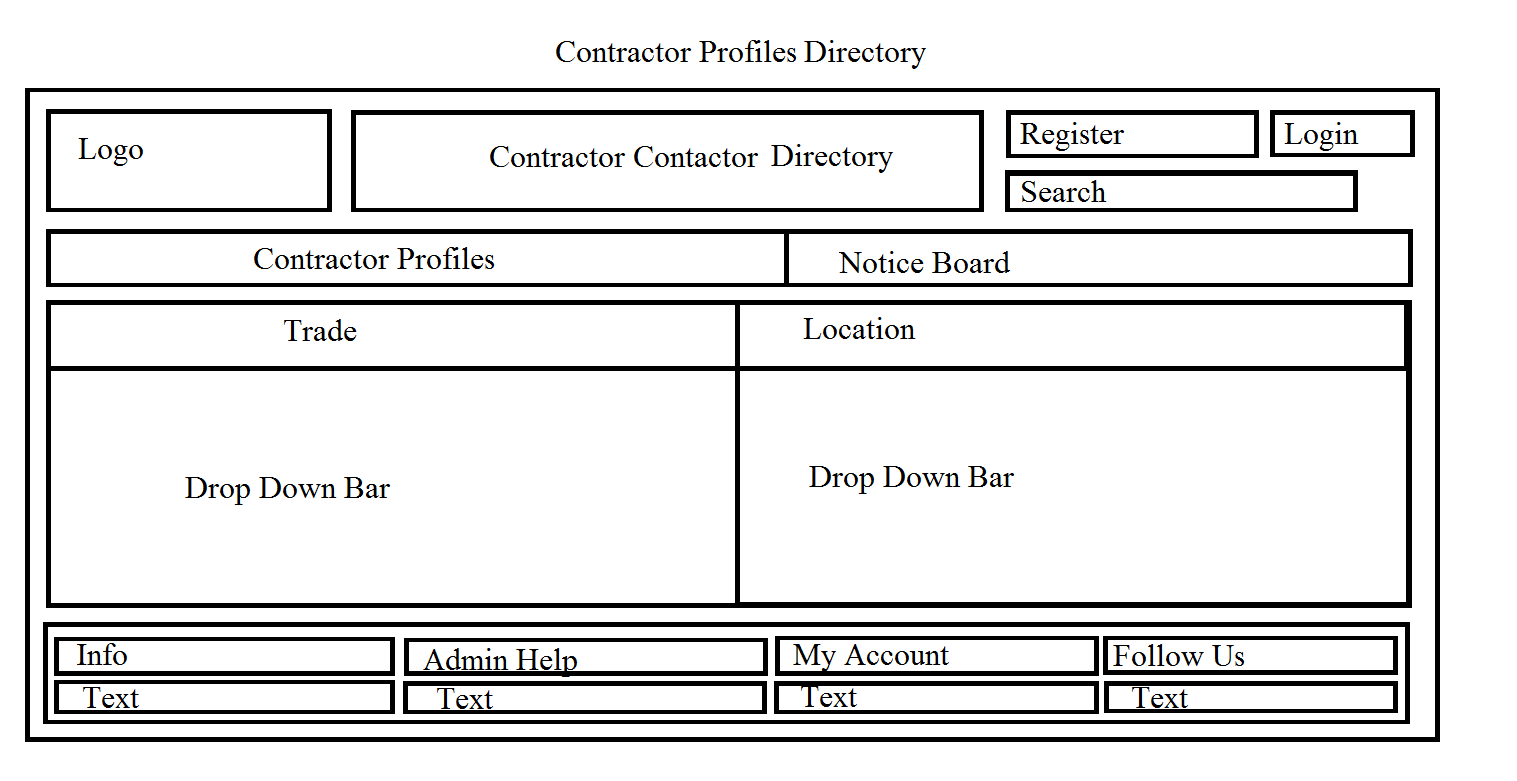
Fig 52 The contractor Directory.

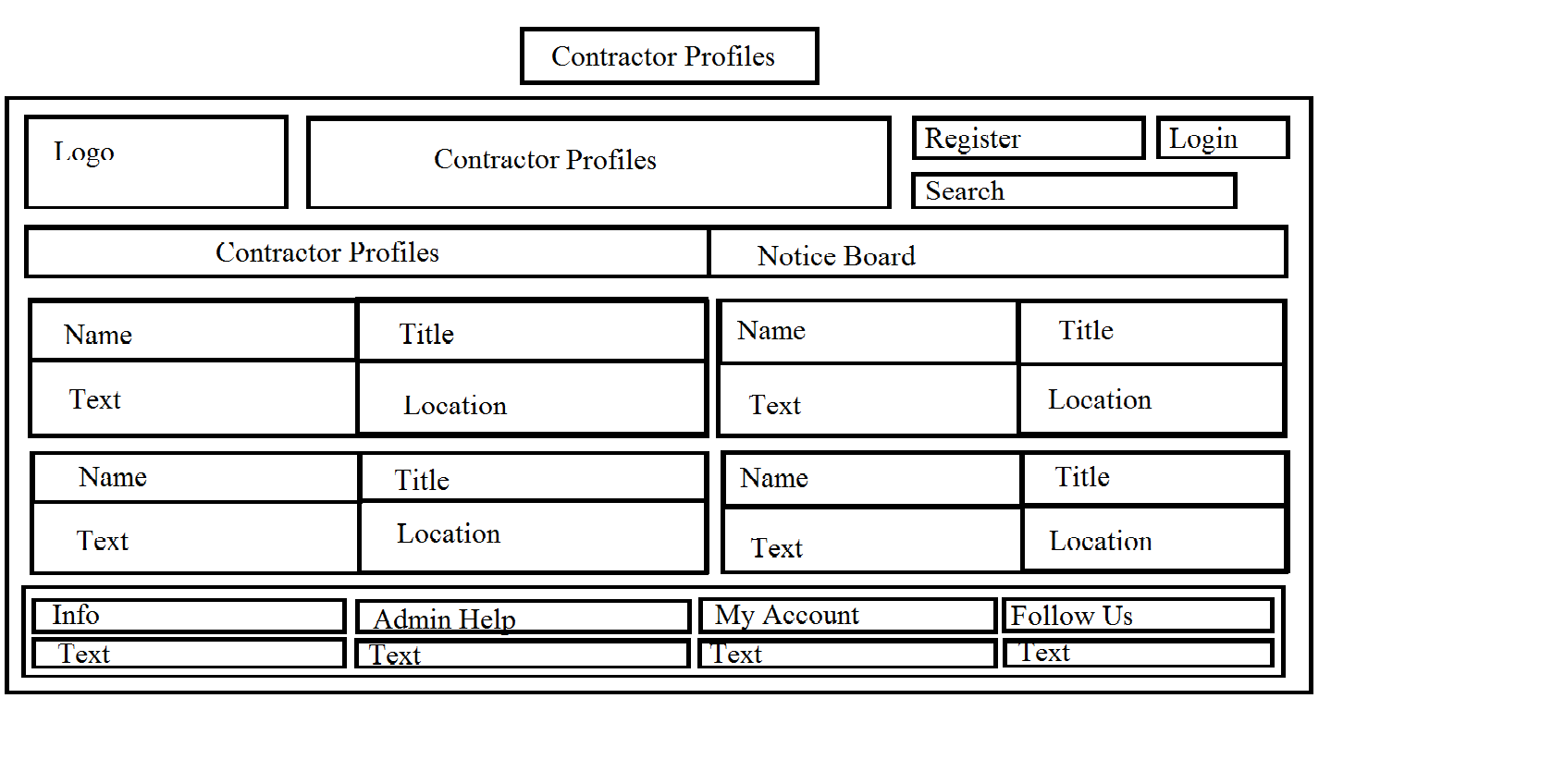
Fig 53. The Contractor Profiles

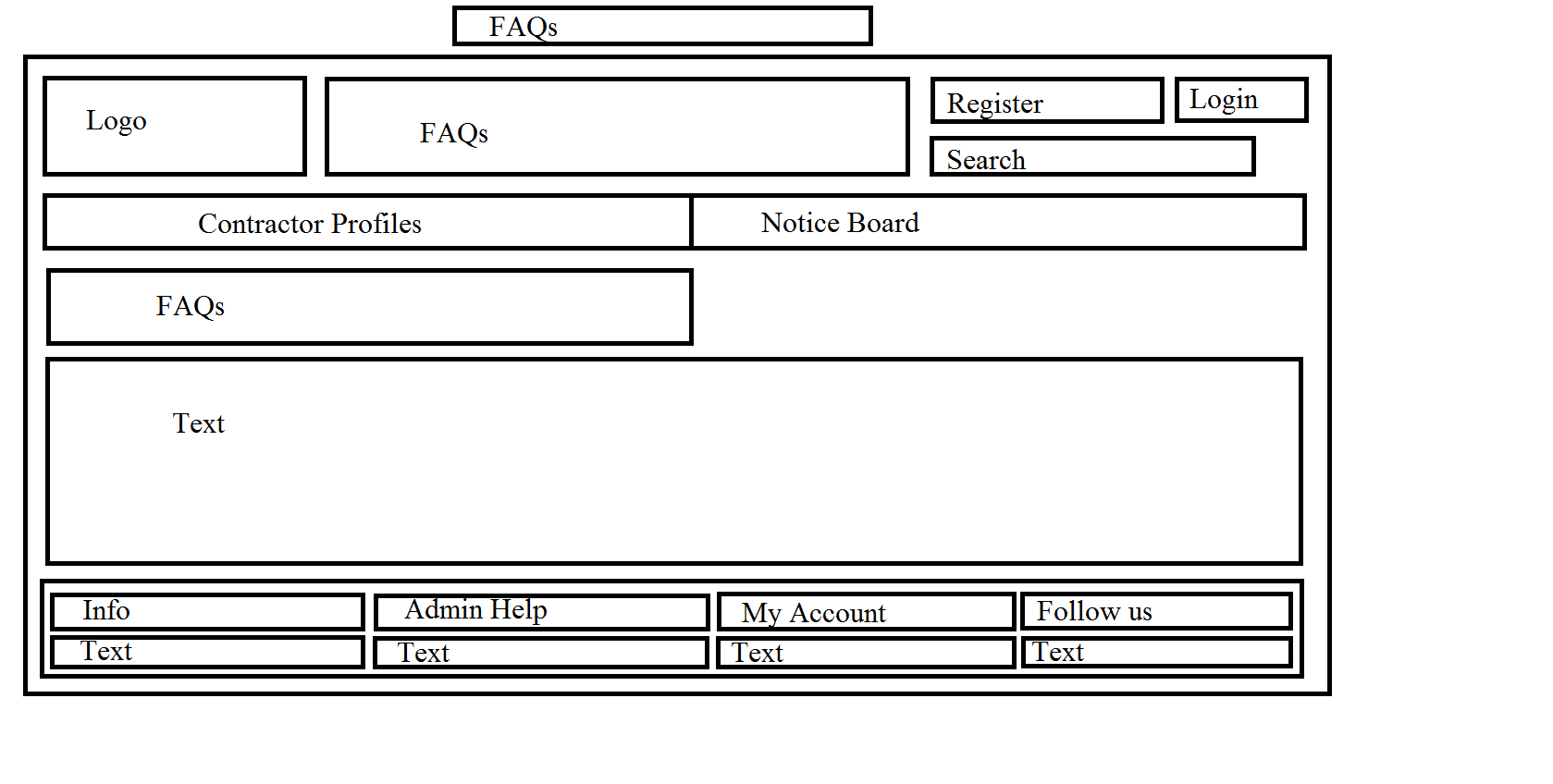
Fig 54. The Help Section

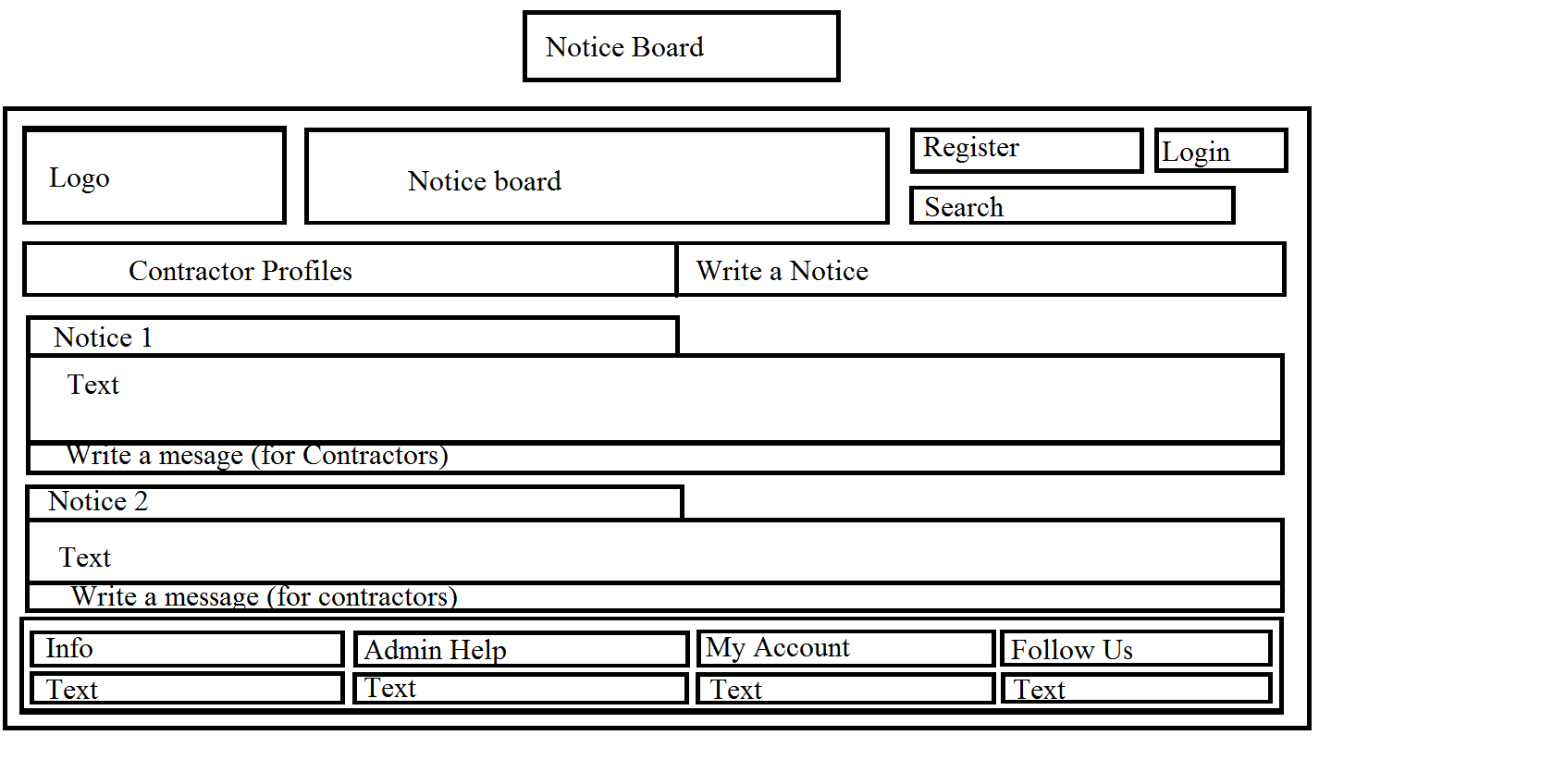
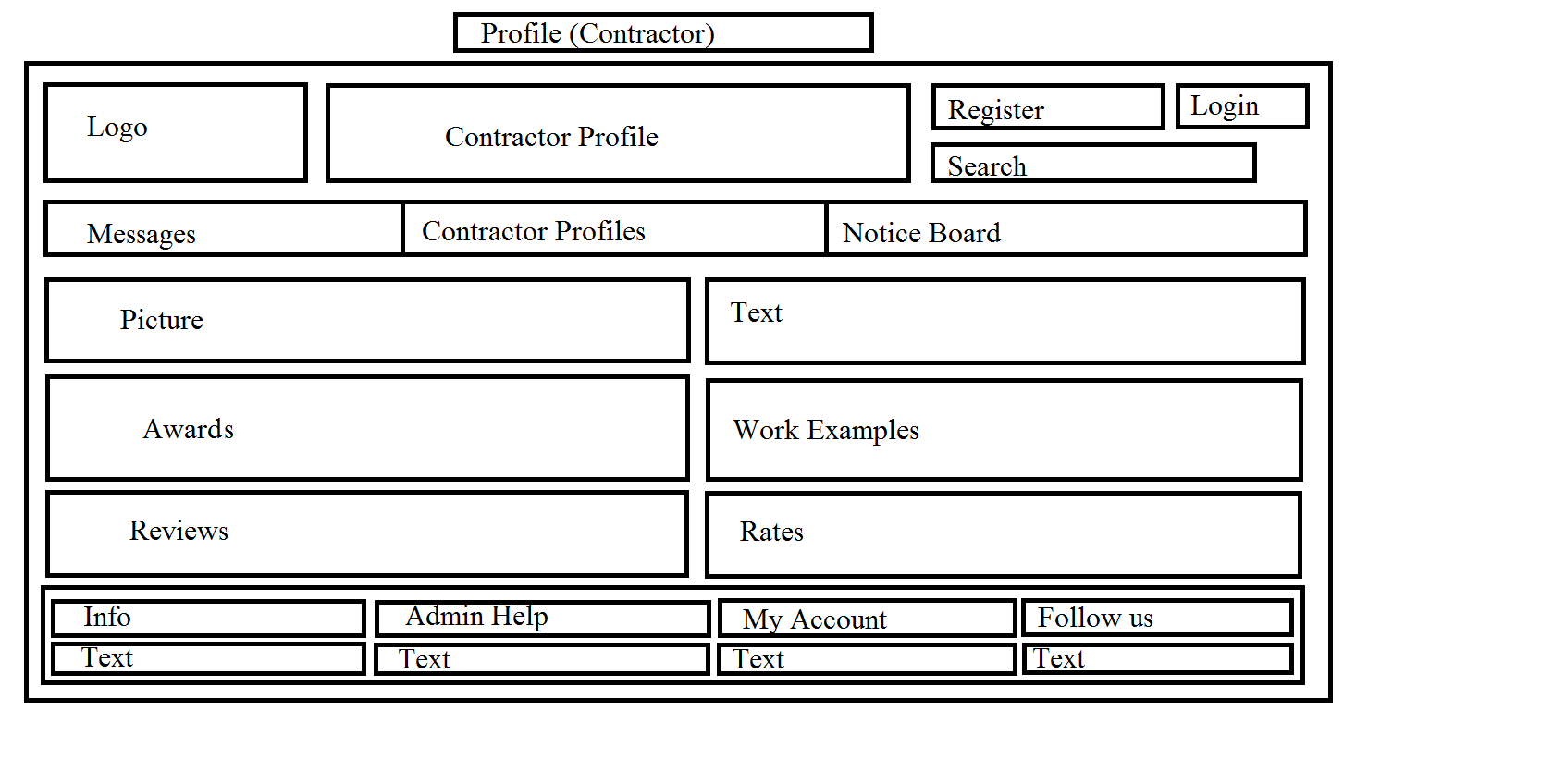
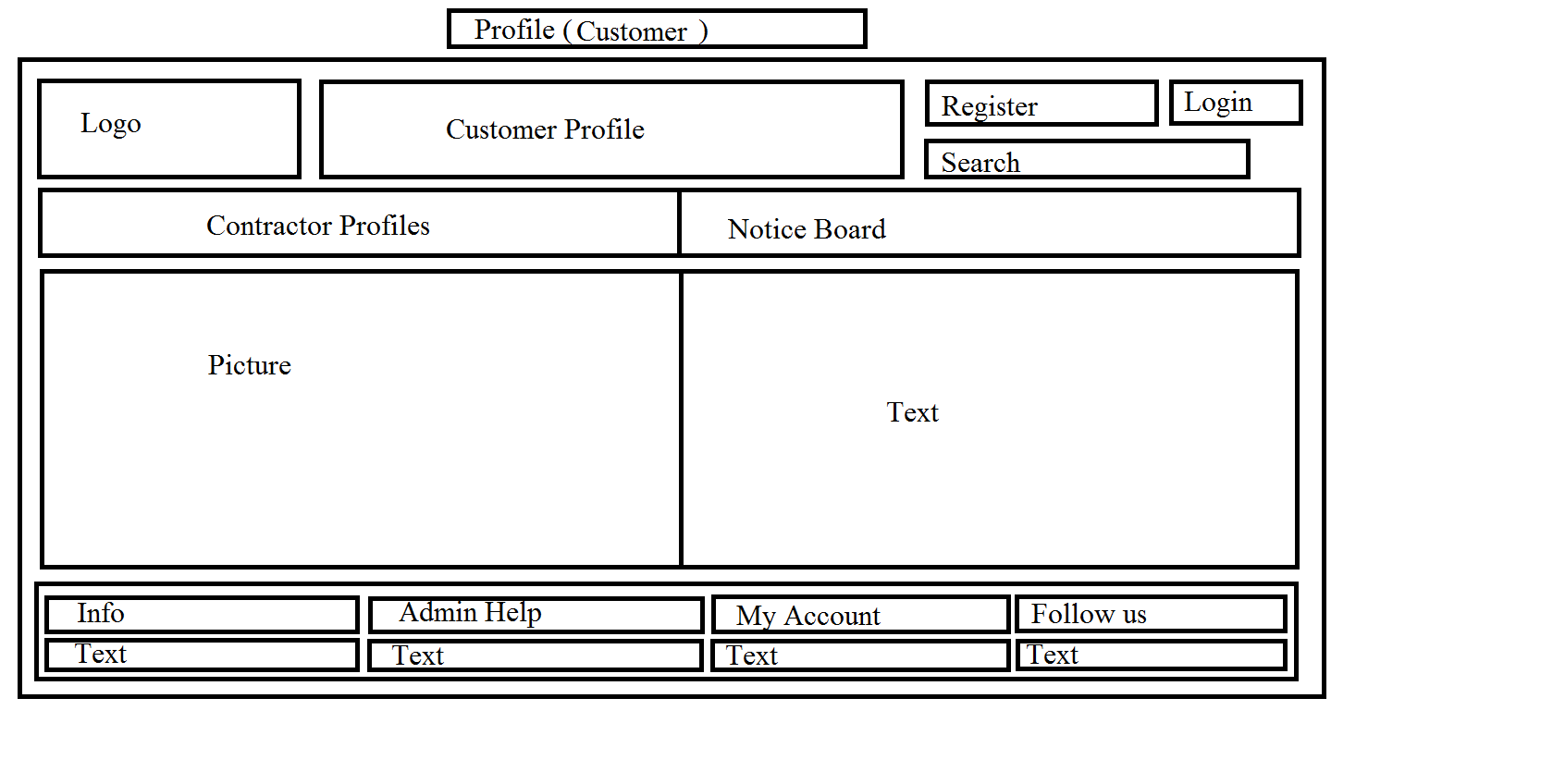
Fig 55. The notice Board

Fig 55. The contractor Profile.Fig 56. The customer profile.

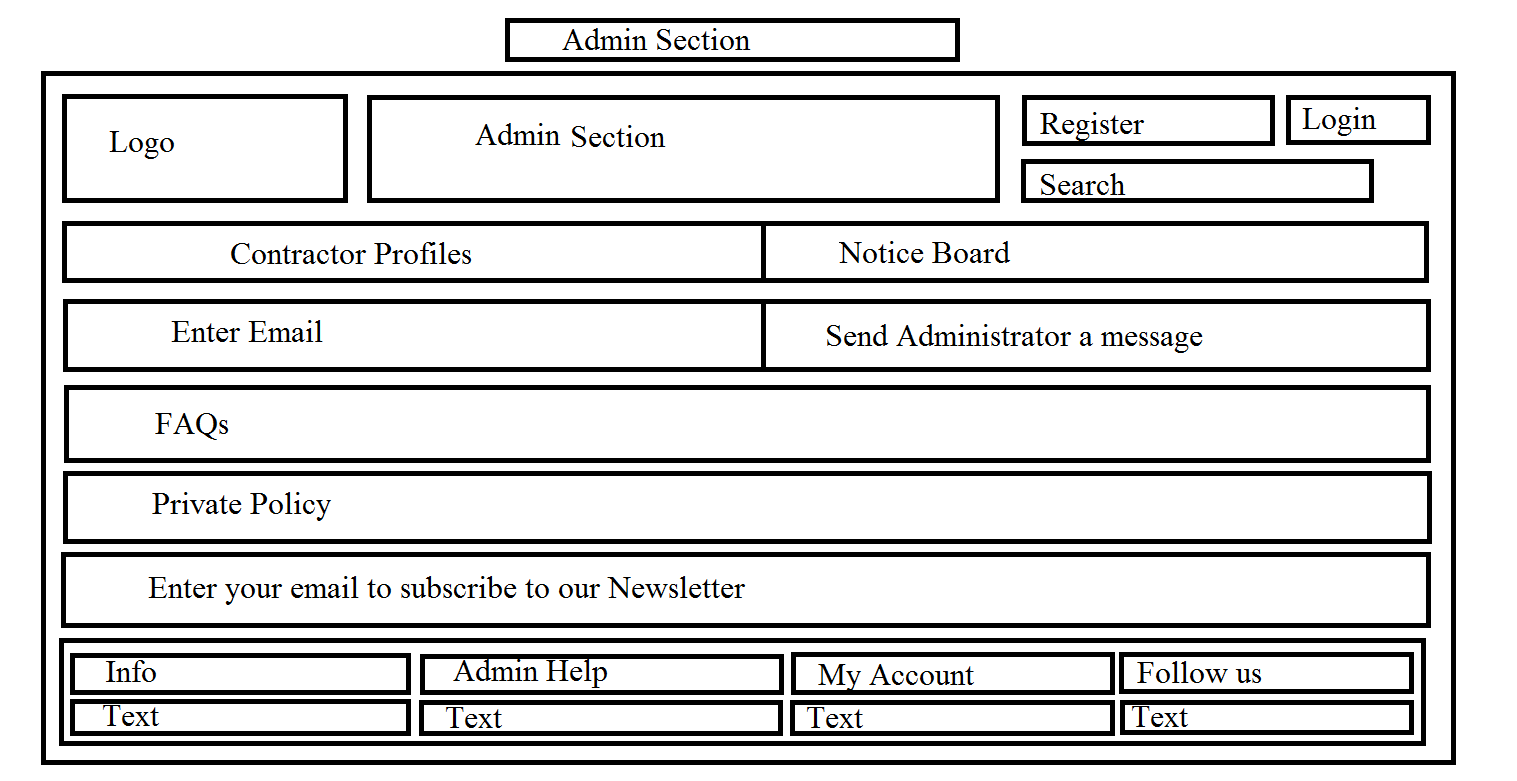


Fig 56. The Admin Section.

# System Architecture

Fig 57. The class Diagram.

# System Evolution

The initial Web application will be limited in its functionality. The main goal will be to get the functionalities above in good working order, with clean code that is easy to edit and update, and runnable so that the website can go live so that the Web Application can get users. Priority will be to make it easy to use and versatile to make it user friendly. It should also be aesthetically pleasing. The Web Application should be usable on a mobile, PC and tablet. Later versions will improve the messaging system, the security, the breakdown of the Contractor profiles to make the trades more segregated and to improve the maps and location aspect.