National College of Ireland

Postgraduate Diploma in Cloud Computing

2011/2012

Darragh Breathnach

Student ID:11106417

Email: darraghb@gmail.com

Property Management Tool

Project in

Web Application Frameworks



I hereby certify that this material, which I now submit for assessment of the programme of study leading to the award of Postgraduate Diploma in Cloud Computing is entirely my own work and has not been taken from the work of others save and to the extent that such work has been citied and acknowledged within the text of my work.

Signed: ………………………………

Date:

Student Number: 11106417

**Table of Contents**

[1. Introduction 4](#_Toc312059049)

[1.1. Motivation 4](#_Toc312059050)

[1.2. Scope 4](#_Toc312059051)

[2. Requirements 6](#_Toc312059052)

[3. Architecture 7](#_Toc312059053)

[4. Operation 10](#_Toc312059054)

[5. Testing 11](#_Toc312059055)

[6. RDoc 12](#_Toc312059056)

[7. Discussion 13](#_Toc312059057)

[References 14](#_Toc312059058)

[Appendix A 15](#_Toc312059059)

[Appendix B - Zip File 17](#_Toc312059060)

# Introduction

This web application is an easy to use property management tool. It’s designed for use by a property managing agent, rather than the owner’s of the properties themselves.

It provides an application to create new development, new units in those developments and new owners for those units. Developments can be mapped using Google Maps, and there is a section to determine whether individual owners have voting rights at AGMs, based on their payment status of their management fee.

## Motivation

My interest in the topic of Property Management arose when I bought my first apartment and became involved in the resident’s committee. I am now a director of our management company and have direct involvement with the managing agent, to ensure the interests of our members are seen to.

As we, the owners, recently took over control of the management company from the developer appointed directors, we have the final say in the running of the development. Therefore, when it comes to voting at AGMs or any general meeting, it is essential only those who contribute to the running of the development, financially, can have their say. The issue of residents falling into arrears with their management fees is an increasing problem.

## Scope

The scope hasn’t changed from Project Proposal stage.

Orginal Project Proposal:

Create an application that allows a user to update the database with the following details:

* Development
* Unit in development (address)
* Onwer of unit
* Contact details of owner
* Management fee paid by owner
* Status of payment (for use in voting rights at meetings)

The user can then display owners based on a chosen development or various other options. The user can determine whether the member/owner is eligible to vote at AGMs based on their payment status of their management fees.

The application will allow the development to be displayed on google maps based on the address.

Those elements have been achieved.

# Requirements

Non-Functional:

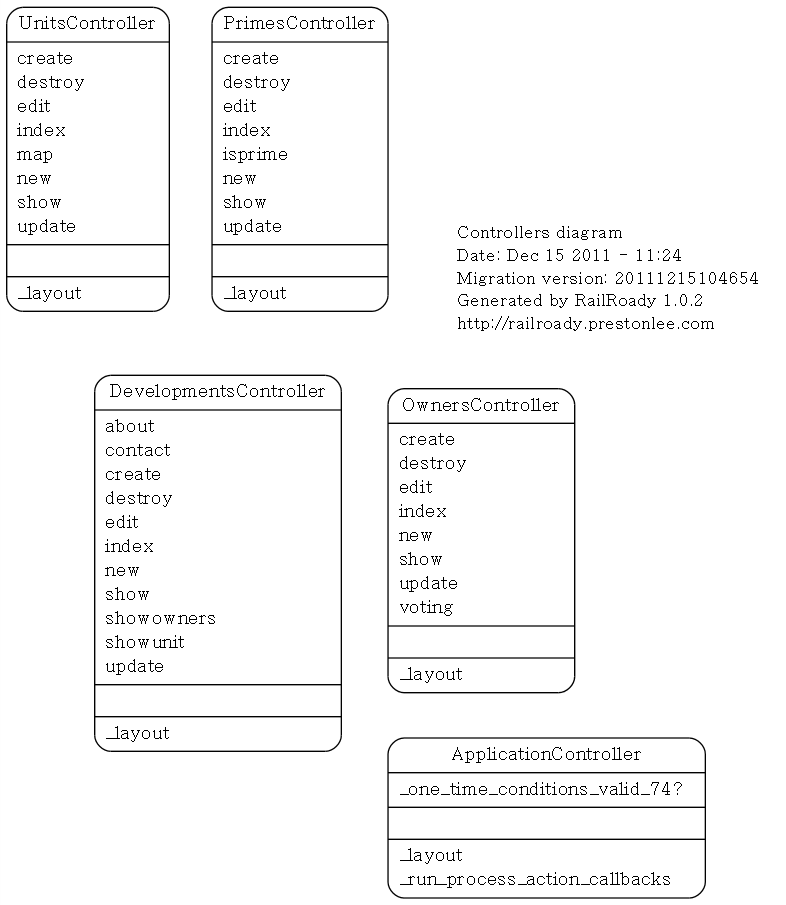
* Create new Developments
* Create new Units
* Create new Onwers

Functional:

* No login required, as this application is not for general use.
* New development created and stored in SQLite database
* CRUD functionality for “developments”, “owners” and “units”.
* New owners can be created and stored in SQLite database
* When “showing” developments, function exists to create a new unit in that development and assign an owner to the new unit, stored in SQLite database.
* Owner details can be displayed, showing their name, address, contact details, management fee, and status of payment.
* Based on a chosen owner, confirm if the owner is in default of management fee, and therefore eligible or not to vote at General Meetings.
* Display development on Google maps using the gem “gmap4rails”.
* Contains a gem created to check whether an inputted number is ‘prime’ or not.

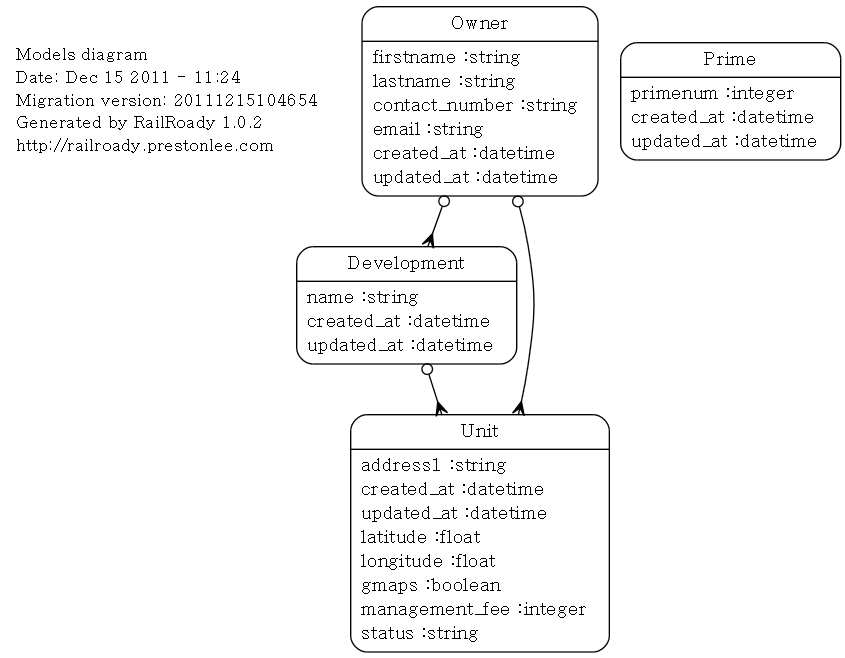
# Architecture

**Controllers:**



* **Units Controller**: Contains CRUD functionality along with the Google maps function and the display function.
* **Development Controller**: Contains CRUD functionality along with displaying functions.
* **Owner Controller**: Contains CRUD and the voting function.
* **Prime Controller**: This is for the function that checks whether an inputted number is prime or not.
* **Application Controller**:

**Models:**



* **Owner:** An owner belongs to a development and a unit. The owner model contains the contact details, which are stored in SQLite.
* **Development:** A development has many units and belongs to an owner. It only contains the name of the development itself.
* **Prime:** Prime isn’t connected to the other models.
* **Unit:** A unit belongs to a development and belongs to an owner. This contains the address of the unit, co-ordinates for displaying this on Google Maps, the management fee charged to that unit, and the status of payment of the management fee. This status is chosen from a drop down box of the following values: “Paid in Full”, “Default”, “DD (Direct Debit)”, and “Payment Plan”.

# Operation

When users open the application, they are on the homepage, which lists the various developments already stored in the database. From here, the user can navigate to all the other sections of the site, including the ‘Owners’ and ‘Voting’ page.

Those developments already stored, and that have units assigned to them with full address details, can be displayed on Google Maps by clicking the ‘Map’ button adjacent to the development.

New developments, and owners can be created from this page, as well as editing, displaying and deleting existing developments.

When displaying existing developments, the user can add new units to that development, or display, edit or delete the details for existing units.

The voting page displays a list of owners who are eligible to vote based on the status of their payment of the management fee. So long as they are not in ‘default’ of payment, they are eligible. This page displays those eligible by unit and by name. For clarity, it also displays those ineligible.

The function to generate those eligible or not is contained in the Owners controller. It generates an array of all the owners whose status is *not* ‘Default’ and another array for those with ‘Default’. This is then outputted on the voting.html.erb page, and displayed in three tables.

As part of the requirements to create a gem for use in the project, I’ve reused the gem we created in the lab. This can be accessed by the “Prme” button on the side of the page. This function determines whether an integer, entered by the user is prime or not.

# Testing

The testing carried out on this application was post and during development, effectively, end user testing.

Test results are contained in Appendix A.

# RDoc

All RDoc generated documentation is contained within the electronic project folder.

See PropertyMnger/doc folder.

# Discussion

Although initial proposal objectives were met, there is scope to improve this application.

In terms of the ‘Voting’ page, this could be enhanced to select a particular development, and view those eligible to cast votes at AGMs from that one development. This would be necessary as the database grew in size.

This application is only designed for use by a property management agent, rather than the owners of the development themselves. A section could be added to provide a forum for residents and a section for them to notify the managing agent of any problems that exist in the development. This would include a log in screen for users. As it stands, log in isn’t required.

A section could be added to include specific details about particular developments, such as their annual budget, maintenance schedules etc.

Currently, I have designed a working solution that can be implemented by Property Managing Agents, at a low level.

The ‘voting rights’ feature is of particular benefit as it gives a quick answer to whether certain residents who attend general meetings should be allowed vote on any items in the agenda.

# References

Google Maps Gem:

<https://github.com/besi/rails-quickies/tree/master/Gmaps4Rails> December 2011.

CSS Template:

<http://www.freecsstemplates.org/preview/plainclean/> December 2011.

Architectue Diagram:

<http://railroad.rubyforge.org/> December 2011.

# Appendix A

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Page: Homepage/Developments** |  |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | 'Show' button clicked | Display units page associated with development | Pass |
|  | 2 | 'Edit' button clicked | Display editing page for development name | Pass |
|  | 3 | 'Destroy' button clicked | Display warning dialog about destroying details | Pass |
|  | 4 | 'Map' button clicked | Display location of development on Google Maps | Pass |
|  | 5 | 'Add new owner' clicked | Display page to create new owner | Pass |
|  | 6 | 'New Development' clicked | Display pag to create new development | Pass |
|  | 7 | 'Homepage' button clicked | should always bring you to development listing page | Pass |
|  | 8 | 'Voting' button clicked | Display voting eligibility page | Pass |
|  | 9 | 'About' button clicked | Displays 'about' page, containing instructions | Pass |
|  | 10 | 'Owners Details' button clicked | Displays owners details | Pass |
|  | 11 | 'Developments' button clicked | same as 'Homepage' | Pass |
|  | 12 | 'Contact' button clicked | Displays contact details | Pass |
|  | 13 | 'Prime' button clicked | Displays page with function to check prime numbers | Pass |
|  |  |  |  |  |
|  |  | **Page: Voting** |  |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Test all page links | Display appropriate page | Pass |
|  |  |  |  |  |
|  |  | **Page: About** |  |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Test all page links | Display appropriate page | Pass |
|  |  |  |  |  |
|  |  | **Page: Owners Details** |  |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Test all page links | Display appropriate page | Pass |
|  | 2 | 'New Owner' button clicked | Display page to create new owner | Pass |
|  | 3 | 'Show' button clicked | Display owner details | Pass |
|  | 4 | 'Edit' button clicked | Display editing page for owner details | Pass |
|  | 5 | 'Destroy' button clicked | Display warning dialog about destroying details | Pass |
|  |  |  |  |  |
|  |  | **Page: Contact** |  |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Test all page links | Display appropriate page | Pass |
|  |  |  |  |  |
|  |  | **Page: Prime** |  |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Test all page links | Display appropriate page | Pass |
|  | 2 | Enter various numbers to check whether prime | Display if prime or not | Pass |
|  |  |  |  |  |
|  |  | **Page: http://0.0.0.0:3000/owners/new** | |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Enter owner details and click 'Create Owner' | New owner created | Pass |
|  | 2 | 'Back' button clicked | Return to previous screen | Pass |
|  |  |  |  |  |
|  |  | **Page: http://0.0.0.0:3000/developments/new** | |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Enter new development details and click 'Create Development' | New development created | Pass |
|  |  |  |  |  |
|  |  | **Page: http://0.0.0.0:3000/developments/1** | |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Click 'Add Unit' | Display page to enter new unit details | Pass |
|  | 2 | Click 'Edit' | Allow editing of Development name | Pass |
|  | 3 | Click 'Back' | Return to previous page | Pass |
|  |  |  |  |  |
|  |  | **Page: http://0.0.0.0:3000/developments/1/units/new** | |  |
|  | Number | Test | Expected Outcome | Result |
|  | 1 | Enter new unit details and click 'Create Unit' | New unit created | Pass |
|  | 2 | Click 'Back' button | Return to previous page | Pass |

# Appendix B - Zip File

The zipped file uploaded to moodle for the project contains all the folders generated in the Ubuntu environment, including Rdoc files.

The RoR project is called “PropertyMngr”.

This report is also available in the folder in a folder called “Project Report”.